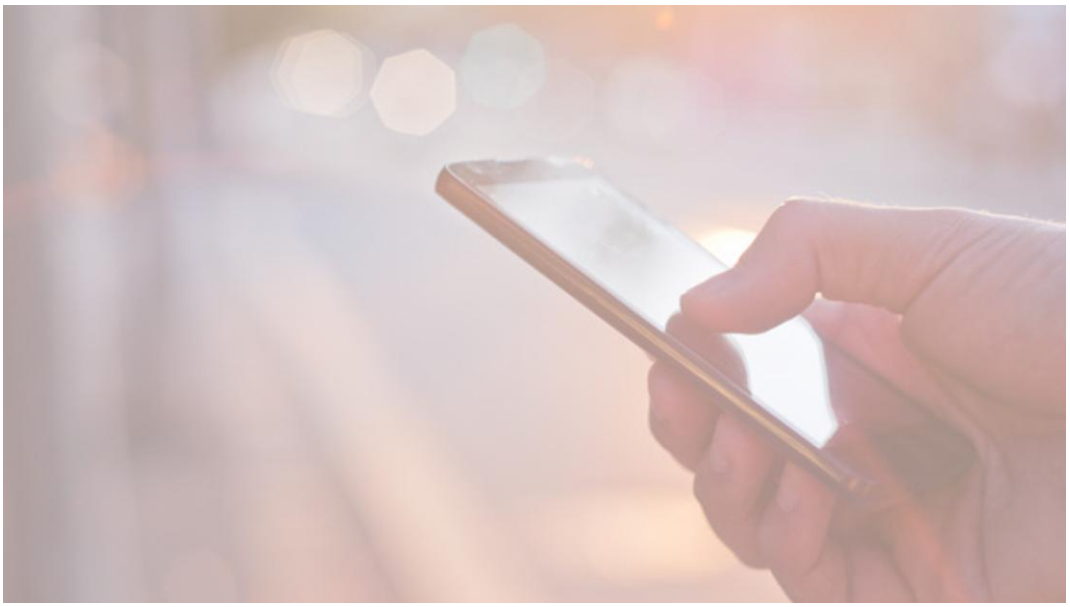
SOUBEYRAND Adrien

Juin 2018

AFPA Rhone-Alpes CCP1 : Developpeur Client Serveur

DOSSIER TECHNIQUE(DT)

Application gestion de contact téléphoniques : Projet Memen'tel



---------- AFPA 2018 ----------

SOMMAIRE

Table des matières

1) Contexte

Un répertoire téléphonique permet d’avoir ses contacts à portée de main, de répertorier les informations qui les concernent tel que le nom, le numéro de téléphone, l'adresse et de les retrouver facilement, en général de manière alphabétique.

Le format numérique est intéressant car permet de:

- faciliter les recherches de ses contacts.
- Mettre à jour des contacts sans contraintes.
- Gagner de la place et d'avoir un caractère écologique.

****

2) Cahier des charges :

Afin d'offrir des fonctionnalités utiles et pratiques pour l'utilisateur, le cahier des charges doit répondre aux points suivants :

- L'application doit proposer des écrans simples et intuitifs, en conséquence peu surchargés visuellement
- Elle doit permettre d'ajouter des contacts, de mettre à jour les informations de ces derniers ou de supprimer un contact au besoin. Les informations obligatoires sont le nom et le prénom. Les informations facultatives sont les numéros, les adresses, la date de naissance, la photo et le pseudonyme.
- L'utilisateur doit pouvoir créer des catégories et regrouper des contacts par ce moyen. A titre d'exemple, l'utilisateur pourra créer les catégories: famille, amis, connaissances, société, restaurant, etc.
- Le contact peut être professionnel. Dans ce cas il sera possible d'ajouter un complément d'informations spécifiques à ce cas. Un professionnel possédera plus d'informations tels que le fax, les numéros professionnels, l'adresse de société, etc.

Les points cités permettront donc d'avoir des informations variées concernant les contacts et de les trier facilement.

****

3) Choix technologique :

Suite à l'étude du cahier des charges, j'ai choisis de développer l'application en C#, avec interface graphique en Windows Forms (intégrée dans le Framework ".NET").

C# permet le développement d'application lourde avec l'utilisation de l'infrastructure logicielle que forme ce Framework de Microsoft.

Un gestionnaire de données en MySQL hébergera la base de données de l'application.

****

4) Conception de la Base de Donnée :

Règles de gestions

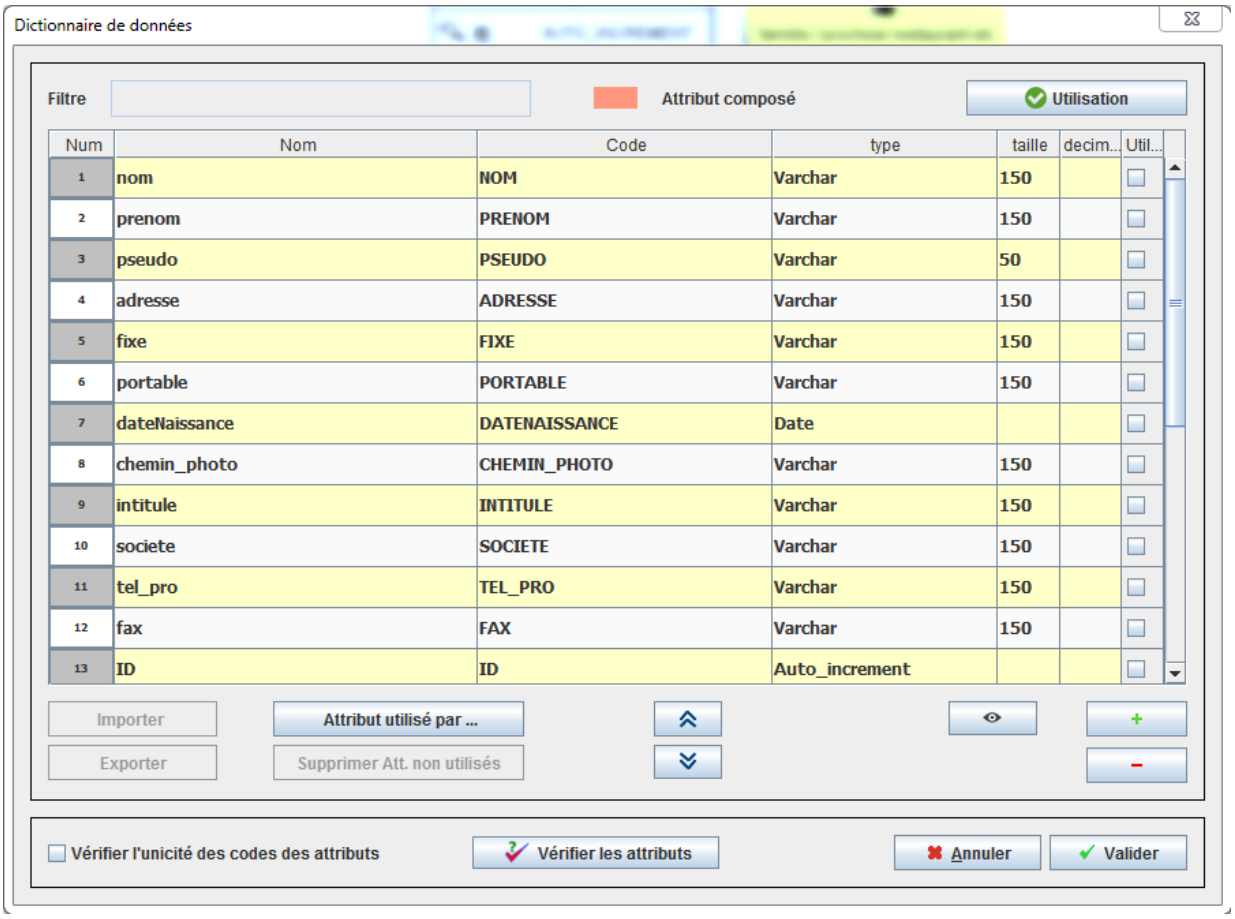
Un contact peut faire partie d'aucune, une ou plusieurs catégories. Et une même catégorie peut être assignée à aucun, un ou plusieurs contacts.

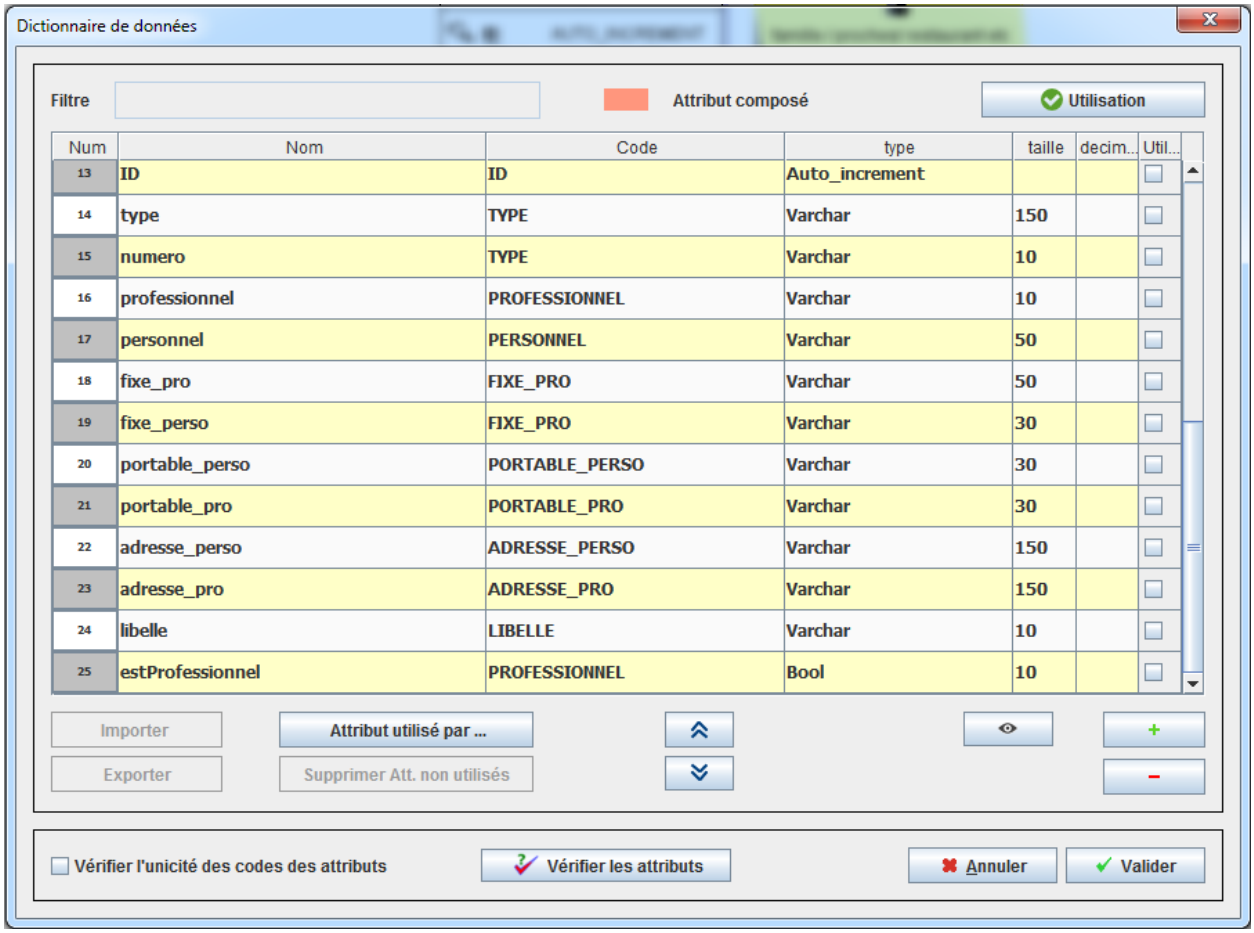
Un contact peut avoir aucune, une ou au maximum deux adresses : l'adresse personnelle et l'adresse professionnelle

Un contact peut posséder aucun, un ou au maximum cinq numéros de téléphone : le numéro fixe personnel, le numéro de fixe professionnel, le numéro de portable personnel, le numéro de portable professionnel, le numéro de fax.

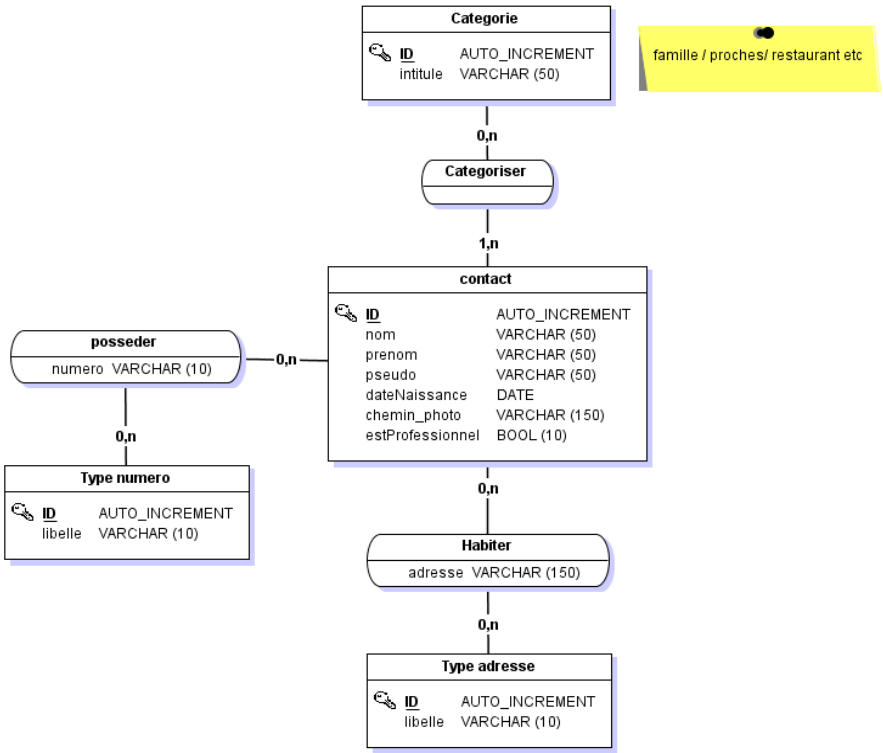
Dictionnaire de données

Les règles de gestion préétablies permettent de définir le contenu du dictionnaire de données. Il contient le vocabulaire de la nomenclature des données, le code correspondant, le type (auto incrémenté, caractères, date, entiers, etc.) ainsi que la taille du nombre de caractères qui sera autorisé dans la base de données.





Modèle Conceptuel de Données (JMerise)

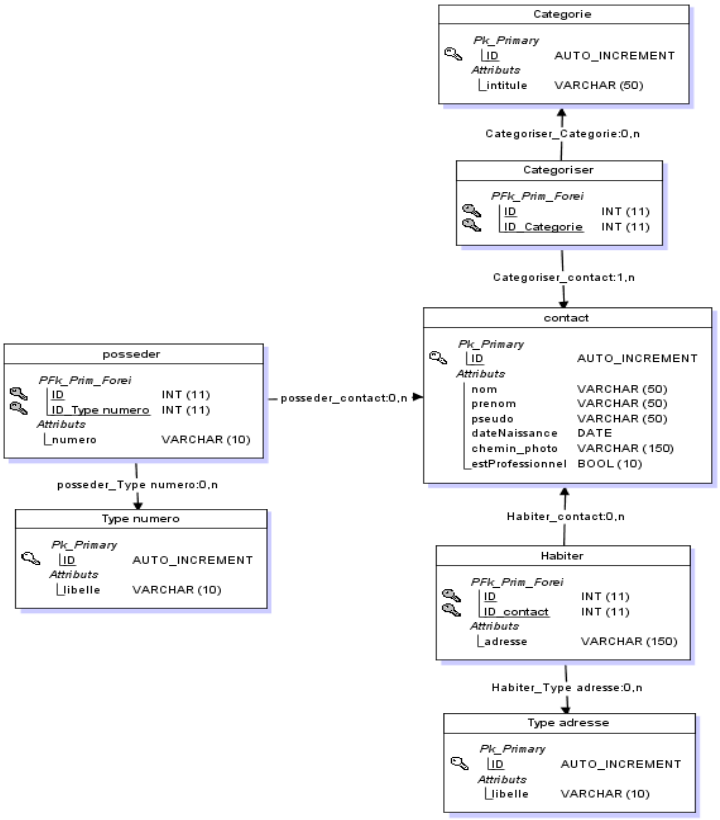


A partir du dictionnaire de données est établi le MCD, la description graphique du modèle des données. L'entité Contact est centrale dans cette base, elle est associée :

- à la table Categorie, destinée à recevoir les catégories qui seront définies par l'utilisateur. Elle est mise en relation avec Contact par l'intermédiaire de la table d'association Categoriser.
- à la table Type numero qui répertorie les différents type de numéro de téléphone : Fixe personnel et professionnel, portable personnel et professionnel et fax. Elle est mise en relation avec Contact par l'intermédiaire de la table d'association Posseder.
- à la table Type adresse qui répertorie les différents types d'adresse : personnel et professionnel. Elle est mise en relation avec Contact par l'intermédiaire de la table d'association Habiter.

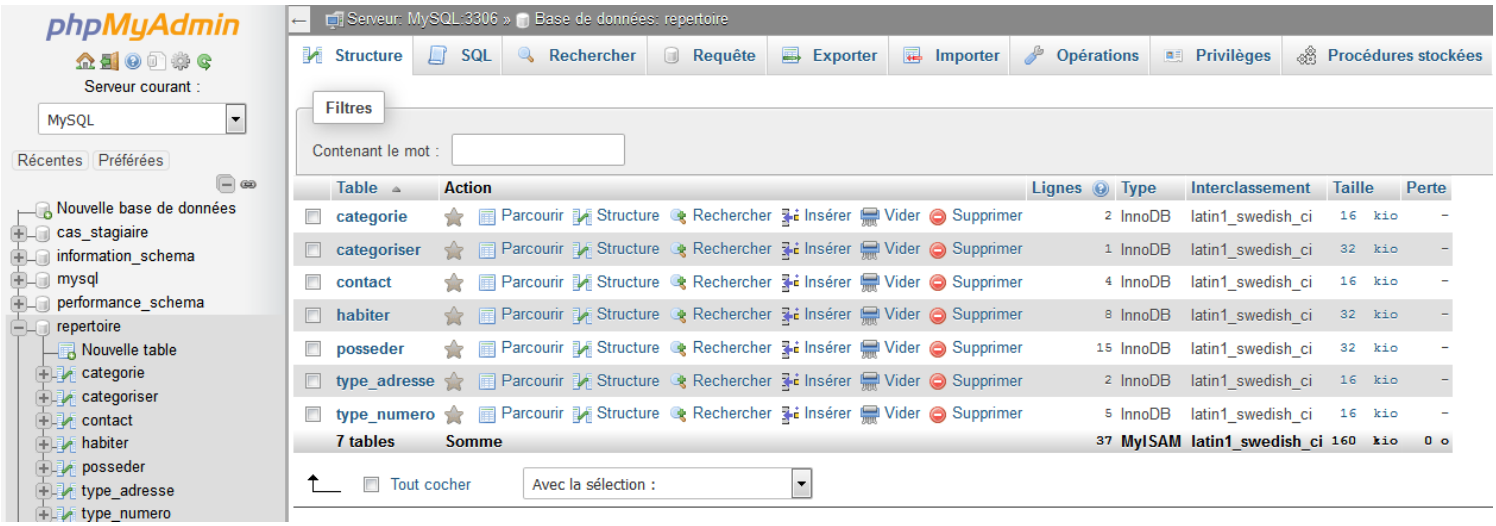
Modèle Logique des Données (Jmerise)

Après conversion, le logiciel nous fournit le MLD ainsi que le script MySQL qui serviront à établir la base de donnée au sein de phpMyAdmin



```
#------------------------------------------------------------

# Table: contact

#------------------------------------------------------------

CREATE TABLE contact(

      ID             Int  Auto_increment  NOT NULL ,

      nom            Varchar (50) NOT NULL ,

      prenom         Varchar (50) NOT NULL ,

      pseudo         Varchar (50) NOT NULL ,

      dateNaissance    Date NOT NULL ,

      chemin_photo     Varchar (150) NOT NULL ,

      estProfessionnel Bool NOT NULL

          ,CONSTRAINT contact_PK PRIMARY KEY (ID)

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Categoriser

#------------------------------------------------------------

CREATE TABLE Categoriser(

      ID         Int NOT NULL ,

      ID_Categorie Int NOT NULL

          ,CONSTRAINT Categoriser_PK PRIMARY KEY (ID,ID_Categorie)

          ,CONSTRAINT Categoriser_contact_FK FOREIGN KEY (ID) REFERENCES contact(ID)

          ,CONSTRAINT Categoriser_Categorie0_FK FOREIGN KEY (ID_Categorie) REFERENCES
Categorie(ID)

)ENGINE=InnoDB;
```

Légende : extrait du script MySQL



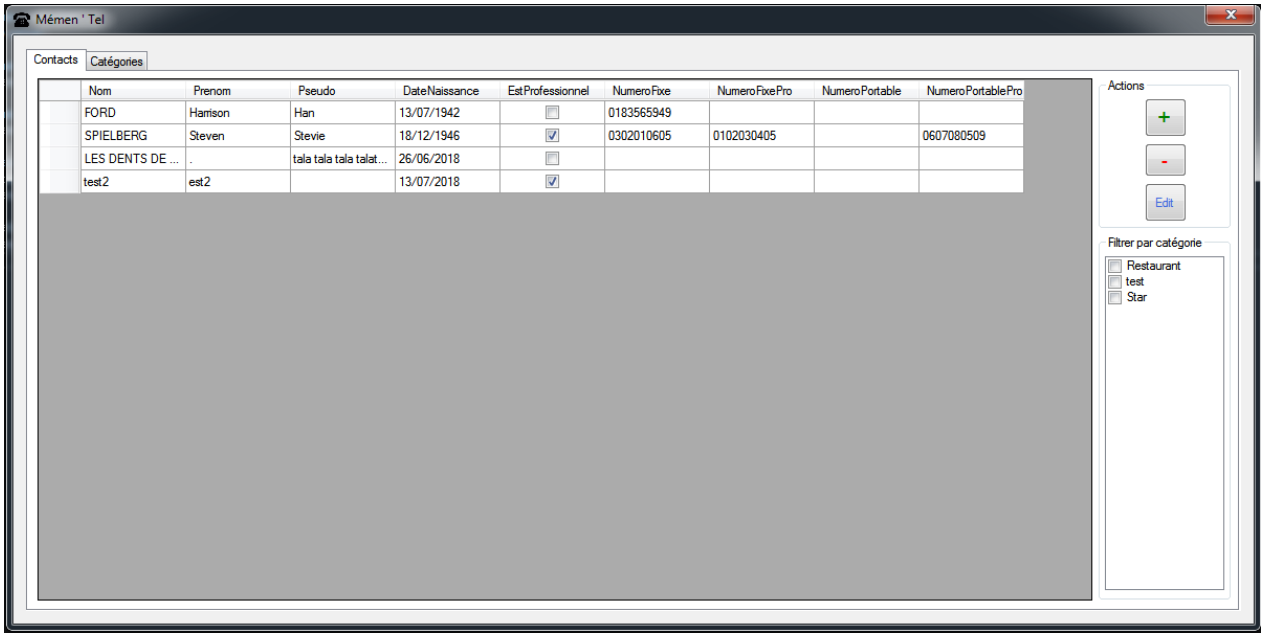
La base de données est établie sur WAMP en phpMyAdmin.

****

5) Interface graphique :

La fenêtre principale propose 2 onglets, le premier affiche la liste des contacts avec les actions qui leur sont liées et le second onglet affiche la liste des catégories et leurs actions.

Onglet Contact



Légende : interface de la liste des contacts

Description de l'interface :

- la grille des contacts avec les entêtes Nom, Prénom, Pseudo, Date de Naissance, Si professionnel, Numéro Fixe, Numéro Portable, Numéro Fixe Professionnel, Numéro Portable Professionnel, Fax.
- les boutons d'ajout, d'édition et de suppression déclencheront des actions.
- La liste des catégories avec les cases à cocher déclenchera l'action de filtre sur la grille.

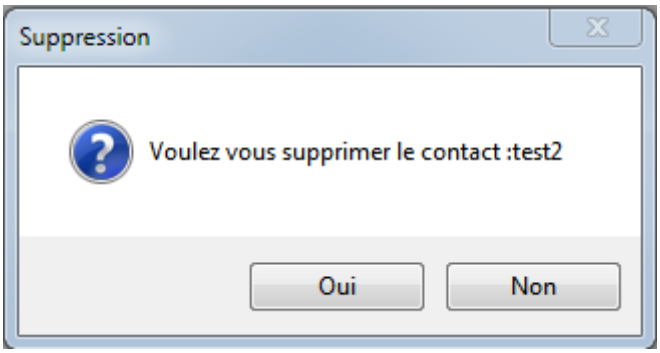
Actions :

- Sur le clic du bouton d'ajout ouvre la fenêtre d'ajout de contact.
- Sur le clic du bouton d'édition ouvre la fenêtre de modification du contact sélectionné.
- Sur le double clic d'une ligne de la grille des contacts aura la même action que le clic sur le bouton d'édition.
- Sur le clic du bouton de suppression supprime le contact sélectionné.
- Sur la coche ou décoche d'une ou plusieurs catégorie filtre les contacts à afficher suivant les catégories auxquelles ils sont rattachés.

Contact affiche les contacts et leurs informations(général, avec les actions disponibles : ajouter, modifier et supprimer (zoom). Une listCheckedBox permet de sélectionner une ou plusieurs catégorie auxquelles sont rattachés les contacts (2 écrans).

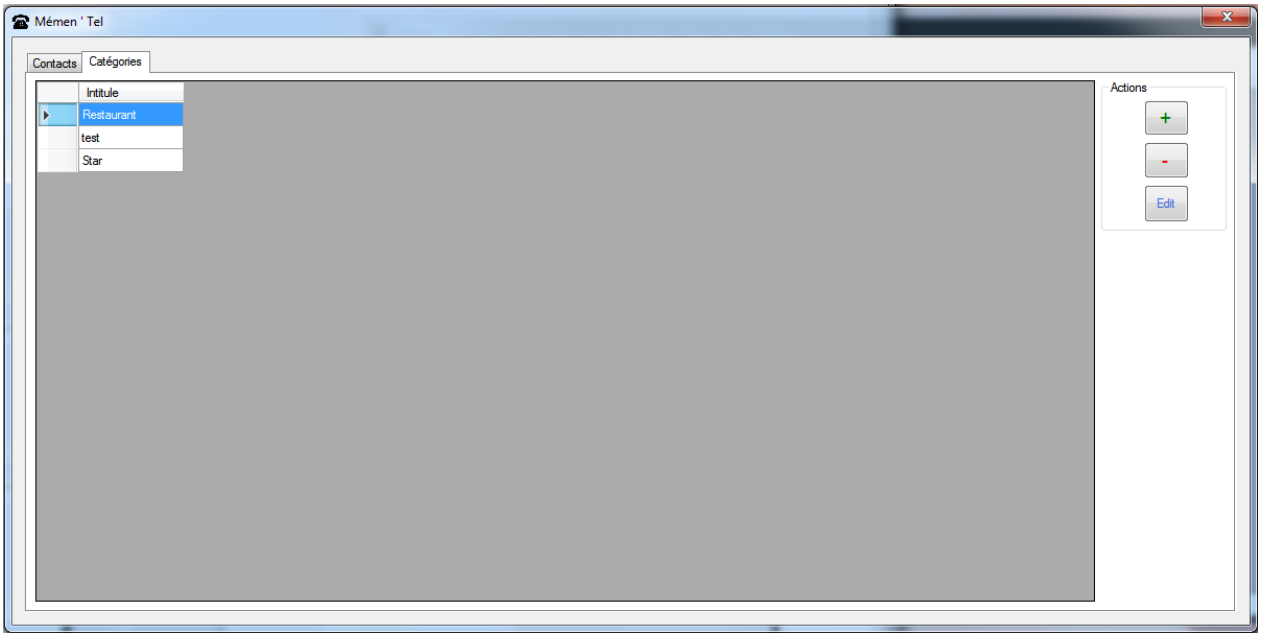
Ajouter et modifier ouvrent le formulaire ajouter/éditer un contact (image) qui présentent les mêmes contrôles graphiques. Lorsqu'on ajoute, le formulaire apparaît vierge et il est nécessaire d'enregistrer un nom et un prénom obligatoirement. Lorsqu'on modifie, les données sont déjà remplies.

Supprimer efface le contact de la base de donnée, une fenêtre s'ouvre et demande la confirmation (image).



Légende : fenêtre de confirmation de la suppression du contact

Onglet catégorie



Légende : interface de la liste des catégories

Description de l'interface :

- la grille des catégories avec l'entête qui correspond au libellé de la catégorie.
- les boutons d'ajout, d'édition et de suppression déclencheront des actions.

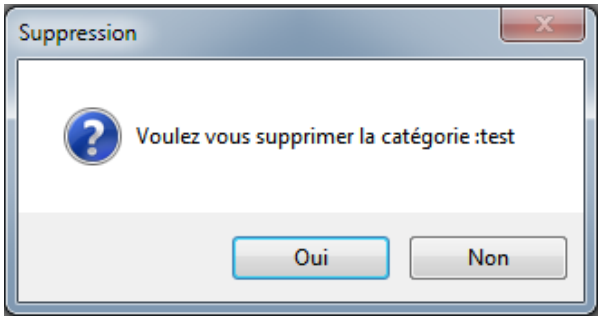
Actions :

- Sur le clic du bouton d'ajout ouvre la fenêtre d'ajout de catégorie
- Sur le clic du bouton d'édition ouvre la fenêtre de modification de la catégorie sélectionnée.
- Sur le clic du bouton de suppression supprime la catégorie sélectionnée.

Catégorie (image) : affiche les catégories avec les boutons d'actions (semblables à ceux de contacts) ajouter, modifier, supprimer (zoom).

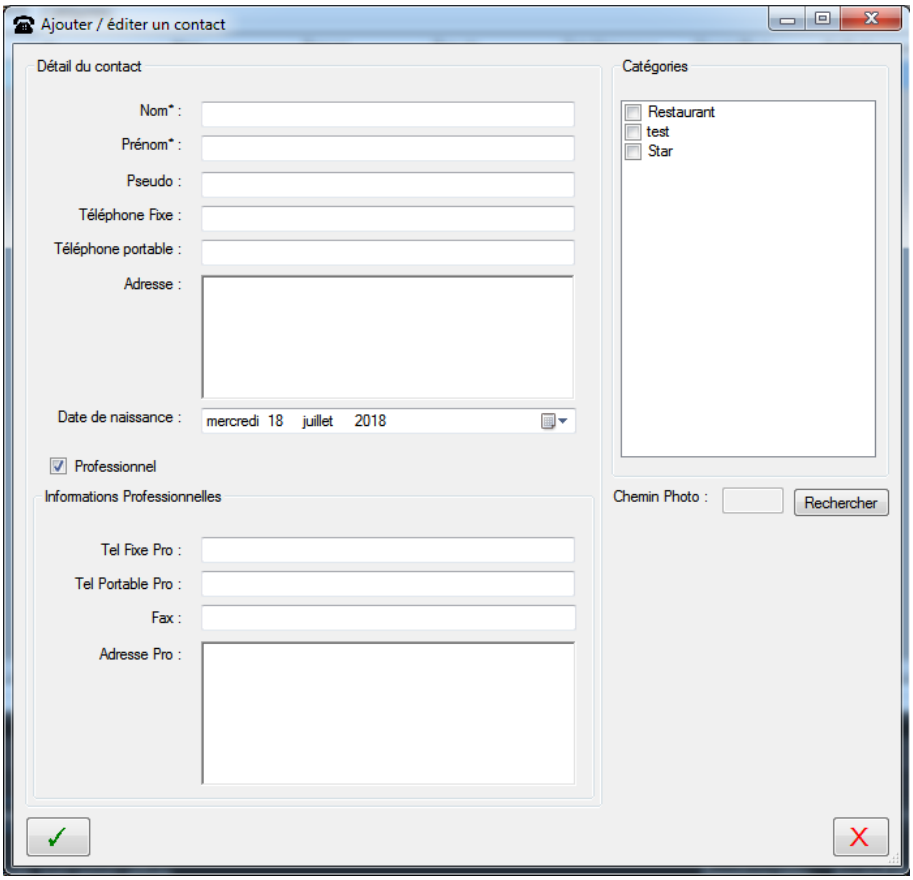
Ajouter ou modifier ouvrent le même formulaire (image)

Supprimer: Le clic sur le bouton affiche la fenêtre de confirmation de suppression et si l'utilisateur valide la demande : on supprime de la base de donnée la catégorie, on supprime l'objet de l'application. Pour finir on met à jour la grille des contacts ou des catégories.



Légende : formulaire de confirmation de la suppression de la catégorie

formulaire d'ajout/édition d'un contact



Légende : interface du formulaire

Description de l'interface :

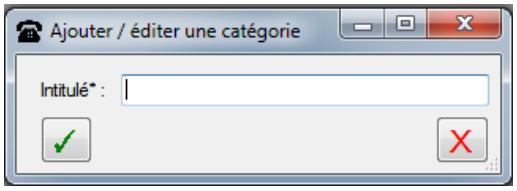
- le champ texte "nom" permet d'afficher et de modifier le nom du contact édité ou d'en créer un nouveau.
- le champ texte "prénom" permet d'afficher et de modifier le prénom du contact édité ou d'en créer un nouveau.
- le champ "pseudo" permet d'afficher et de modifier le pseudo du contact édité ou d'en créer un nouveau.

- le champ "téléphone fixe" permet d'afficher et de modifier le numéro fixe personnel du contact édité ou d'en créer un nouveau.
- le champ "téléphone portable" permet d'afficher et de modifier le portable personnel du contact édité ou d'en créer un nouveau.
- le champ "adresse" permet d'afficher et de modifier l'adresse du contact édité ou d'en créer une nouvelle.
- le champ "pseudo" permet d'afficher et de modifier le pseudo du contact édité ou d'en créer un nouveau.
- le champ "date de naissance" permet d'afficher et de modifier la date de naissance du contact édité ou d'en créer un nouveau. Sans date choisie, c'est la date du jour qui est choisie par défaut.
- Dans la GroupBox "informations professionnelles", le champ "tel fixe pro" permet d'afficher et de modifier le fixe professionnel du contact édité ou d'en créer un nouveau.
- Dans la GroupBox "informations professionnelles", le champ "tel portable pro" permet d'afficher et de modifier le portable professionnel du contact édité ou d'en créer un nouveau.
- Dans la GroupBox "informations professionnelles", le champ "fax pro" permet d'afficher et de modifier le fax du contact édité ou d'en créer un nouveau.
- Dans la GroupBox "informations professionnelles", le champ "adresse pro" permet d'afficher et de modifier le portable professionnel du contact édité ou d'en créer un nouveau.

Actions :

- Sur le clic du bouton "valider" : Si le nom ou le prénom sont vides, une boite de dialogue invitant l'utilisateur à remplir ces champs est affichée. Si le nom et le prénom ne sont pas vides alors on met à jour ou on crée le contact dans la base de donnée, on crée l'objet contact dans l'application et on ferme le formulaire d'ajout/édition du contact
- Sur le clic du bouton "Annuler" : Ferme le formulaire d'ajout édition de contact.
- Le clic sur le bouton "rechercher" : ouvre la boite de dialogue de recherche d'image (types acceptés : PNG, JPG, BMP). Suite à la sélection d'une image, celle-ci est affichée en miniature en bas à droite du formulaire et le chemin de la photo est enregistré dans le champ texte "chemin photo".
- Sur la coche, affiche la GroupBox "informations professionnelles".

formulaire d'ajout/édition d'une catégorie



Légende : interface du formulaire

Description de l'interface :

- le champ texte "intitulé" permet d'afficher et de modifier le nom de la catégorie éditée ou d'en créer un nouveau.

Actions :

- Sur le clic du bouton "valider" : Si l'intitulé est vide, une boite de dialogue invitant l'utilisateur à remplir ces champs est affichée. Si l'intitulé n'est pas vides alors on met à jour ou on crée la catégorie dans la base de donnée, on crée l'objet catégorie dans l'application et on ferme le formulaire d'ajout/édition de la catégorie
- Sur le clic du bouton "Annuler" : Ferme le formulaire d'ajout édition de catégorie.

6) Programmation :

Chargement des contacts depuis la base de données

```
        /// <summary>
        /// initialise tous les contacts depuis la base de données
        /// </summary>

        public static void ChargerContact()
        {
            //efface la liste suite à suppression
            Program.ListContact.Clear();

            string requete = "SELECT * FROM contact";
            //créée la commande de requête avec l'instance de connexion à la BDD
            MySqlCommand commande = new MySqlCommand(requete, GetConnexion());
            //Appel de la méthode ExecuteReader de la variable Commande de
//typeMySQLCommande et qui retourne une instance de MySQLDataReader qui sera déclaré
            // dans la variable DataReader
            MySqlDataReader dataReader = commande.ExecuteReader();

            //parcourt de toutes les lignes retournées par la requête
            while (dataReader.Read())
            {
                //déclare toutes les variables correspondantes aux champs d'une ligne
                int id = dataReader.GetInt32("ID");
                string nom = dataReader.GetString("nom");
                string prenom = dataReader.GetString("prenom");
                string pseudo = dataReader.GetString("pseudo");
                DateTime dateNaissance = dataReader.GetDateTime("dateNaissance");
                string cheminPhoto = dataReader.GetString("chemin_photo");
                bool estProfessionnel = dataReader.GetBoolean("estProfessionnel");

                //instanciation d'un nouveau contact avec les données de la ligne
                Contact contact = new Contact(id, nom, prenom, pseudo, dateNaissance,
cheminPhoto, estProfessionnel);

                //on ajoute l'instance créée dans la liste de persistance
                Program.ListContact.Add(contact);

            }
            //ferme la requête
            dataReader.Close();
            //cloture la connexion
            CloseConnexion();

        }
```

Légende : extrait de code

L'extrait ci dessus permet d'instancier pour chaque contact de la base de donnée un nouvel objet contact qui sera ajouté à la liste de persistance : Program.ListContact. Cette liste est accessible partout dans l'application et est utilisée pour afficher la grille des contacts, pour filtrer les contacts par catégorie, etc.

Extrait de code des constructeurs de AjouterEditContact

```
        public AjouterEditContact()
        {
            InitializeComponent();
            initialisation();
            gbxProfessionnel.Hide();
            modification = false;

        }
        /// <summary>
        /// Surcharge du constucteur
        /// </summary>
        /// <param name="contact">le contact à modifier</param>
        public AjouterEditContact(Contact contact)
        {
            InitializeComponent();
            initialisation();
            modification = true;
                                    contactToUpdate = contact;
```

Légende : extrait de code

L'extrait ci dessus présente le constructeur de la Form AjouterEditContact et une surcharge du constructeur. Lorsqu'on instancie ce formulaire en ne passant aucun contact en paramètre, on passera dans le premier constructeur et on affichera des champs vides dans la Form. Si un contact est passé en paramètre du constructeur à l'instanciation, on passe dans la surcharge du constructeur. Dans ce constructeur on initialise tous les champs du contact à éditer et on passe le booléen modification

Extrait de code de la classe Categorie

```
/// <summary>
    /// la classe catégorie
    /// </summary>
    public class Categorie
    {//*à montrer
        /// <summary>
        /// idMax unique à la classe
        /// </summary>
        static int idMax = 0;
        int id;
        string intitule;

        /// <summary>
        /// Les contacts appartenant à la catégorie (construit à partir de
catégoriser)
        /// </summary>
        List<Contact> listContact = new List<Contact>();

        /// <summary>
        /// Constructeur de la classe initialisant des champs privés
        /// </summary>
        /// <param name="ID">ID unique de l'objet</param>
        /// <param name="intitule">intitulé de la catégorie</param>
        public Categorie(int ID, string intitule)
        {
            this.id = ID;
            this.intitule = intitule;


            if (ID > Categorie.idMax)
            {
                Categorie.idMax = ID;
            }
        }
```

Légende : extrait de code

Cette classe est instanciée dans 2 cas, si on récupère la catégorie depuis la base de données ou depuis l'interface AjouterEditCategorie . Le champ static idMax permet connaitre l'id maximum utilisé en BDD afin d'insérer dans celle-ci une nouvelle catégorie(le même mécanisme est présent dans la classe Contact). La liste Contact se trouve dans la classe Categorie. Ainsi une catégorie sait à quels contacts elle est affiliée, ce qui est initialisé lors du chargement des informations depuis la table catégoriser (le même mécanisme est présent dans la classe contact). Ces listes facilitent l'affichage et la gestion des données.

7) Conclusion :

Le développement de cette application à permis la mise en pratique de mes connaissances théoriques, acquises au long de la formation, à la réalité pratique.
Sont survenus de nombreuses difficultés lors de la conception de l'application et de la programmation.
Quelques erreurs de ma part, lors de la mise en place du MCD ainsi que durant la programmation, ont ralenti le reste du développement.
En remettant en question certaines de mes compréhensions qui n'étaient pas adaptées, j'ai repensé le MCD avant de m'être trop avancé, puis confirmé dans ma pensée par des recherches ultérieures, j'ai pu terminer cette application.
J'ai aussi effectué quelques modifications dans l'ergonomie de l'application, en particulier pour ce qui est de l'affichage de l'adresse et de la photo, afin de ne pas surcharger inutilement l'interface.
Cette expérience m'a été d'une grande aide dans la consolidation de mes connaissances et me sera utile pour améliorer la conceptualisation de mes applications futures ainsi que leur programmation.