Table des matières

[A. Introduction : 2](#_Toc326103225)

[1. Le Cahier des Charges : 2](#_Toc326103226)

[2. Matériel et logiciels utilisés : 2](#_Toc326103227)

[B. Remerciements : 2](#_Toc326103228)

[C. Fonctionnalités : 3](#_Toc326103229)

[1. Fonctionnalités générales : 3](#_Toc326103230)

[2. Modules nécessaires : 3](#_Toc326103231)

[D. Solution : 3](#_Toc326103232)

[1. La fenêtre principale : 3](#_Toc326103233)

[2. Recherche : 4](#_Toc326103234)

[3. La liste de contacts : 4](#_Toc326103235)

[4. Les informations : 5](#_Toc326103236)

[E. Conclusion : 5](#_Toc326103237)

# Book Blue.pngIntroduction :

Pour ce projet, nous devions réaliser une solution client/serveur en langage de programmation Java s’appuyant sur une interface graphique Swing. Le thème de l’application était la gestion de contacts.

## Le Cahier des Charges :

Nous avions pour consigne de créer une application de gestion de contact en respectant le cahier des charges suivant :

* Les contacts (étudiants BTS) sont stockés dans une base de données MySQL ;
* Chaque contact possède sur sa fiche : *Nom*, *Prénom*, *Email*, *Adresse*, *Code* *postal*, *Ville*, *Entreprise*, *Loisirs* ;
* Chaque contact possède une photo stockée sur le serveur et qui sera visible sur l'interface de contact lorsque le contact est sélectionné.

## Matériel et logiciels utilisés :

La consigne nous conseillait d’utiliser un nouvel environnement de développement que nous n’avions encore jamais utilisé pour travailler le langage Java : Netbeans IDE. Car cet IDE est plus adaptés pour travailler une IHM avec la bibliothèque Swing que l’ancien IDE que nous utilisions : Eclipse.

Pour les aspects techniques, nous nous sommes appuyés sur :

* la **Javadoc**:<http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html> ;
* le livre « ***Apprenez à programmer en Java*** » de Cyrille HERBY ;
* le site internet « ***Developpez.com*** » : <http://www.developpez.com/> ;



# Remerciements :

Nous tenons tout d’abord à remercier Monsieur AUBISSE, notre directeur. Nous remercions également Monsieur MBIANDOU, professeur de Java dans notre promotion, pour sa patience et l’aide qu’il nous a apporté. Monsieur PETIT pour sa formation sur l’objet. Nous remercions également nos responsables d’alternance Monsieur COURBIS et Monsieur BERNAUDAT.

# Magnifier.pngFonctionnalités :

## Fonctionnalités générales :

Pour être **efficace** et **utile** à l’utilisateur, une application de gestion de contacts doit contenir certains modules qui devront répondre au mieux aux besoins de l’utilisateur. En effet, si la solution ne permet de trouver facilement les contacts ou si les fiches d’informations sur les contacts sont incomplètes, l’application ne répondra pas aux besoins de l’utilisateur.

## Modules nécessaires :

Voici ci-dessous les principaux modules nécessaires à une solution de gestion de contacts :

* Une liste qui affiche l’ensemble des contacts de l’utilisateur ;
* Un champ de recherche qui permettra à l’utilisateur de pouvoir trouver avec une facilité étonnante le contact qu’il recherche ;
* Un formulaire avec les données du contact, auparavant sélectionné dans la liste, qui affichera toutes les informations du contact ;
* Une fonction d’ajout de contact, qui permet à l’utilisateur d’ajouter un contact dans la base de données ;
* Une fonction de modification, qui permet à l’utilisateur de mettre à jour les informations de ses contacts sur la base de données.



# Solution :

Nous allons maintenant vous présenter les différentes solutions que nous avons entreprises de réaliser afin de répondre aux fonctionnalités recherchées.

## La fenêtre principale :

Nous avons choisis une fenêtre attractive et intuitive. Afin que cette application soit accessible et utilisable par tous (la figure 1 présente la fenêtre générale). Comme vous pouvez le voir, cette fenêtre est très simple mais cependant complète en fonctionnalité.

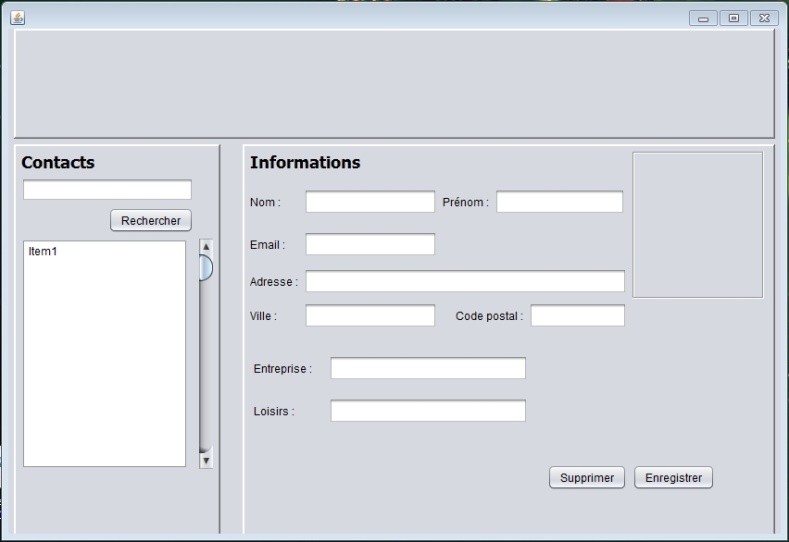


Figure 1

## Recherche :

Afin de pouvoir trouver au plus vite le contact désiré, nous avons intégré un champ de recherche dans la partie supérieure gauche, l’utilisateur entre le nom du contact désiré appuie sur « *Rechercher* » et le nom du contact désiré apparaît dans la liste (la figure 2 présente le champ de recherche).

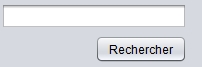


Figure 2

## La liste de contacts :

Cette liste contient l’ensemble des contacts. Pour observer, modifier ou supprimer un contact, il suffit que l’utilisateur clique sur un contact présent dans la liste et les informations de ce contact apparaissent dans la fenêtre de droite (la figure 3 présente la liste de contact).

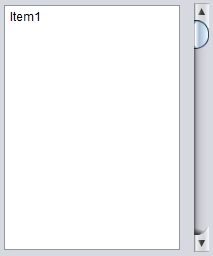


Figure 3

## Les informations :

Enfin, une fenêtre présente les différentes informations du contact sélectionné par l’utilisateur. Elle permet également de faire d’autres fonctions comme la mise à jour ou la suppression d’un contact (la figure 4 présente les informations du contact).

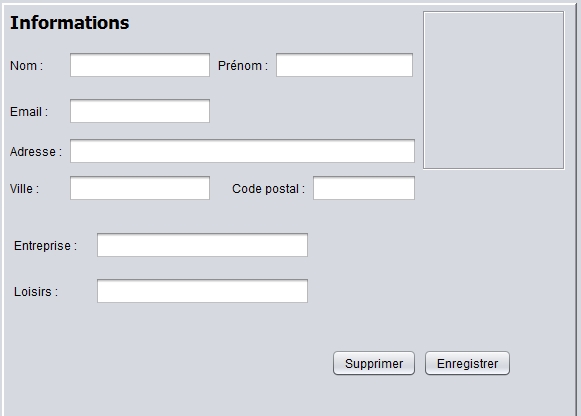


Figure 4



# Conclusion :

En conclusion, nous avons pu ajouter une liste comprenant l’ensemble des contacts, un champ de recherche, une fiche-contact regroupant l’ensemble des informations du contact, des modules d’ajout, suppression et de modification de contact.

Nous avons rencontré des problèmes avec la manipulation des images. En effet, nous n’avons pas réussi à ajouter des photos propres à nos contacts et les ajouter à la base de données. Cela est aussi dut au manque de temps, car lorsque nous avons fini notre projet nous n’avons pas eu le temps d’achever nos recherches au sujet des images.

De plus, le champ de recherche nous a posé quelques soucis au niveau de la syntaxe SQL et nous as donc fait perdre du temps car nous avons eu dut mal à comprendre notre erreur qui concerné l’ajout d’une variable Java au sein de notre requête.

Pour conclure, ce projet nous a beaucoup appris sur le langage Java. En effet, nous avons dut faire beaucoup de recherche sur la Javadoc, sur différents forums et dans des livres afin de manipuler le code et de bien le comprendre.