MANZANO Lilian *Groupe T*

PERE Brandon

**Rapport final - CDAA**

SUJET

Ce rapport porte sur une application créé en C++ en utilisant QT. Cette application est un gestionnaire de contacts (nom, prénom, téléphone, …), avec des fonctionnalités de recherches et d’ajout de notes.

**TABLE DES MATIÈRES**

[Fonctionnalités 3](#_Toc121930777)

[Fenêtre principale : 3](#_Toc121930778)

[Interface : 3](#_Toc121930779)

[Recherches : 3](#_Toc121930780)

[Barre d’outils : 4](#_Toc121930781)

[Fenêtre de détails de contact : 4](#_Toc121930782)

[Infos du contact : 4](#_Toc121930783)

[Interactions du contact : 4](#_Toc121930784)

[Fenêtre d’édition des tags : 5](#_Toc121930785)

[Fenêtre d’édition de contact : 5](#_Toc121930786)

[Fenêtre d’historique des modifications : 6](#_Toc121930787)

[Fenêtre de choix des dates : 6](#_Toc121930788)

[Indications à l’utilisateur : 6](#_Toc121930789)

[Fichier et Dossiers 7](#_Toc121930790)

[Architecture de la base de données 8](#_Toc121930791)

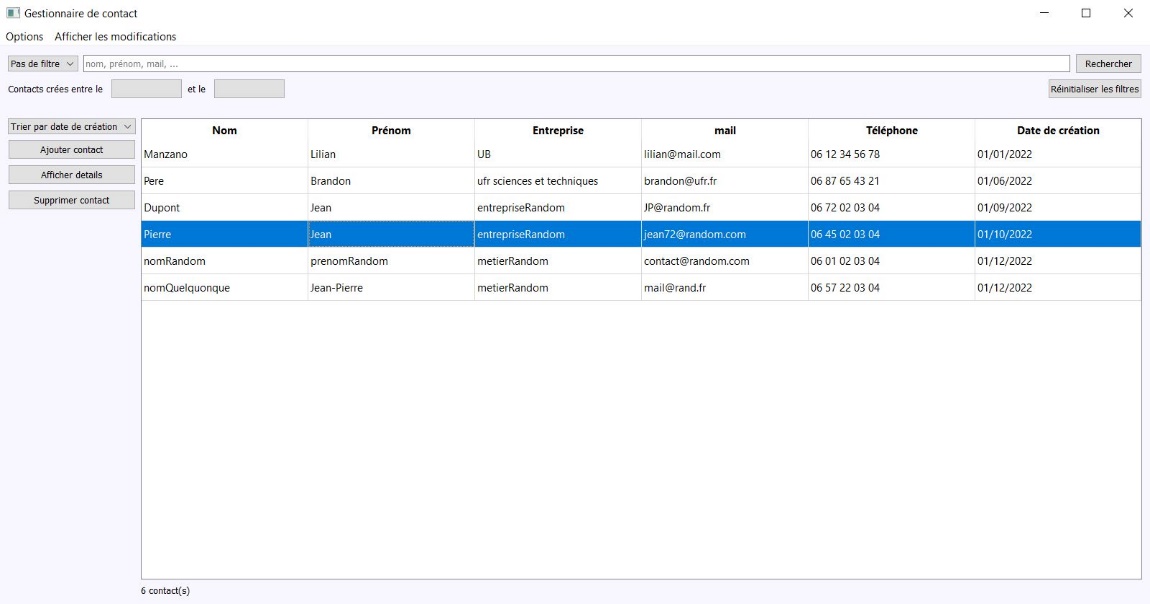
[Diagramme de la base de données : 9](#_Toc121930792)

[Description de la base de données : 9](#_Toc121930793)

# Fonctionnalités

## Fenêtre principale :

### Interface :



La fenêtre principale affiche la liste de contacts et leurs caractéristiques dans un tableau, celui-ci permet de sélectionner une ligne correspondant à un contact.

On trouve en haut une barre de recherche et différents filtres pour trouver des contacts en fonctions de leurs propriétés (nom, prénom, entreprises, …), de leur date de création, ou encore des tags @todo et @date qu’ils possèdent.

En bas du tableau, on y voit indiqué le nombre de contacts correspondant à la recherche, si aucune recherche n’a été faite c’est donc le nombre total de contacts qui y est indiqué.

Enfin à gauche du tableau on trouve un sélecteur et 3 boutons. Le sélecteur permet de choisir un tri des contacts dans le tableau, soit par date de création, par ordre alphabétique ou alphabétique inverse, ces deux derniers tris prennent en compte le nom du contact. De plus il y a un bouton qui permet de supprimer le contact sélectionné, un bouton qui permet d’ouvrir la fenêtre d’édition afin d’ajouter un nouveau contact, et pour finir le troisième bouton permet d’ouvrir la fenêtre de détails du contact sélectionné.

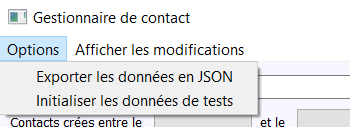
### Recherches :

La barre de recherche et les filtres peuvent tous être cumulés, cela permet ainsi de nombreuses possibilités :

* Il y a un sélecteur de filtres parmi lequel on peut choisir de ne pas en appliquer ou filtrer par nom, prénom, entreprise, mail, ou téléphone.
* Il y a aussi 2 boutons permettant de choisir une date de départ et de limite pour les dates de création des contacts, ainsi si les 2 dates sont sélectionnées on peut trouver des contacts crées sur une période précise.
* Dans la barre de recherche en plus des informations du contact, on peut renseigner des tags @todo ou @date, tout ce qui se trouvera après un tag @todo jusqu’au prochain tag sera considéré comme contenu du tag.

Par exemple si nous voulons tous les contacts crées après le 01/12/2022 ayant une adresse mail «.com» et possédant un tag @todo renseignant une réunion le 01/01/2023, nous pouvons faire :

### Barre d’outils :



Sur cette fenêtre principale on y trouve aussi une barre d’outils. Cette dernière possède un menu options qui permet d’initialiser des données de tests ou d’exporter les données de la base de données en format JSON (l’historique des modifications n’est pas exporté), le fichier «.json» apparait dans le dossier «data» crée par notre application près de l’exécutable. En plus de ce menu, il y a un autre bouton dans la barre d’outils qui permet d’ouvrir la fenêtre d’historique de toutes les modifications.

## Fenêtre de détails de contact :

### Infos du contact :



Cette fenêtre permet de montrer les détails du contact sélectionné dans le tableau de la page principale, on y voit toutes les informations du contact, ainsi que sa photo (l’image par défaut ci-dessus). On peut également y trouver un bouton qui ouvre une fenêtre pour modifier le contact, et un bouton qui ouvre une fenêtre de l’historique des modifications uniquement de ce contact.

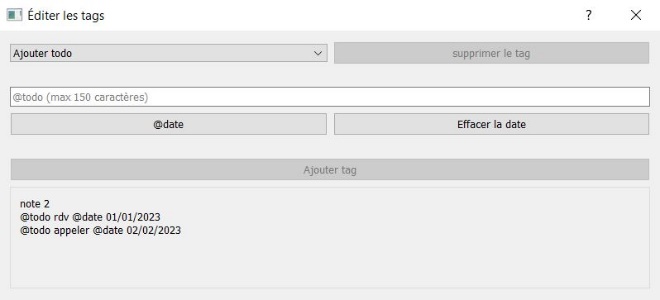
### Interactions du contact :

En seconde partie de la fenêtre on peut éditer les interactions de ce contact, il y a un sélecteur, s’il est sur le premier choix, l’on peut ajouter une interaction. Les autres choix de ce sélecteur sont en fait les interactions que possède le contact, en en sélectionnant une on peut choisir de la modifier, de la supprimer ou d’éditer ses tags (cela ouvrira une nouvelle fenêtre) . De plus on voit en dessous le contenu de l’interaction et ses tags.

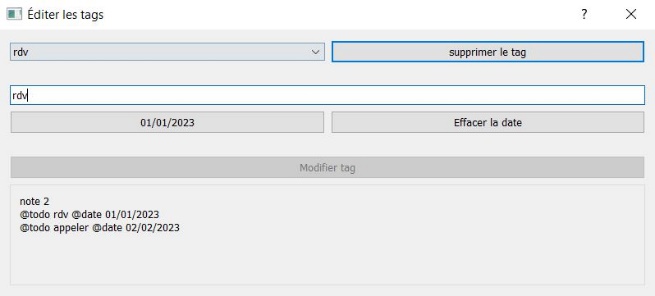
Une image contenant texte

Description générée automatiquement

## Fenêtre d’édition des tags :

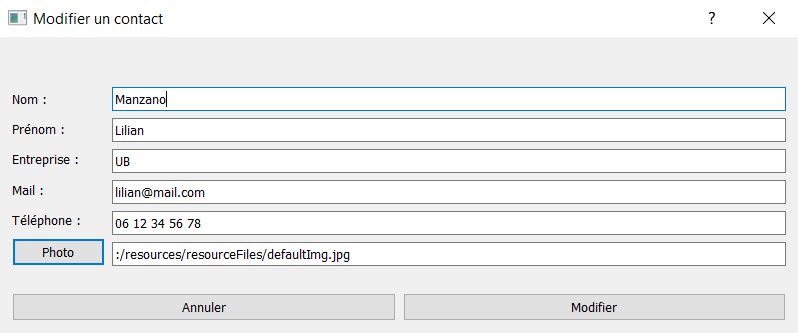


L’édition des tags est similaire à l’édition des interactions, il y a un sélecteur dont le choix par défaut est d’ajouter un nouveau tag, les autres choix sont les tags déjà existants que l’on peut modifier. Il faut au minimum renseigner un contenu qui sera la tag @todo, et il est possible de renseigner une date qui sera donc le @date. On voit en bas l’interaction concernée et tous les tags.



## Fenêtre d’édition de contact :

Comme nous avons vu précédemment, il y a 2 cas où la fenêtre d’édition de contact s’affiche, lorsque nous voulons crée un nouveau contact depuis la fenêtre principale, ou quand nous voulons en modifier un depuis la fenêtre de détails d’un contact. Pour ces deux utilisations, c’est la même fenêtre, celle-ci est juste prérempli avec les informations du contact que nous voulons le modifier.

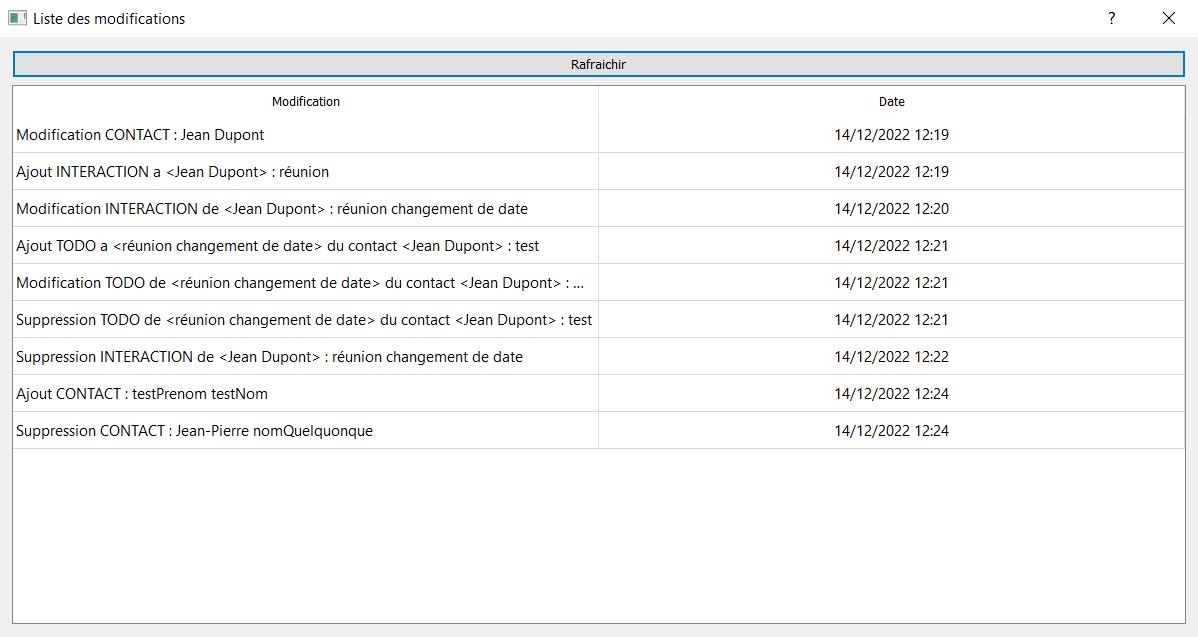


Pour éditer un contact il faut remplir obligatoirement les champs nom, prénom, entreprise, mail et téléphone. La photo est facultative, si aucun lien n’est renseigné ou si celui-ci est erroné, l’image par défaut sera associée au contact, en revanche quand le lien vers une image est bon (image existante et sous le format JPG, JPEG, ou PNG), l’image est copié dans le dossier «img» crée par notre programme près de l’exécutable, ainsi l’utilisateur peut modifier l’emplacement ou supprimer l’image d’origine sans impacter l’application.

## Fenêtre d’historique des modifications :

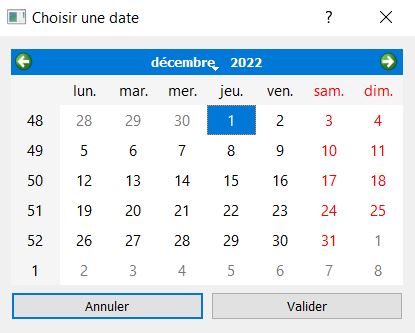
Comme vu précédemment la fenêtre d’historique des modifications s’ouvre dans 2 cas, on peut voir l’historique d’un contact en particulier depuis la fenêtre de détails de celui-ci, c’est-à-dire sa création, ses modifications, mais aussi tout ce qui concerne ses interactions et tags.

Et depuis la page principale on peut choisir de voir l’historique global, c’est-à-dire toutes les informations vues juste avant pour tous les contacts, mais aussi les suppressions des contacts qui n’existent plus.



Si des modifications sont faites pendant que cette page est ouverte, le tableau n’est pas actualisé automatiquement pour les afficher, il y a en haut de la fenêtre un bouton pour rafraichir et les afficher sans avoir besoin de fermer et rouvrir la fenêtre.

## Fenêtre de choix des dates :

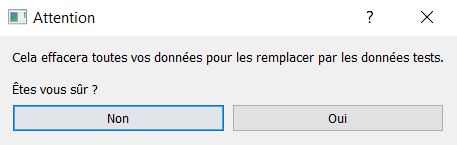
Pour choisir des dates que ce soit pour les filtres de recherches où les tags @date, ce sont des boutons qui ouvre une fenêtre de dialogue avec un calendrier où il est possible de sélectionner une date, cela nous permet notamment de ne pas avoir à vérifier le format de la date contrairement à un champ d’édition

## Indications à l’utilisateur :

Pour faciliter l’utilisation de notre application, tout au long de l’utilisation de celle-ci nous informons l’utilisateur de nombreuses informations grâce à des message rouges pour une erreur, un message vert pour les réussites, et une boîte de dialogue posant une question avec oui ou non comme réponse. Par exemple :







# Fichiers et Dossiers

En plus du fichier «.pro», et des sources «.h» et «.cpp», notre projet possède un dossier «resourceFiles» et un fichier «resource.qrc» renseignant tout ce que contient ce dossier. Ce dossier possède :

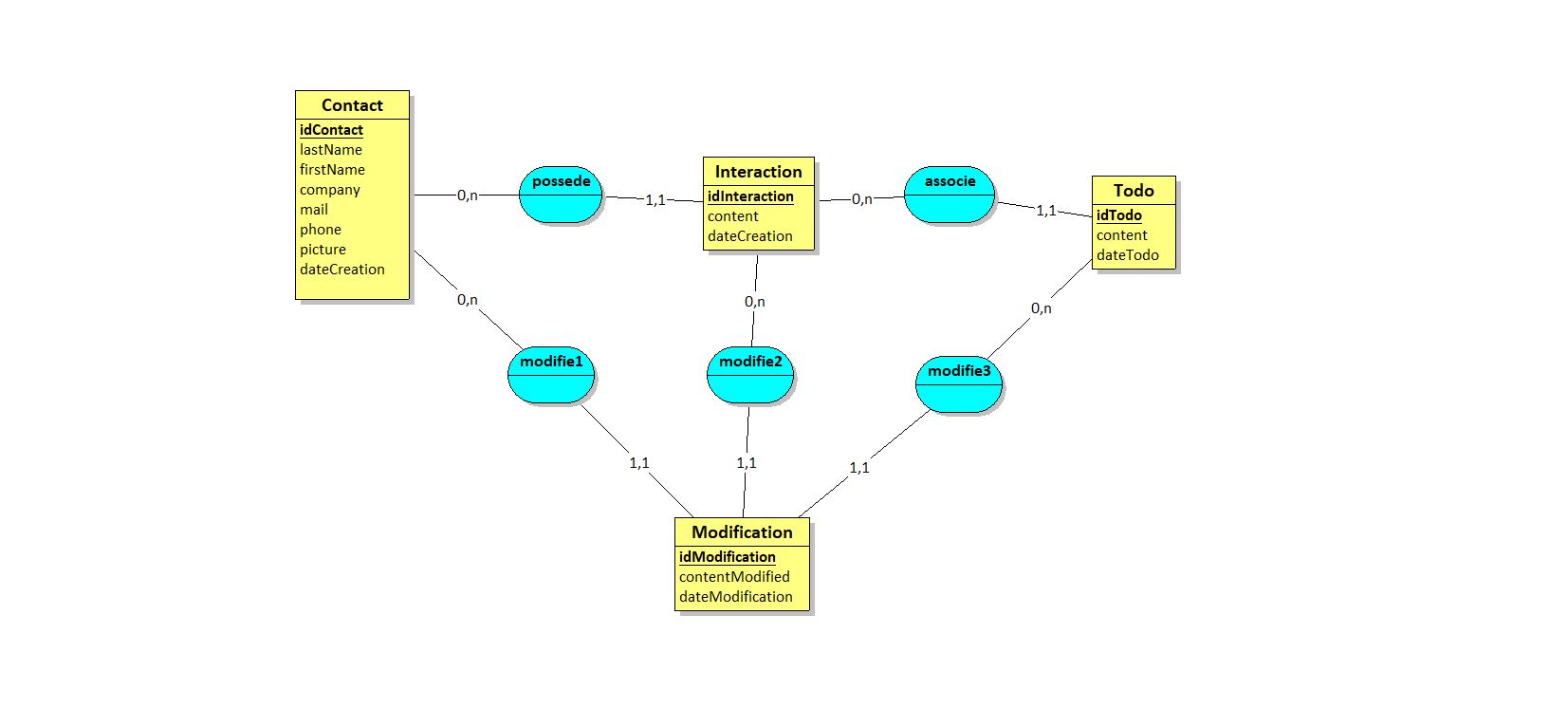
* L’image par défaut d’un contact.
* Un fichier de style «style.qss» que nous n’avons pas encore vraiment rempli, mais que nous avons mis pour éventuellement perfectionner le projet par la suite.
* Le fichier «dataTest.sql» qui contient des requêtes d’insertions pour initialiser les données de tests.
* Les 4 fichiers qui contiennent les instructions pour créer les 4 tables de notre base de données : «initContact.sql», «initInteraction.sql», «initTodo», «initModification».

En plus de cela notre programme va quand il va être lancé créer 2 dossiers près de l’exécutable :

* «data» qui va contenir la base de données SQLite et les éventuelles fichier JSON lors d’un export,
* «img» qui va contenir les images copiée lorsque l’utilisateur choisi sur son ordinateur une photo pour un contact.

# Architecture de la base de données

## Diagramme de la base de données :



## Description de la base de données :

Notre base de données n’a pas évolué depuis le premier jalon, elle possède toujours 4 tables qui sont respectivement :

* **Contact** (idContact, lastName, firstName, company, mail, phone, picture, dateCreation)
* **Interaction** (idInteraction, content, dateCreation, *idContact*)
* **Todo** (idTodo, content, dateTodo, *idInteraction*)
* **Modification** (idModification, contentModified, dateModification, *idContact*, *idInteraction*, *idTodo*)

# DIAGRAMME UML

# DIAGRAMME DES CLASSES