**Université Paul Sabatier - STRI**

2016

**Projet JAVA M1**

**CONNECT !**

**HERNANDEZ Quentin VAURIGAUD Jordan IUNGMANN Victor**

Sommaire

[I. Cahier des Charges 4](#_Toc442812224)

[II. Architecture 5](#_Toc442812225)

[III. Analyse Conceptuelle 6](#_Toc442812226)

[A. Diagramme de cas d’utilisation 6](#_Toc442812227)

[B. Diagramme de classe. 6](#_Toc442812228)

[C. Diagrammes d’activités. 6](#_Toc442812229)

[D. Diagramme de séquence. 6](#_Toc442812230)

[IV. Protocole entre Client/Serveur 6](#_Toc442812231)

[A. Connexion 6](#_Toc442812232)

[B. Déconnexion 6](#_Toc442812233)

[C. Modification des informations 6](#_Toc442812234)

[D. Inscription 6](#_Toc442812235)

# Suivi du Projet

|  |  |  |  |
| --- | --- | --- | --- |
| Auteur | Date | Version | Commentaires |
| Groupe | 10/02/2016 | V1.0 | Présentation du sujet + Conception de la phase 1 |
|  |  |  |  |
|  |  |  |  |

# Cahier des Charges

Le client nous a demandé de concevoir et réaliser une application d’annuaire d’étudiants.

Celui-ci devra être construit avec une architecture client-serveur. Le langage imposé est le Java.

Un étudiant renseignera les informations suivantes :

* Nom
* Prénom
* Année de diplomation
* Adresse e-mail
* Numéro de Téléphone

Un étudiant doit pouvoir s’inscrire, consulter la liste des étudiants, rechercher un étudiant et consulter la fiche d’un étudiant.

Il peut également modifier les informations de sa fiche et choisir de les partager avec les visiteurs.

# Architecture

Afin de pouvoir travailler A TOUT MOMENT depuis chez nous, nous utilisons un gestionnaire de version qui git hébergé chez GitHub ([https://github.com/Shuntor/JavaM1](https://github.com/Shuntor/JavaM1%20) ).

La base de donné sera avec un format MySQL. La liaison avec le programme sera donc faite à l'aide du plug-in JDBC. Les membres du projet travaillant sous des systèmes d'exploitation différents nous utilisons Maven pour gérer les dépendances entre ces derniers.

Pour se connecter l'étudiant utilisera son adresse mail en tant que identifiant accompagné d'un mot de passe qu'il aura choisi lors de la création de son compte. Nous avons fait le choix de ne pas créer de pseudonymes afin de faciliter la recherche des étudiants entre eux.

# Analyse Conceptuelle

## Diagramme de cas d’utilisation

## Diagramme de classe.

## Diagrammes d’activités.

## Diagramme de séquence.

# Protocole entre Client/serveur

## Connexion

Tout d’abord, il faut définir les informations qui seront utilisées par le client pour se connecter.   
Le nom ou le prénom : impossible car il y a des possibilités d’homonymes  
On choisira donc d’utiliser comme identifiant, l’adresse e-mail. Evidemment, on l’associera un mot de passe définit par l’utilisateur lors de l’inscription.

Afin de savoir si un client est connecté ou non, on pourrait assigner une variable dans un champ de la trame émise lorsque ce dernier demande à consulter le profil d’une personne afin de ne lui envoyer seulement les informations auquel il a le droit de regard.   
Il s’agit d’une approche comprenant énormément de risques.

On opte donc pour assigner une variable au niveau du thread, sur le serveur.

La trame sera envoyé comme suit :

Et la réponse du serveur sera :

## Déconnexion

Il y a deux cas pour lesquels une déconnexion est demandée par le client :

* Simple déconnexion du compte.
* Déconnexion du compte suite à l’arrêt du programme client. Dans ce cas il faudra également mettre fin au Thread

Les messages envoyés dans chacun des cas respectivement seront :

## Modification des informations

On récupérera toutes les données de l’étudiants depuis le serveur et afin qu’il puisse les modifier par la suite.

## Inscription

L’étudiant enverra les informations nécessaires à la constitution de sa fiche personnelle lors de l’inscription.