UFR sciences & techniques

UNIVERSITÉ DE BOURGOGNE

CDAA-R

LACHAUD Samuel / PAZOLA Loïs – Info S5 TP4

SOMMAIRE

I) Introduction

Qu'est-ce que CDAA-R

Le projet CDAA-R est un projet développé en **QT** (partie graphique) et en **C++** (partie métier). Celui-ci consiste en la création d'une application de gestion de contacts avec toutes les fonctions qui lui sont liées. Les notions alors abordées sont :

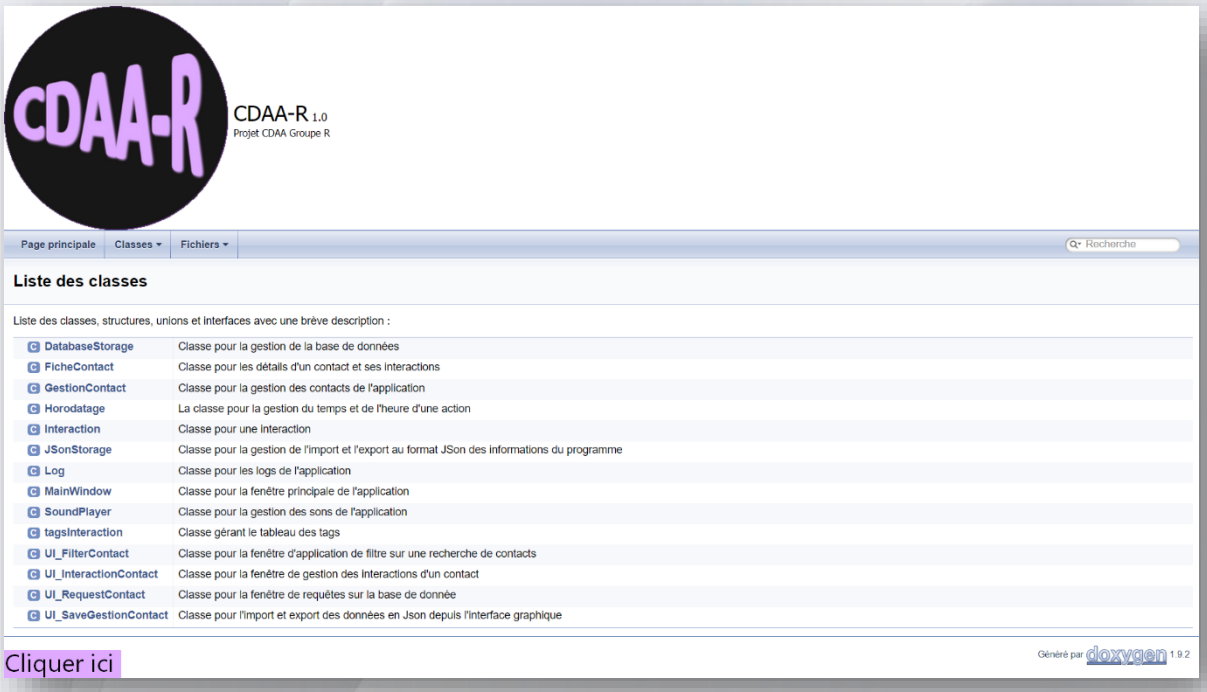
- La conception graphique
- La programmation Objet
- La gestion de base de données **SqLite**
- La gestion de fichiers **JSON**
- Les différents contrôleurs de QT

Nous allons donc présenter dans ce rapport notre application finie, des coulisses de celle-ci, jusqu'à son installation.

Documentation Doxygen

Notre projet étant en source libre sur **GitHub**, la documentation **Doxygen** est un outil indispensable pour permettre à toute personnes s'intéressant au projet de savoir rapidement les composants et le fonctionnement de celui-ci. On peut alors retrouver cette documentation **ci-dessous** ou à l'adresse suivante :

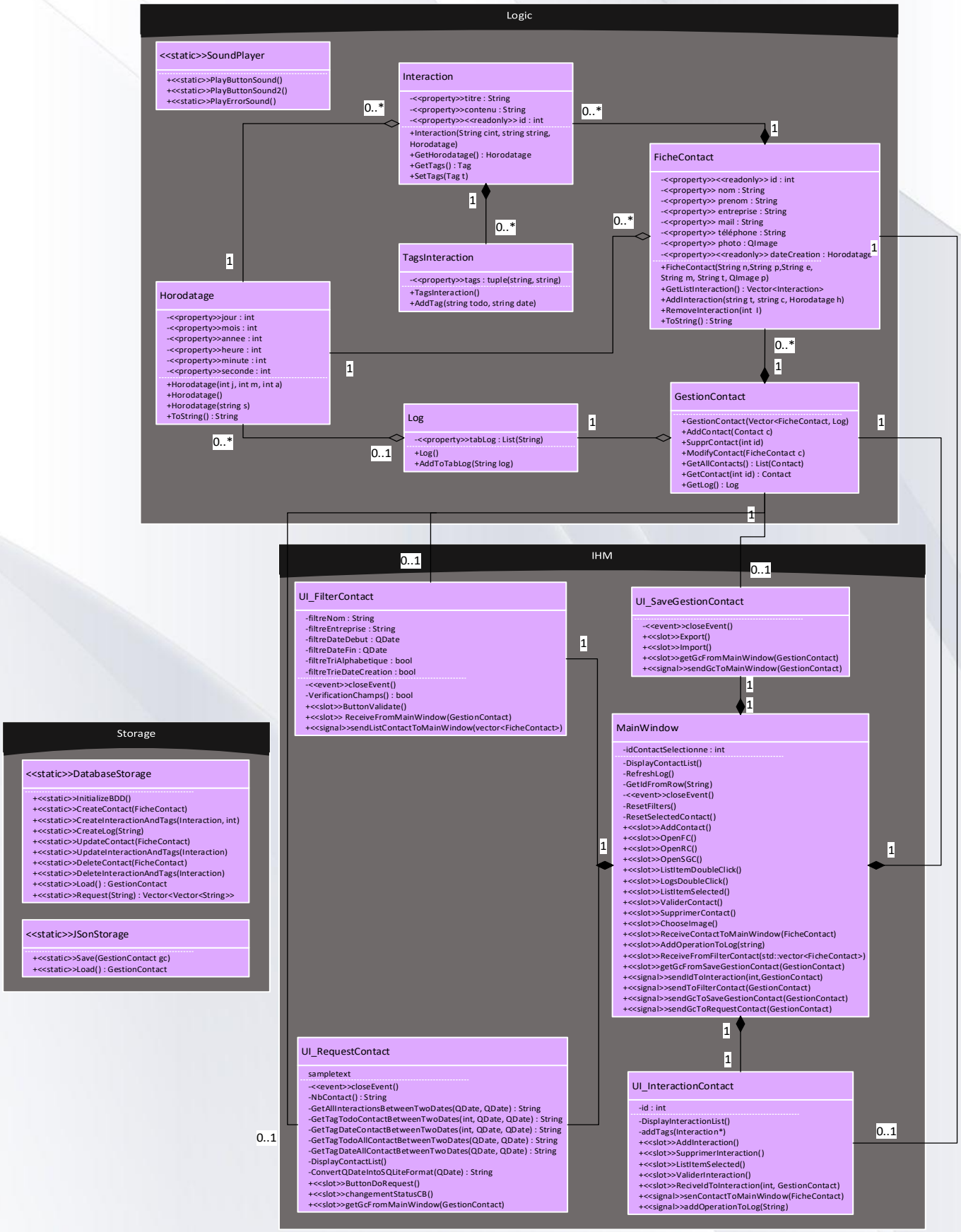
➔ https://github.com/samlach2222/ProjetCDAA-R/tree/main/Doxygen/html



Cliquer ici

II) Conception

Diagramme de Classe



Nous trouvons sur la feuille précédente le diagramme UML de Classes présentant notre projet. Il est composé de la partie **Métier/Logic** qui forme l'ensemble des classes gérant les structures et les choses internes, la partie **Interface/IHM** gérant toutes les classes visuelles, avec des fenêtres montrées au joueur, et enfin la partie **Stockage/Storage** gérant les entrées et sorties du projet que ce sois par fichier ou par base de données. Les classes qui composent ces trois packages et leurs liaisons seront décrites dans la partie suivantes.

Explication des liaisons interclasses

1. Package Métier/Logic

Dans ce premier package, la classe principale est la classe **GestionContact**, Elle est donc composée de plusieurs contacts, elle gère donc ceux-ci. Pour toutes modifications, on a une agrégation de la classe **Log** sur **GestionContact**, afin de notifier tout changement. A un contact on peut ajouter des interactions, **FicheContact** est donc composée d'**Interactions**. Une interaction est elle-même composée d'un ou plusieurs **tagsInteraction** (tags date et todo). Afin de toujours savoir quand un ajout a été fait, les classes **FicheContact**, **Interaction** et **Log** sont horodatées avec la classe **horodatage** dans des liaisons d'agrégation

Avec ces compositions nous avons donc des liaisons qui sont grammaticalement logiques, en effet, s'il n'y a plus de gestions de contacts, les contacts n'existent plus, s'il n'y a plus de contact, il n'y a plus d'interactions liées et de même pour les tags. Nous avons donc ainsi un respect des principes SOLID

Enfin, la dernière classe est SoundPlayer, c'est une classe statique, elle est appelée sans être instanciée, elle n'a donc pas de liaisons UML.

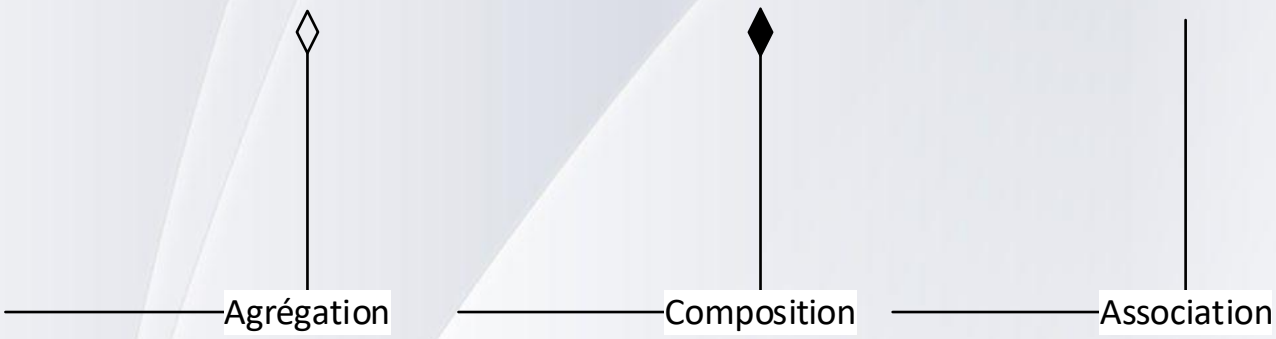
2. Package Interface/IHM :

Au sein de ce package, les liaisons sont plus faciles. Toutes les classes UI_nomDeLaClasse sont des composants de **MainWindow**. En effet si celle-ci est détruite, il n'y a plus de programme, donc toutes les autres fenêtres formées par les classes UI sont fermées, il y a donc compositions.

Si les liaisons intra-package sont faciles, les liaisons extra-packages sont plus complexes. **MainWindow** est composée de **GestionContact** du package précédent. Par la même occasion, **UI_SaveGestionContact** récupère **GestionContact** pour l'import et l'export. Enfin **UI_InteractionContact** récupère **FicheContact** pour récupérer directement les interactions de ce contact.

3. Package Stockage/Storage :

Pour finir ce package est composé uniquement de classes statiques, il ne comprend donc aucunes liaisons UML.

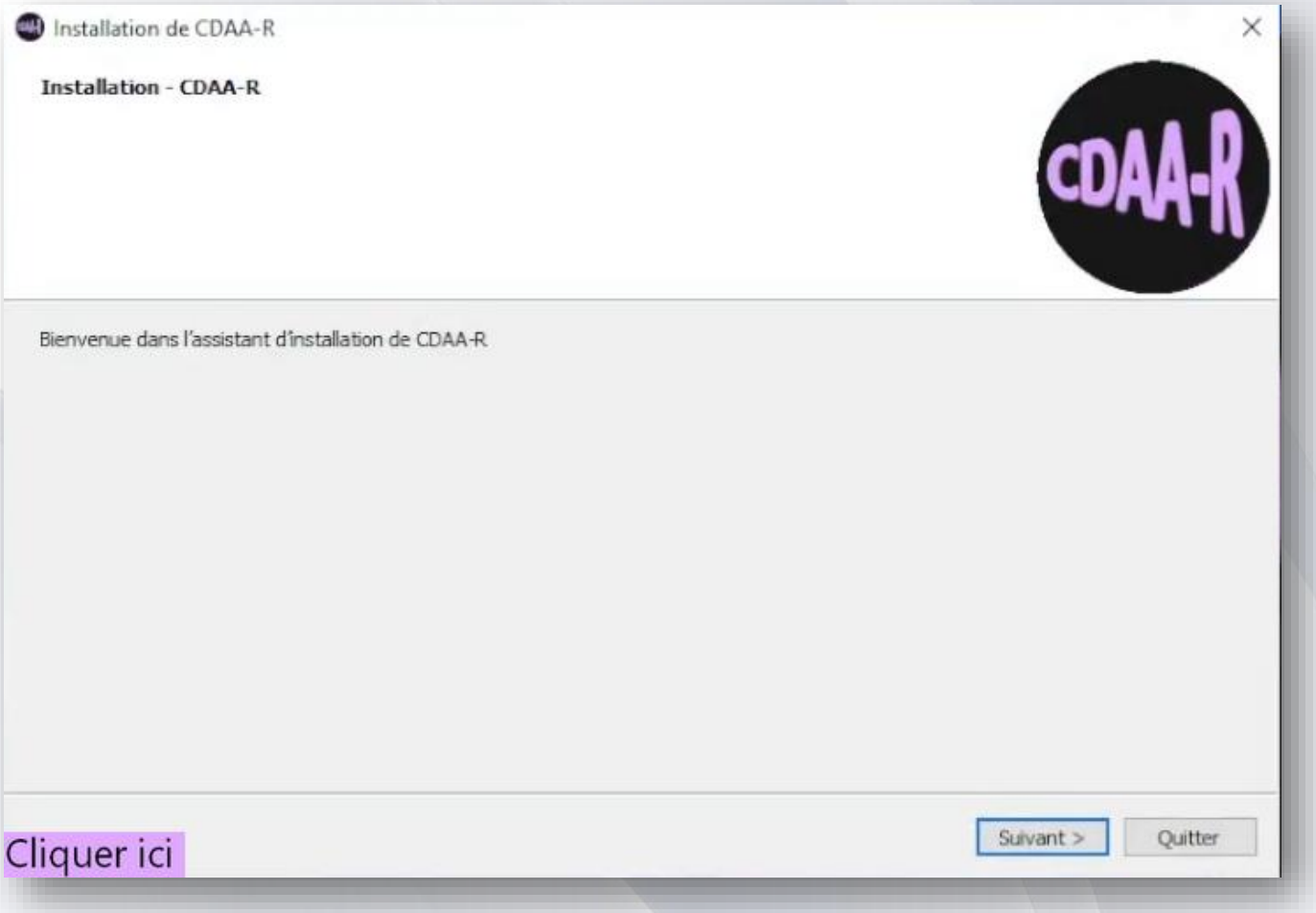


III) Installation

Afin de permettre à toute personne d'utiliser facilement notre application, et ce sur une multitude de supports, nous mettons à disposition un installateur pour trois types de logiciels différents.

Installation EXE + UWP

Notre installateur se présente comme un fichier exe permettant d'installer notre application. Par le double clic dessus-la fenêtre s'ouvre et l'application se retrouve installée en suivant la procédure suivante :

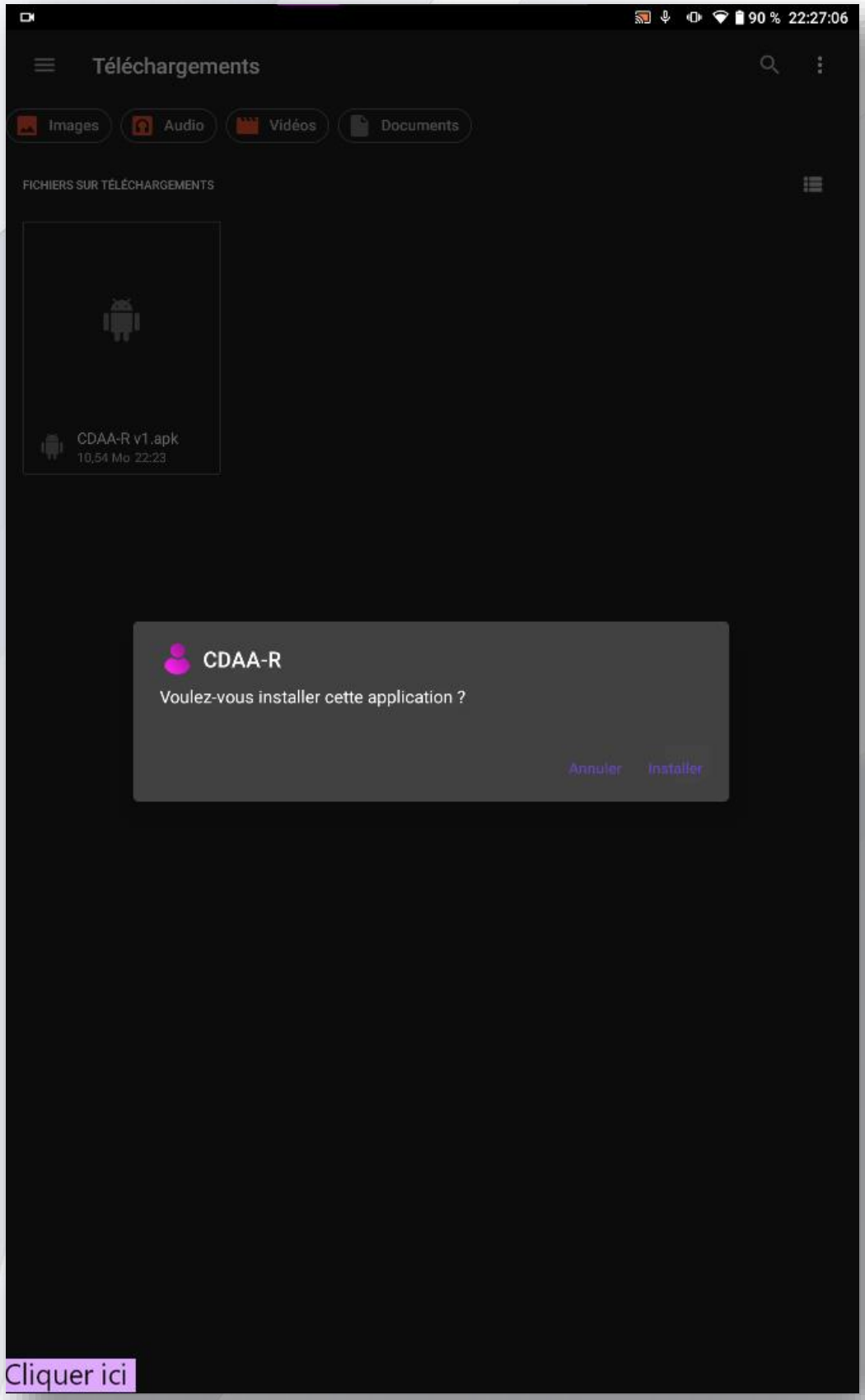


A la fin de l'installateur, le répertoire d'installation s'ouvre, on y retrouve l'exécutable de l'application Windows. Si on regarde dans le menu démarrer, on trouvera également l'application UWP (Application Windows store).

Enfin dans le dossier d'installation on trouve également le fichier APK (Android Package Kit) qui va nous servir pour la partie suivante

Installation APK

Une fois que nous avons notre fichier APK, nous le récupérons depuis notre appareil Android (tablette, smartphone). En cliquant sur celui-ci, une demande d'autorisation vous sera faite, il suffit d'accepter pour lancer l'installation de celui-ci comme ci-contre :

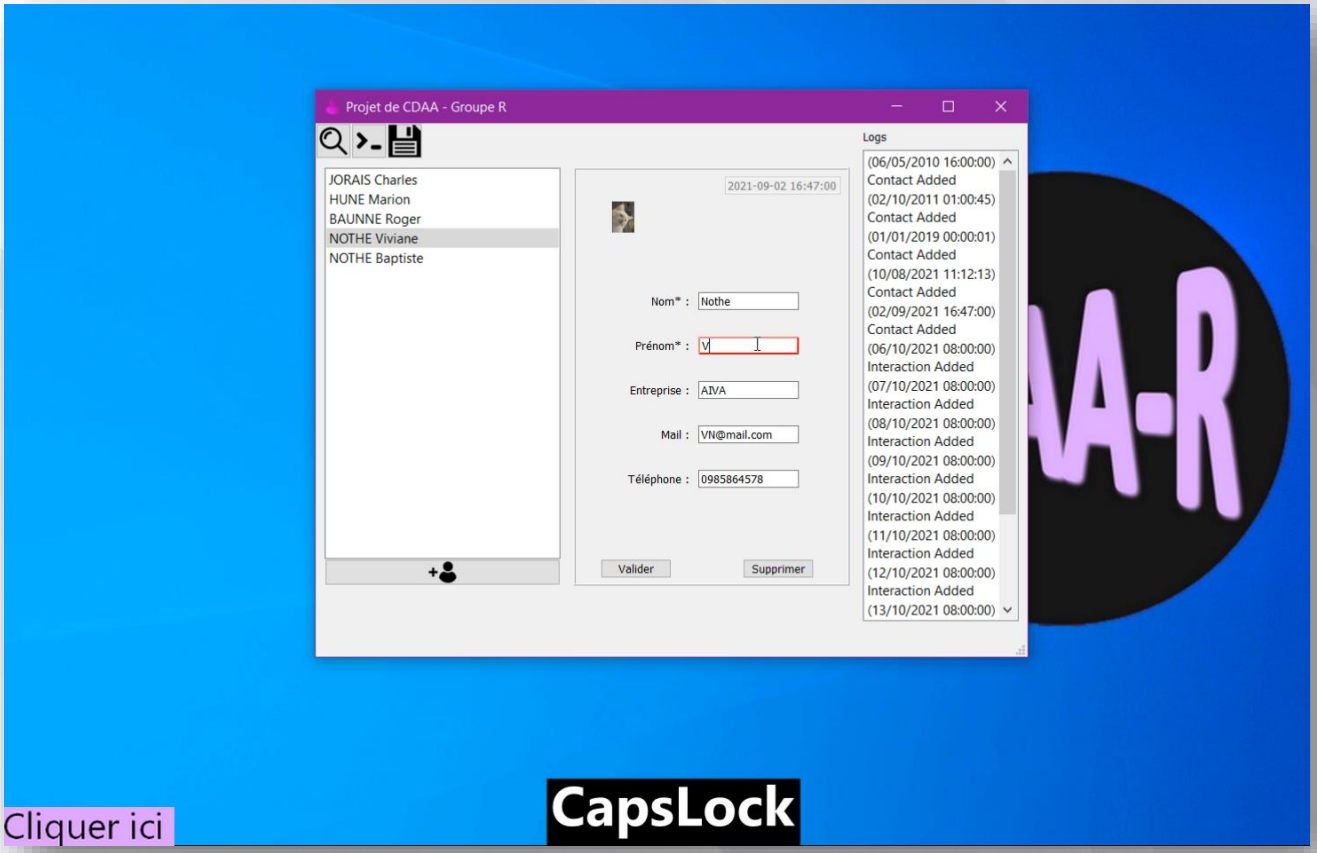


IV) Utilisation

Maintenant que l'application est installée sur notre support préféré, nous allons voir deux points qui se distingue de l'utilisation normale que l'on aurait de l'application.

Touches clavier pour naviguer

Beaucoup d'utilisateurs ayant un usage avancé d'un ordinateur utilisent les raccourcis clavier pour naviguer sur un logiciel. La navigation à l'aide du clavier est donc totalement possible sur notre application, comme montré ci-dessous :



Feedbacks sonores

Dans un jeu vidéo comme dans une application, aussi petite soit-elle, la patte musicale est super importante, on reconnaitra facilement une interface avec des bons retours (ou feedbacks) sonores (comme les menus de Nintendo avec des bruitages très reconnaissables et qui donnent plaisir à juste naviguer dans cette interface).

A notre petite échelle, nous avons donc musicalement travaillé sur les sons des boutons afin d'obtenir une sonorité agréable à l'oreille, qui donne envie de cliquer sur les différents boutons de notre application. Après analyse des sons de plusieurs applications et systèmes d'exploitation, nous avons décidé de procéder à un enregistrement personnel, avec une série de traitement pour façonner ce bruitage à notre image.

Le deuxième bruitage est le message d'erreur, celui-ci se doit d'être impactant et de ne pas frustrer l'utilisateur, nous l'avons donc créé en conséquence. Vous pouvez alors les entendre ci-dessous :