

# **IDCard**

Projet J2E

Numan Gezgin

Richard Haddad

Description et mise en place du projet

**Table des matières**

Informations préliminaires.....3

Fonctionnalités du projet.....4

Fonctionnement et copies d'écran.....6

Manuel de démarrage.....8

Conclusion.....9

# Informations préliminaires

Il existe en plus de ce document :

- Un **cahier des charges** (CDC-IDCard.pdf)
- Un **MCD** de la base de données (IDCard\_MCD.architect) lisible via le logiciel gratuit SQL Architect Community Edition
- Trois fichiers de **script SQL** permettant la mise en place de la base de donnée et des données de test
- Une **JavaDoc** dans le dossier apidocs/

Le tout est accessible depuis le site [https://chnapy.github.io/IDCard\\_Server/](https://chnapy.github.io/IDCard_Server/)

Le code source également via le **github** du projet : [https://github.com/Chnapy/IDCard\\_Server](https://github.com/Chnapy/IDCard_Server)

Nous avons utilisé une base de donnée **PostgreSQL**.

Le projet utilise également **Maven** pour certaines dépendances :

- **JOOQ** (une sorte d'ORM pour la base de donnée) : <https://www.jooq.org/>
- **Driver PostgreSQL** (le driver PostgreSQL, utilisé par JOOQ)
- **Jackson** (pour le JSON) : <https://github.com/FasterXML/jackson>
- **Reflections** (pour l'utilisation d'annotations personnalisées, via l'introspection) : <https://github.com/ronmamo/reflections>

Également, l'ensemble du contenu du package bdd.generated est un ensemble de classes générées par JOOQ.

L'application utilise **Java 1.8.0\_40** et a été testée avec **GlassFish 4.1.1**.

Les navigateurs testés : **Google Chrome 56, Mozilla Firefox 43**

Il est recommandé de feuilleter le cahier des charges afin d'avoir une idée de l'intérêt de l'application.

# Fonctionnalités du projet

Actuellement, l'application permet l'ensemble de ces actions :

- **Inscription**

En renseignant un pseudonyme, un email, et un mot de passe.

Une inscription réussie mène à une connexion automatique.

- **Connexion**

En renseignant un pseudonyme OU un email, et un mot de passe. L'application différencie automatiquement le pseudonyme de l'email, et agit en conséquence. La connexion utilise le système de session.

Une connexion réussie mène à l'accès à la page de configuration.

- **Accès à la page de configuration**

L'accès à la page de configuration se fait via la connexion, ou lors de l'accès au site déjà connecté.

Une fois arrivé sur la page, celle-ci liste l'ensemble des propriétés dont l'utilisateur possède au moins une valeur.

Chacune de ces propriétés contenant :

- Un nom (exemple : login)
- Un type (exemple : String)
- Une liste de valeurs, dont chacune contient :
  - Une valeur, changeable ou non selon les paramètres de la propriété
  - Une liste horizontale de sites web pouvant accéder à la valeur (représentés par leurs icônes)
  - A la condition que la valeur soit supprimable, un bouton de suppression
- Un bouton permettant d'ajouter une valeur

- **Ajout d'une valeur**

En cliquant sur le bouton d'ajout en bas de propriété, un champ input apparaît. Entrez alors une nouvelle valeur, puis appuyez sur entrée. La valeur s'ajoute à la liste.

- **Modification d'une valeur**

A condition que le champ soit modifiable, vous pouvez modifier la valeur d'un champ puis appuyer sur entrée pour valider.

- **Suppression d'une valeur**

A condition que la valeur soit supprimable (et donc que le bouton soit visible), vous pouvez cliquer sur le bouton de suppression, puis confirmer l'action. La valeur est alors supprimée.

- **Suppression d'un site**

Pour les valeurs où des sites sont présents, vous pouvez les supprimer en cliquant sur la

croix qui apparaît lorsque vous passer votre souris sur l'icône. Cliquez dessus, puis confirmez l'action. Le site est alors supprimé de la liste de visibilité.

- **Déconnexion**

Dans le header à droite cliquez sur le bouton « off » pour vous déconnecter. Confirmez l'action et vous serez redirigé sur la page d'accueil, déconnecté.

La persistance de chacune de ses actions peut être testée en actualisant la page.

A noter également la présence dans le header à droite d'un indicateur :

Il indique l'état de la dernière action. Ici on voit que l'action « Ajout de valeur » est un succès.

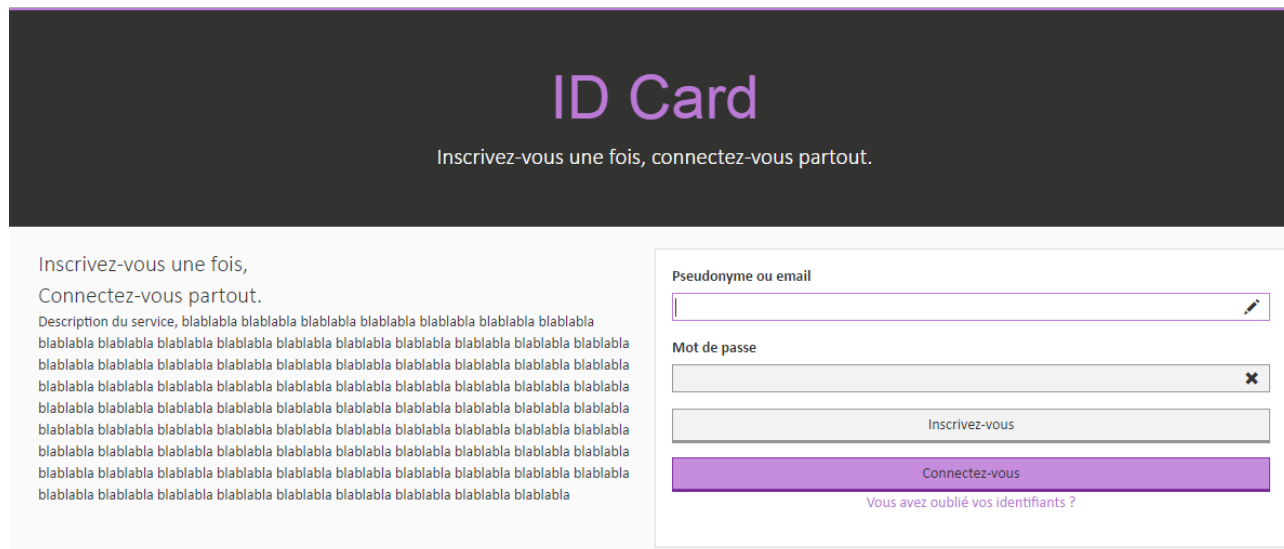


L'application est sécurisée. L'accès aux différents services est vérifié, les informations envoyées au serveur également.

# Fonctionnement et copies d'écran

L'accès au site se fait via l'url suivante (en localhost) : **http://localhost:8080/IDCard\_server/** (le port pouvant changer selon votre configuration).

Vous devriez arriver sur cette page :



The screenshot shows the 'ID Card' login page. At the top, there's a dark header with the text 'ID Card' in purple and 'Inscrivez-vous une fois, connectez-vous partout.' below it. The main content area is split into two columns. The left column contains the text 'Inscrivez-vous une fois, Connectez-vous partout.' followed by a long, repetitive string of 'blablabla' text. The right column contains a login form with fields for 'Pseudonyme ou email' and 'Mot de passe'. Below these fields are two buttons: 'Inscrivez-vous' and 'Connectez-vous'. At the bottom of the form, there's a link that says 'Vous avez oublié vos identifiants ?'.

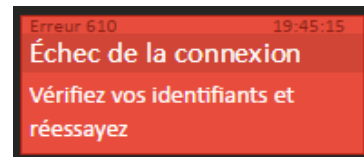
Il s'agit de la page d'accueil.

Vous pouvez vous **connecter** avec les identifiants de test : **chnapy – 123456**

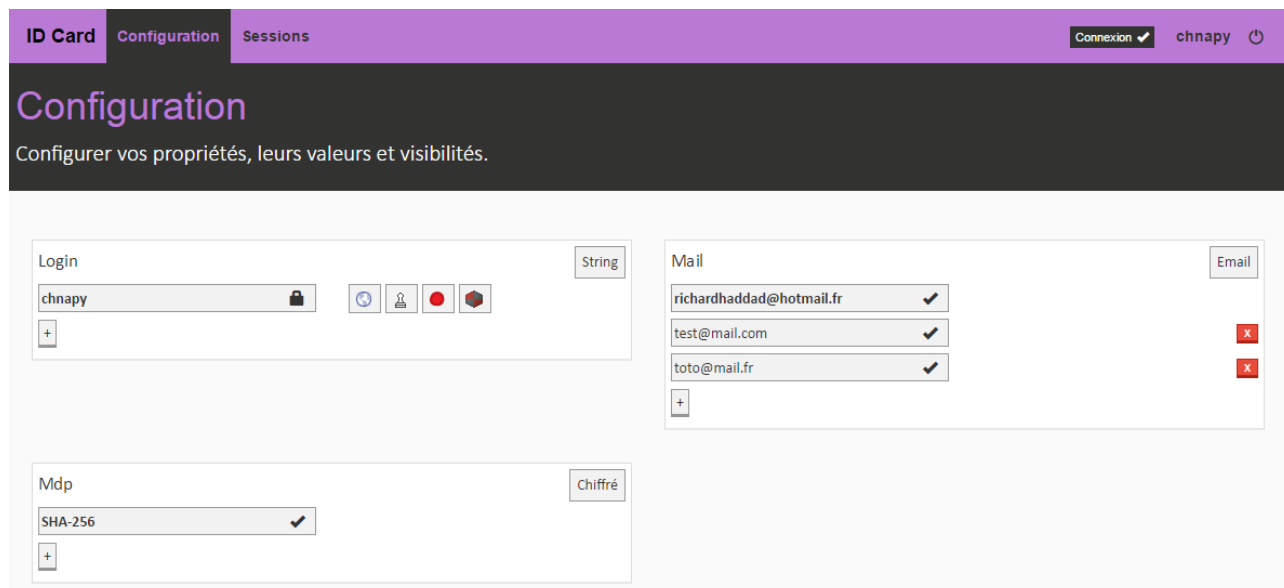
Ou à la place du login « chnapy » vous pouvez essayer **mail@mail.com**. Le site détecte seul s'il s'agit d'un email ou d'un login.

Ou alors vous **inscrire** avec vos propres données. Le site vous avertira si le format ne convient pas.

Lors de la connexion, si vous avez entré de mauvais identifiants, une alerte apparaîtra sur le côté droit :



Une fois connecté, vous arrivez sur la page de configuration :



The screenshot shows the 'ID Card' configuration page. At the top, there's a purple header with 'ID Card', 'Configuration', and 'Sessions' tabs. On the right, there's a 'Connexion' button with a checkmark and the text 'chnapy'. Below the header, the page title is 'Configuration' with the subtitle 'Configurer vos propriétés, leurs valeurs et visibilité.' The main content area is divided into three sections: 'Login', 'Mail', and 'Mdp'. The 'Login' section has a text input field with 'chnapy' and a 'String' button. The 'Mail' section has a list of email addresses: 'richardhaddad@hotmail.fr', 'test@mail.com', and 'toto@mail.fr', each with a checkmark and a red 'X' button. The 'Mdp' section has a text input field with 'SHA-256' and a 'Chiffré' button.

Les actions possibles sont alors multiples.

En premier lieu vous pouvez vous **déconnecter** en cliquant sur le bouton tout en haut à droite. Confirmez l'action et vous serez redirigé sur la page d'accueil, déconnecté.

Vous pouvez **modifier** une valeur en modifiant le contenu de son champ, puis en appuyant sur entrée.

Un point sur cette fonction : les non modifiable (avec un cadenas sur le champ) ne peuvent pas être modifié. De plus, il est déconseillé de vouloir modifier un champ de la propriété Mdp (mot de passe). Son fonctionnement particulier n'a pas encore été implanté, il est donc possible qu'une erreur se lance. Vous pouvez toujours essayer afin de voir comment le site gère les erreurs imprévues.

Vous pouvez **ajouter** une valeur, en cliquant sur le bouton « + » situé en bas de chaque propriété. Remplissez alors le champ apparu, puis appuyez sur entrée. Le champ est alors ajouté.

Sur les valeurs supprimable, vous pouvez cliquer sur le bouton « x » afin de **supprimer** la valeur. Confirmez l'action, le champ disparaît alors.

# Manuel de démarrage

L'application utilisant plusieurs dépendances Maven, vous devez configurer votre outil de travail afin qu'il prennent en compte le fichier **pom.xml** (à la racine) qui gère les dépendances Maven. Faîtes également attention de bien utilisé la version 8 de Java. L'application a été testée avec GlassFish mais peut normalement fonctionner avec TomCat, sans garantie néanmoins.

La base de donnée doit être **PostgreSQL** (a priori une version plutôt récente, testé avec la 9.6).

D'autres points doivent être définis :

- l'emplacement de la base de donnée : **localhost**
- le port : **5432**
- le nom de la base : **idcard**
- le nom du rôle : **idcard**
- le mot de passe du rôle : **Vtffede1**

L'ensemble de ces 5 propriété peut être modifié dans la classe Java **bdd.BDDInfos**.

Dans tous les cas, il faut qu'il y ait un schéma nommé **public**.

Ensuite, Les trois scripts SQL fournis doivent être utilisés dans cet ordre :

- **installation\_bdd.sql** : créé les tables etc
- **donnees\_requises.sql** : ajoute les données nécessaires au fonctionnement du site
- **donnees\_test.sql** : ajoute des données de test

Une fois tout ces points réglés, vous pouvez lancer l'application et accéder à la partie précédente [#2.Fonctionnement et copies d'écran](#)



## Conclusion

Hormis la partie sur les mots de passe, nous estimons que l'ensemble des fonctions implémentées fonctionnent parfaitement.

Il reste bien sûr de nombreux points à améliorer. Notamment les informations de la base de donnée doivent être stockées dans un fichier à part (JSON/XML) plutôt que dans une classe Java. Aussi la partie modèle du code mériterait un certain nettoyage.

De très nombreuses fonctions restent à implémenter : l'oubli de mot de passe, la confirmation d'inscription par mail, la suppression d'une propriété, l'ajout de sites sur les valeurs, pouvoir trier les propriétés, afficher les sessions en cours et toutes les actions relatives à celles-ci.

Et comme indiqué dans le cahier des charges, rendre l'application accessible sous forme d'API par les applications et sites externes.

Un vrai chantier en somme.