

Projet de développement d'application Desktop

Rudolph Attiso & Youssef Lasri
Apprentis Concepteur et Développeur d'applications en herbe

Cahier des charges : Javacard

1. Introduction
2. Analyse Fonctionnelle..
 - Création du “persona”.
 - Construction d'un Sprint pour la création du backlog avec le user stories.
3. Maquettage de l'application.
 - Création du Zoning et du Wireframe.
 - Validation de l'interface avec le product owner.
4. Conception de l'application.
 - Réflexion architecturale.
 - Conception de diagramme.
5. Début des développements.
6. Test fonctionnels au cours des développements.
7. Conclusion
 - Livraison de l'applications.



1 Introduction

- Methodes de developpement application Desktop.

L'objectif de ce projet est de développer une application “desktop” Java permettant de gérer une liste de contacts et d'exporter les informations

Choix technologiques :

- l'application doit être codée en Java (version 17 ou ultérieure) • l'interface graphique devra être construite avec JavaFX
- les vues de l'application devront être codées en FXML
- pas de contrainte sur les bibliothèques utilisables

Modalités de travail :

- date de début du projet : 26/07/2024
- présentation du projet : 07/08/2024
- équipes de 2
- gestion du projet inspiré par la méthode Scrum • utilisation d'un dépôt Git partagé

Livrables attendus :

- code source de l'application
- diagrammes UML éventuels
- “zoning” et “wireframe” de l'interface graphique

Les fonctionnalités principales attendues pendant le Sprint définie à l'aide de l'indicateur MoSCoW :

Fonctionnalité	MoSCoW
Interface graphique ergonomique proposant une gestion des contacts (CRUD sur les contacts)	M
Sauvegarde des informations de contact pour les recharger lorsque le logiciel se lance	M
Export d'un ou plusieurs contacts en JSON	M
Export d'un ou plusieurs contacts en vCard	M
Rechercher dans la liste des contact	S
Générer un QRCode contenant les informations de la vCard et l'afficher à l'écran	C
Export d'un ou plusieurs contacts en CSV	C

Dans la suite de ce document, vous trouverez des informations détaillées sur ces différentes fonctionnalités.



2

Analyse fonctionnelle

- Création du persona
- Construction d'un backlog avec le user stories
- Mise en place du Dépôt Git partagé

Création du persona

Julie Martin



Intitulé de poste

Chef de Projet

Âge

Entre 25 et 34 ans

Niveau d'études

Licence ou diplôme équivalente

Réseaux sociaux



Secteur d'activité

Technologie

Moyen de communication préféré

- Téléphone
- E-mail
- Réseaux sociaux

Outils nécessaires au quotidien

- Système de gestion de contenu
- Outil de gestion de projet
- Applications de stockage et de partage de dossiers en ligne

Responsabilités

Saisissez un texte

Indicateurs de performance

Données sur les nouveaux visiteurs, Vérification de l'identité des utilisateurs accédant à une application.

Supérieur hiérarchique

Directeur de Projet

Taille de l'entreprise

11 à 50 salariés

Objectifs

Pouvoir utiliser application desktop permettant de gérer les conats et de les exporter

Sources d'information

Saisissez un texte

Principaux défis

- Ressources
- Gestion des changements
- Communication interne
- Gestion des projets et organisation
- Développement professionnel
- Résolution des problèmes et prise de décisions

Construction d'un Sprint pour la création du Backlog et de User Stories

Lien: https://tree.taiga.io/project/rudolphattisso-contact_management/backlog

Exemple: Backlog avec création de User Stories qui sont rajoutés au Sprints

The screenshot displays the Taiga Scrum interface. At the top, a progress bar shows 0% completion with 0 defined points, 0 closed points, and 0 points per sprint. Below this, a section titled 'CUSTOMIZE YOUR BACKLOG GRAPH' provides instructions on setting up points and sprints. The main area is divided into two panels: 'Backlog' on the left and '1 SPRINTS' on the right.

Backlog (4 user stories):

USER STORY	STATUS	POINTS
#1 Ajout nouveau contact	New	?
#4 Consultation des contacts	New	?
#10 Recherche des contacts	New	?
#11 Création de QRcode	New	?

1 SPRINTS:

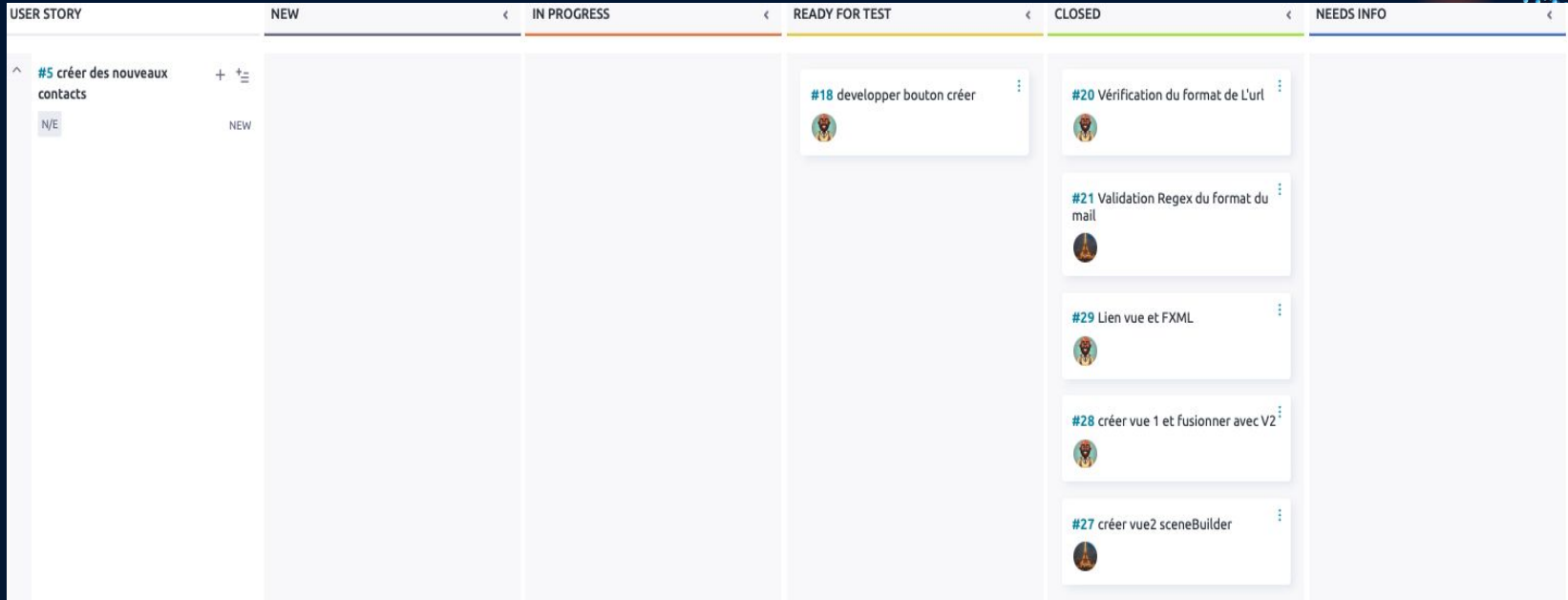
- Conception appli/architecture/DiagUML** (21 Aug 2024-04 Sep 2024)
- #5 créer des nouveaux contacts
- #34. Jeu de test-application
- #3- Persona
- #7 Suppression des contacts
- #6 Modification des informations contact
- #9 Exportation des contacts en fichier JSON
- #12- Exportation des contacts en fichiers CSV

Below the sprint list is a 'SPRINT TASKBOARD' section.

Construction d'un Sprint pour la création du Backlog et de User Stories

https://tree.taiga.io/project/rudolphattisso-contact_management/taskboard/conception-appliarchitecturediaguml

Example: Kanban



Mise en place du Dépôt Git partagé

Lien Git est privé mais vous avez l'honneur d'avoir un aperçu de la page principale.

The screenshot shows the GitHub interface for a repository named 'Project_ContactManagement'. The repository is public and has 1 watch, 0 forks, and 0 stars. The main branch is 'main', and there are 7 branches and 0 tags. The repository was created by 'rudolphattisso' 8 hours ago, with 44 commits. The file list includes 'doc', 'src/main', '.gitignore', 'contact-cv-cvc.json', 'contact-rireud-cecAti.json', 'contact-zer-tyu.json', 'contact.ser', 'contacts.json', and 'pom.xml'. The 'README' section is currently empty, with a prompt to 'Add a README'. The right sidebar contains sections for 'About' (project description), 'Releases' (no releases published), 'Packages' (no packages published), 'Contributors' (3 contributors: rudolphattisso, LasriYoussef, ludovic-esperce), and 'Languages' (Java 99.9%, CSS 0.1%).

Project_ContactManagement (Public)

Watch 1 Fork 0 Star 0

main 7 Branches 0 Tags

Go to file Add file Code

rudolphattisso debu d ela comboBox qui gardait le dernier genre selectionné après... a676271 · 8 hours ago 44 Commits

doc	Rajout du fichier Vue2.fxml	last week
src/main	debu d ela comboBox qui gardait le dernier genre selectio...	8 hours ago
.gitignore	Modification du .gitignore pour ingorer les fichiers .DS_St...	last week
contact-cv-cvc.json	amélioration de l'interface graphique	9 hours ago
contact-rireud-cecAti.json	travail sur la sérialisation Json avec la bibliothèque JSON....	2 days ago
contact-zer-tyu.json	amélioration de l'interface graphique	9 hours ago
contact.ser	debu d ela comboBox qui gardait le dernier genre selectio...	8 hours ago
contacts.json	amélioration de l'interface graphique	9 hours ago
pom.xml	Merge remote-tracking branch 'origin/ContactVCardSeria...	11 hours ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

About

L'objectif de ce projet est de développer une application "desktop" Java permettant de gérer une liste de contacts et d'exporter les informations sous différents formats.

Activity 0 stars 1 watching 0 forks Report repository

Releases

No releases published [Create a new release](#)

Packages

No packages published [Publish your first package](#)

Contributors 3

- rudolphattisso**
- LasriYoussef** Lasri Youssef
- ludovic-esperce** Ludovic Esperce

Languages

Java 99.9% CSS 0.1%



3

Maquettage de l'application

- Création du Zoning et du Wireframe

Creation de Zoning et Wireframe

II.1. Gestion de contacts

L'application devra proposer une **interface graphique ergonomique** permettant de gérer une liste de contacts.



Les opérations attendues sont les opérations **CRUD** :

- consultations d'une liste de contact(s)
- création de contact(s)
- modification de contact(s)
- suppression de contact(s)

Une contact est défini par les caractéristiques suivantes :

- **nom** (obligatoire)
- **prénom** (obligatoire)
- **genre** : homme/femme/non-binaire (obligatoire)
- **date de naissance** (optionnel)
- **un pseudonyme** (optionnel)
- **adresse** (obligatoire)
- **numéro de téléphone personnel** (obligatoire)
- **numéro de téléphone professionnel** (optionnel)
- **une adresse email** (obligatoire)
- **une adresse postale** (obligatoire)
- **un lien vers la page Github ou Gitlab du contact** (optionnel)

Creation de Zoning et Wireframe

Zoning et Wireframe adapté sur deux formats: 1024px x 768px & 1920px x 1080px

This wireframe diagram illustrates a user interface layout. It features a large rectangular area on the left labeled "Tableau" (Table), which is divided into a grid of columns. To the right of this area is another large rectangular area labeled "Formulaire" (Form). At the bottom of the layout, there are four buttons: "Boutton export" (Export button), "Boutton export" (Export button), "Boutton creer" (Create button), and "Boutton modifier" (Modify button). Below the "Boutton creer" and "Boutton modifier" buttons is a third button labeled "Boutton supprimer" (Delete button).

This detailed wireframe shows a user interface layout with a table and a form. The table has five columns: "Genre", "Nom", "Prénom", "Mail", and "Numéro personnel". To the right of the table is a form with the following fields: "Nom", "Prénom", "Genre", "Date de naissance", "Pseudonyme", "Adresse", "Numéro personnel", "Numero professionnel", "Mail", and "Lien Github/Gitlab". Each field has a corresponding "x" button. At the bottom of the form are three buttons: "Créer", "Modifier", and "Supprimer". Below the table, there are two buttons: "Export JSON" and "Export vCard".



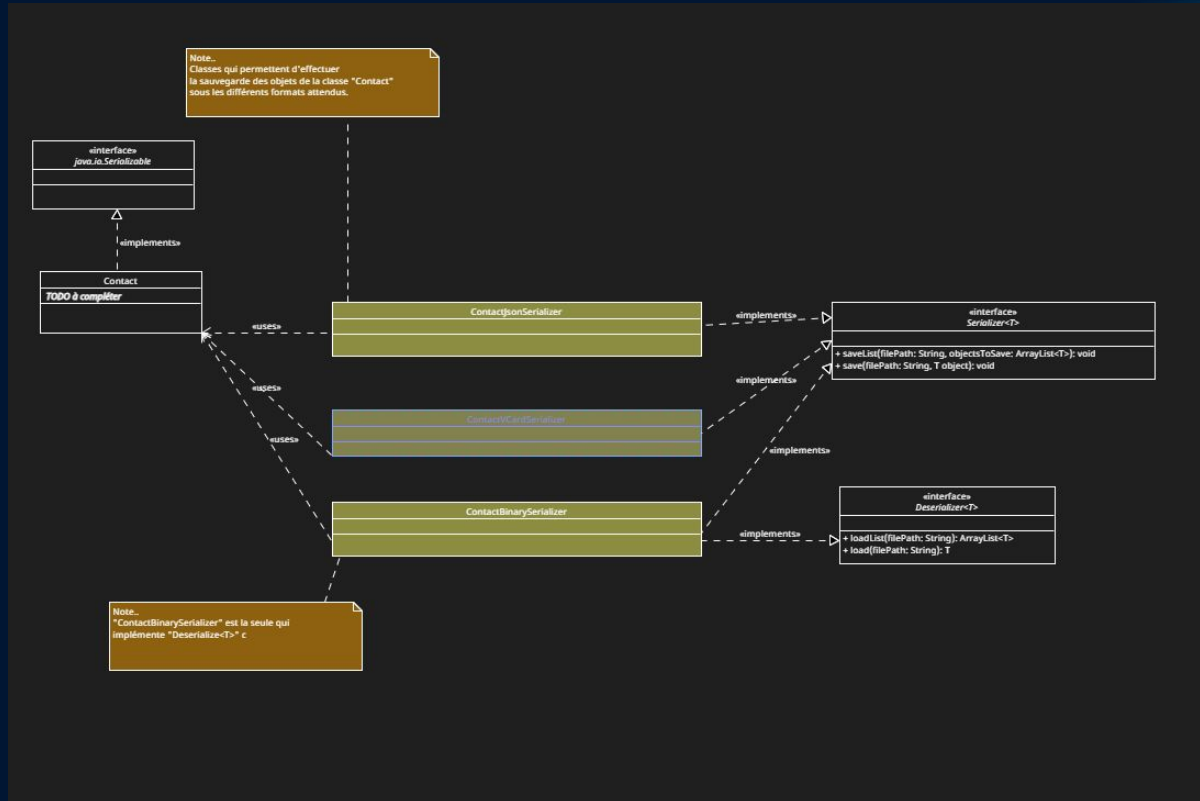
4

Conception de l'application

- Réflexion architecturale
- Conception de diagramme

Conception du diagramme

Diagramme uml initial

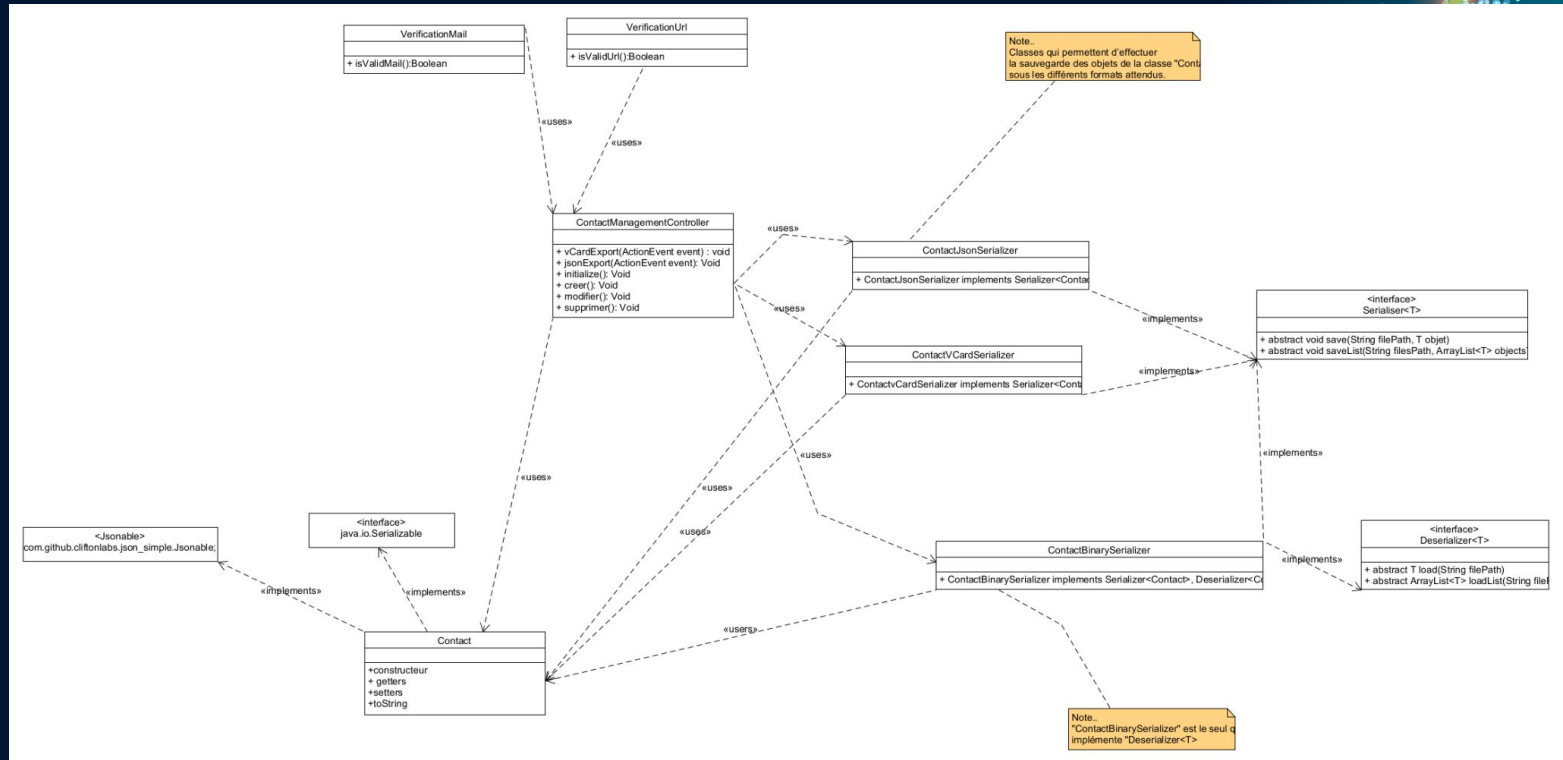




5 Début des développements

Conception du diagramme

Diagramme UML final



Conception du diagramme

Veille OWASP: https://owasp.org/www-community/OWASP_Validation_Regex_Repository

```
<?xml version="1.0"?>
```

```
<regex>
```

```
<name>url</name>
```

```
<pattern><![CDATA[^((((https?|ftp|gopher|telnet|nntp)://)|(mailto:|news:))(%[0-9A-Fa-f]{2}|[-()_.!~*'"/?:@&=+$,A-Za-z0-9])+)([!.!';/?:,][[:blank:]:blank:]?)?$]]></pattern>
```

```
<description>A valid URL per the URL spec.</description>
```

```
</regex>
```

```
<regex>
```

```
<name>e-mail</name>
```

```
<pattern><![CDATA[^[a-zA-Z0-9_+&*-]+(?:\.[a-zA-Z0-9_+&*-]+)*@(?:[a-zA-Z0-9]+\.[a-zA-Z]{2,})$]]></pattern>
```

```
<description>A valid e-mail address</description>
```

```
</regex>
```

```
<regex>
```



6 Tests fonctionnels du développement

Vue initiale

Contacts

Genre	Prénoms	Noms	Mail	Tel
Aucun contenu dans la table				

Export JSONExport Vcard

Nom

Prénom

Genre

Date de naissance

Pseudonyme

Adresse

Numéro personnel

Numéro Professionnel

Mail

Lien Github/Gitlab

Créer

Modifier

Supprimer

Vue améliorée

Gestion de la liste des contacts

Liste de Contacts

Noms	Prénoms	Mail	Tel
Aucun contenu dans la table			

Nom*

izhdfz

Prénom*

izhdfz

Genre*

Choix du genre

Date de naissance*

05/06/1995

Pseudonyme

izhdfz

Adresse*

izhdfz

Numéro personnel*

izhdfz

Numéro Professionnel

izhdfz

Mail*

r@50.com

Lien Github/Gitlab

https://gemzo.com

Créer | Save

Modifier

Supprimer

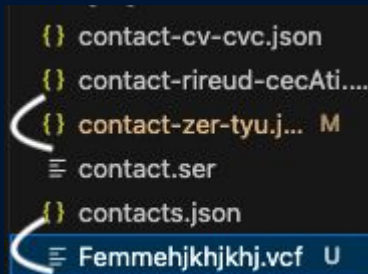
Version By R. et Y.L

Export JSON

Export Vcard

22

Test fonctionnel du développement



Test: Export JSON

Test: Export vCard

```
1  {
2    "mail": "o@gmail.com",
3    "prénom": "tyu",
4    "genre": "Homme",
5    "adresse": "32 pal 41000",
6    "date de naissance": "14/09/1994",
7    "lienGit": "https://www.google.com/search?q=jfx+:how+to+get+the+value+of+combobox+display&sca_esv=b5919fffce7396f9&sxsrf=ADLYWILHsYyXNIDBPVWjJHaU8a02P98klw:1723019948651&ei=rDKzZs23J4yJkdUPyI-SKA0&ved=0",
8    "nom": "zer",
9    "pseudo": "gu",
10   "telPerso": "0608090804",
11   "code Postale": "32 pal 41000"
12 }
```

≡ Femmehjkhkhj.vcf

```
1  BEGIN:VCARD
2  VERSION:4.0
3  N:hjk;hkhj;;Femme;
4  FN:hkhj hjk
5  BIRTHDAY:sdd
6  TEL;TYPE=persnum, voice;VALUE=uridsd fsdfs
7  TEL;TYPE=worknum, voice; VALUE=urifsd
8  ADR;TYPE=HOME; PREF=1; LABELfsdfs
9  EMAIL:e@gmail.com
10 LINKGIT:https://www.radix-ui.com/colors
11 END:VCARD
```



7 Points d'améliorations

Quelques pistes d'amélioration

- Contraindre à l'aide des REGEX les champs nom prénom et num pro et perso afin d'avoir qu'ils ne contiennent que les formats attendus
- Le datepicker devra se remettre à la date du jour après ajout d'une date de naissance.
- Ajouter un petit message quand le message a été créé et supprimé
- Logique de l'export Json : bug sur la sélection (à moitié résolu)
- Ajout des Loggers aux points critiques susceptibles bloquer l'application
- Ajout du QRCode et export CSV;
- Inclure une barre de recherche.



8

Conclusion

- Livraison de l'application

Merci pour votre écoute

