

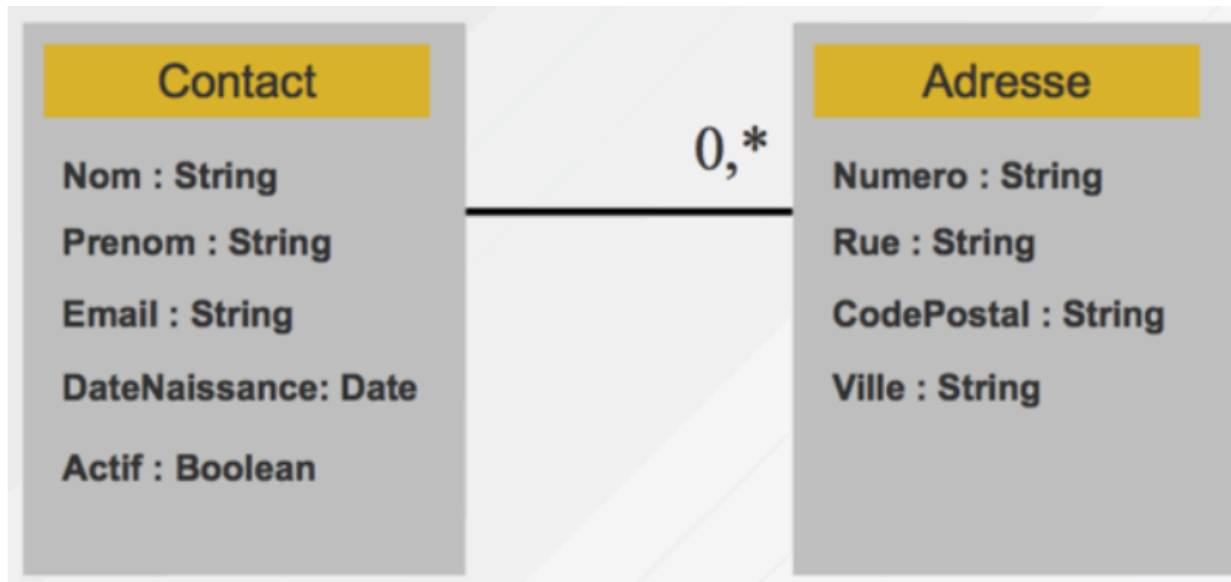
Projet Architecture

IHM: Contactor

Corentin Faucher
Amélie Maucade
Hyacinthe Malaspina

Description du projet

- Nous devons réaliser une application web permettant de gérer une liste de contacts
- Chaque contact peut disposer de plusieurs adresses
- Les classes contact et adresse doivent respecter l'architecture suivante



Fonctionnalités accueil

La page d'accueil permet:

- D'ajouter de nouveaux contacts
- Et d'effectuer une recherche de contacts avec tous les critères, email, noms, prénoms
- De visualiser la liste des contacts sous forme d'un tableau avec:
 - Nom et prénom
 - Date de naissance
 - Email
 - Actif

Fonctionnalités accueil

- Au clic sur un contact, un pop-up jaillit pour présenter ses différentes adresses en plus de ses nom, prénom, et date de naissance.
- La page d'accueil propose aussi un bouton d'édition de contact.

Page d'accueil

Contactor

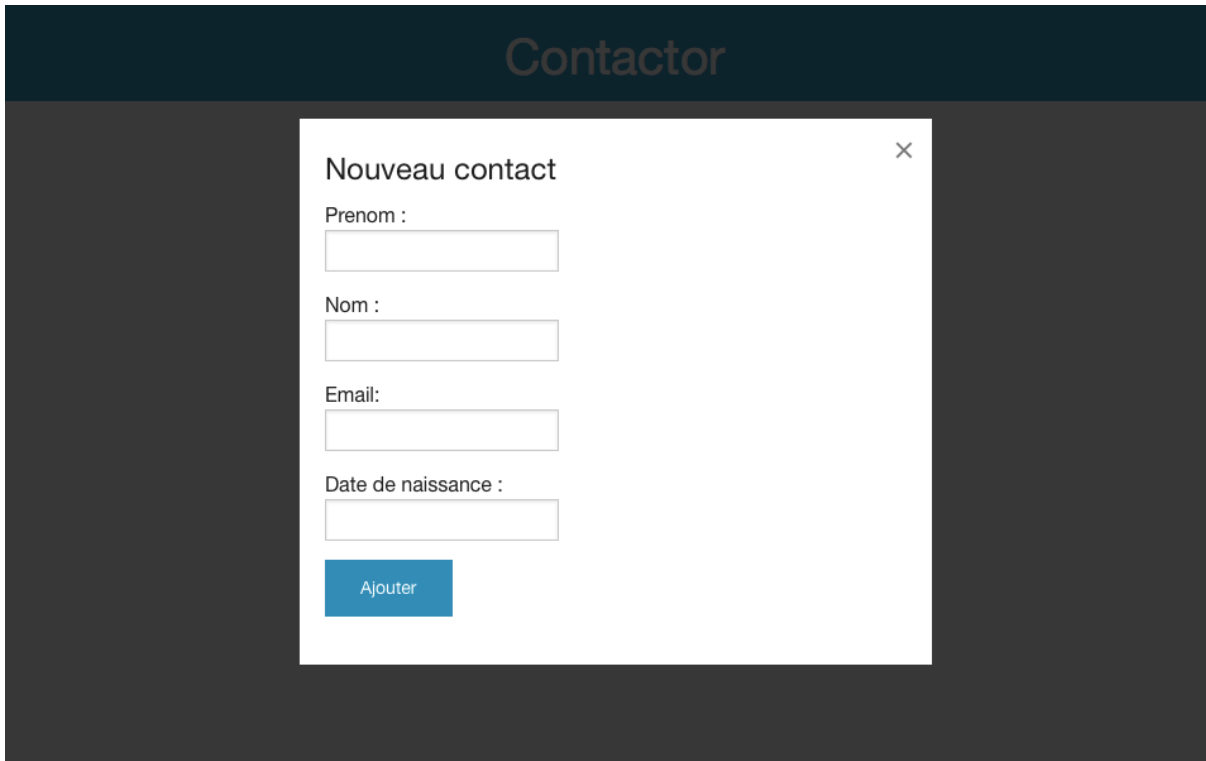
Votre liste de contact est vide.

Contactor

	Prenom	Nom	Date de naissance	Actif	Email	
🔍	Hyacinthe	Malaspina	02/10/2014	✖	malaspina@et.esiea.fr	✎
🔍	Amelie	Maucade	10/02/2014	✖	maucade@et.esiea.fr	✎
🔍	Corentin	Faucher	10/01/2014	✖	faucher@et.esiea.fr	✎

Fonctionnalité ajout de contact

- L'ajout d'un contact se fait par un pop-up
- On y saisit les informations: nom, prénom, email, et date de naissance.



The screenshot displays a web application interface with a dark blue header bar containing the word 'Contactor' in a light brown font. Below the header, a dark gray sidebar is visible on the left. The main content area features a white pop-up window titled 'Nouveau contact' with a close button (X) in the top right corner. The form inside the pop-up includes four input fields: 'Prenom :', 'Nom :', 'Email:', and 'Date de naissance :'. Each field is represented by a white rectangular box with a thin gray border. At the bottom of the form is a blue button with the white text 'Ajouter'.

Fonctionnalité édition de contact

La page édition de contact permet:

- De modifier les différentes informations du contact
- De rajouter des adresses grâce à un pop-up
- De définir si le contact est actif ou non
- D'éditer des adresses
- Et de supprimer le contact

Edition du contact

Contactor

← Retour

Prenom :

Hyacinthe

Nom :

Malaspina

Email:

malaspina@et.esiea.fr

Date de naissance :

04/08/2014

Actif :

☐

Adresses : +

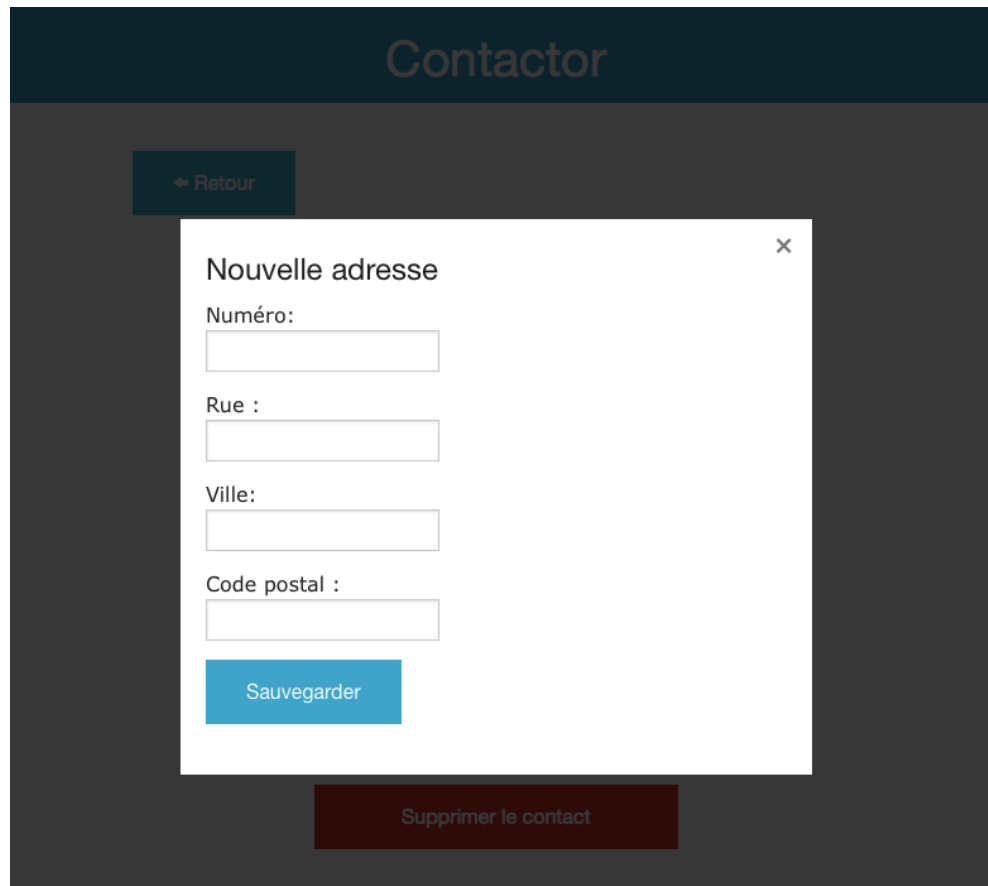
Sauvegarder

Supprimer le contact

Pop-up ajout d'adresse

Ce pop-up permet de remplir simplement les 4 champs de l'adresse du contact:

- Numéro de rue
- Rue
- Code postal
- Ville



The screenshot displays the 'Contactor' application interface. At the top, a dark blue header contains the word 'Contactor'. Below the header, a dark grey sidebar on the left features a 'Retour' button with a left-pointing arrow. The main content area is a light grey rectangle. In the center, a white pop-up window titled 'Nouvelle adresse' is open, featuring a close button (X) in the top right corner. The pop-up contains four input fields, each preceded by a label: 'Numéro:', 'Rue:', 'Ville:', and 'Code postal:'. Below these fields is a blue 'Sauvegarder' button. At the bottom of the main content area, a dark red button labeled 'Supprimer le contact' is visible.

Edition d'adresse

Cette page permet de modifier simplement les 5 champs de l'adresse du contact:

- Numéro de rue
- Rue
- Code postal
- Ville
- Définir comme
adresse de livraison

Contactor

← Retour

Numero :

Rue :

Ville :

Code Postal :

Adresse de livraison : ☐

Sauvegarder

Outils

- Le projet est réalisé en HTML, JavaScript, Java avec Spring MVC
- Nous utilisons le framework CSS Foundation.
- Et le script Magnific Pop-up pour des pop-ups plus élégants
- Nous codons sous Eclipse mais compilons et déployons avec la console maven
- Pour les merges nous utilisons Perforce

Architecture

- Les services Contacts et Adresse garantissent l'accès aux DAO
- Les DAO manipulent les données sous forme de HashMap
- Le service adresse peut traiter un contact pour en retourner sa liste d'adresses
- Quand on fait une recherche on applique un filtre à la liste de contacts ou d'adresse.
- Le HomeController fait le lien avec toutes les JSPs

Problèmes rencontrés

Notre principal problème a été la mise en place de l'environnement de développement.

Nous n'avons pas réussi à faire fonctionner Google App Engine + Spring + Maven dans Eclipse.

Donc nous avons utilisé maven en ligne de commande avec un des archétypes de Google et les dépendances de Spring.

Au final le projet s'est avéré beaucoup plus compliqué que ce que nous pensions, surtout à cause de la mise en place de l'environnement.

Conclusion

Ce projet nous a permis de découvrir Spring MVC, que nous sommes encore loin de maîtriser.

Nous avons appris à mieux utiliser Maven et Google App Engine.

Au final, nous avons trouvé ce projet très intéressant et nous avons apprécié découvrir ces technologies.