

## Rapport Projet 1

### Introduction

Nous avons commencé le projet le 19 octobre et nous l'avons terminé le mercredi 21 octobre. Avec une base de 5 heures de travail par jour chacun, nous arrivons à une charge horaire totale de 30 heures de travail sur ce projet.

Le travail demandé n'était en soi pas très long ni compliqué mais il s'agissait surtout pour nous de nous acclimater à ce nouveau langage qu'est Java. Bien comprendre le fonctionnement des servlets, des JSP, des javabeans et de la connexion à la base de données constituait le gros du travail. Nous sommes donc satisfaits de pouvoir rendre ce projet puisqu'il représente la preuve que nous avons assimiler les concepts vus en cours et mis en pratique dans notre web application.

Tout ce qui était demandé dans le cahier des charges est présent dans notre projet, à savoir :

- Une login page qui différencie instructeur et étudiant et renvoie vers une page d'erreur en cas de mauvais identifiants
- Une session qui se crée après un login réussi
- Un cookie qui se crée après un login ou une registration réussi et qui s'enregistre dans le navigateur du client pendant 24h.
- Afficher une liste de todos
- Pour l'instructeur : ajouter, modifier et supprimer ses todos

Nous avons également pris la liberté d'ajouter certaines fonctionnalités :

- Une registration page qui va créer un nouvel utilisateur dans la DB (forcément un étudiant)
- L'instructeur ne voit que les tâches qu'il a créées et non celle créées par un autre instructeur
- L'instructeur a la possibilité d'assigner une nouvelle tâche à des étudiants en particulier et non à tous les étudiants en même temps

Afin de vous connecter en tant qu'instructeur, vos identifiants sont :

- Username : [nada.nahle@ext.devinci.fr](mailto:nada.nahle@ext.devinci.fr)
- Password : NN (vos initiales)

Pour vous connecter en tant qu'étudiant, vous pouvez simplement créer un nouvel utilisateur qui sera automatiquement enregistré comme étudiant dans la DB. (ou alors : [clement.mutez@edu.devinci.fr](mailto:clement.mutez@edu.devinci.fr), password : cm)

### WebToDoList

#### Database

La base de données MySQL

Pour notre base de données MySQL, nous avons décidé de partir sur deux tables différentes :

- Une table **User** qui stocke les utilisateurs du site et qui a pour attributs :
  1. **Id\_user** qui différencie chaque utilisateur entre eux
  2. **first\_name**
  3. **last\_name**
  4. **username** pour se login

5. **password** pour se login
  6. **instructor** qui est un booléen permettant de savoir si l'utilisateur est un étudiant (false) ou un instructeur (true)
- Une table **Todo** qui stocke tous les todos des étudiants et qui a pour attributs :
    1. **Id** qui différencie les todos entre eux
    2. **Description** qui contient le texte du todo
    3. **Id\_creator** pour enregistrer l'id du créateur du todo (nécessaire pour la fonctionnalité supplémentaire que nous avons faite)
    4. **Id\_user** pour affilier le todo à un étudiant (nécessaire pour la fonctionnalité supplémentaire que nous avons faite)

Avec cette disposition de schéma, un étudiant peut avoir plusieurs todos et un todo peut être distribué à plusieurs étudiants. On a donc une relation (n,n) entre ces deux tables.

### UserDBManager

Cette classe est l'une des plus importantes de notre application puisqu'elle est responsable de la connexion avec la DB. Dans tous les servlets, on va créer un objet de cette classe afin de pouvoir appeler les méthodes de cette dernière qui va envoyer les requêtes SQL à la DB.

Entre autres, on retrouve des fonctions qui vont récupérer les informations d'un utilisateur en fonction de son id ou de son username et de son password ; celles qui vont ajouter, modifier ou supprimer un todo ; celles qui vont ajouter un nouvel utilisateur ; ou encore celles qui vont récupérer la liste des todos créés par un même instructeur ou celle pour un étudiant.

### Javabeans / Helper class

On a logiquement créé autant de helper classe que de tables dans notre schéma de base de données, à savoir 2 : User et Todo.

Ces dernières ont exactement les mêmes attributs que les tables de notre BDD.

On a eu besoin de surcharger les constructeurs de ces deux classes pour différentes raisons :

- Pour la classe User, on voulait pouvoir créer un objet User au sein de l'application sans avoir son id étant donné que l'id d'un user n'est créé qu'au moment de son enregistrement dans la DB.
- Encore pour la classe User, on voulait pouvoir instancier un user avec pour seuls paramètres son id, son prénom et son nom afin de récupérer uniquement les informations dont on avait besoin pour la méthode `getStudents` qui renvoie une liste des utilisateurs étudiants pour afficher juste leur nom et prénom.
- Pour la classe Todo, on a eu besoin de pouvoir instancier un todo avec pour seuls paramètres la description et l'id du créateur ce qui nous est utile lorsque l'on n'a pas encore d'id pour le todo (car ce dernier est créé au moment de la création dans la DB) et lorsque l'on ne veut pas donner ce todo à un user en particulier.

### Servlets et JSP associés

Les servlets servent à faire la connexion entre la DB et l'interface utilisateur (JSP). Lorsque l'utilisateur rentre des informations sur le site ou clique sur un bouton, les servlets vont capter ses

actions/interactions et agir en conséquence cad en changeant l'interface du site dynamiquement (en communiquant avec le JSP) ou en envoyant des requêtes à la base de données MySQL.

A chaque servlet est associé sa page JSP à l'exception du servlet pour la fonction delete qui ne nécessitait pas d'aller sur une nouvelle page mais simplement de mettre à jour la main page en enlevant le todo supprimé.

#### LoginServlet et Login.jsp



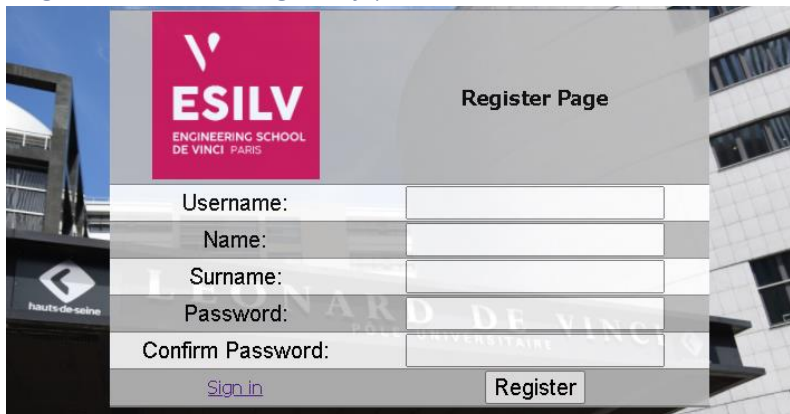
Dès qu'on se connecte à

<http://localhost:8080/WebToDoList>,

On est dirigé vers la page Login. Ayant déjà été connecté dans les dernières 24h, les cookies de username et password remplissent automatiquement les champs.

On peut alors choisir de se login ou bien de se créer un nouveau compte en cliquant sur Registration.

#### RegisterServlet et Register.jsp



La Register page va enregistrer un nouvel élève dans la base de données.

Un nouvel instructeur ne peut pas s'enregistrer à partir d'ici, il faudrait le faire directement dans MySQL.

#### ToDoServlet et Main-page.jsp



La main page se décline en 2 versions :

- Une version étudiante qui ne fait qu'afficher ses todos
- Une version instructeur qui permet d'ajouter, modifier et supprimer les todos

### AddToDoServlet et Add-todo.jsp

Pour ajouter un Todo, l'instructeur doit d'abord écrire la description de ce Todo (300 caractères maximum sans caractères spéciaux ni accent).

Ensuite il peut choisir à quels étudiants affilier ce Todo avant de cliquer sur Save et enregistrer sa requête.

S'il souhaite annuler sa requête, il peut simplement cliquer sur Back to main page.

Description	Action
Profiter des vacances	<a href="#">Edit</a>   <a href="#">Delete</a>
deployer le projet	<a href="#">Edit</a>   <a href="#">Delete</a>
<a href="#">Add a Todo</a>	<a href="#">Log Out</a>

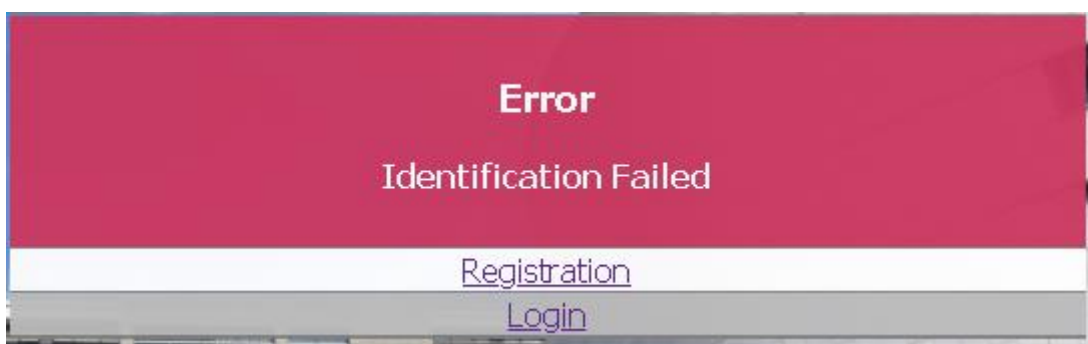
### EditTodoServlet et Edit-todo.jsp

Pour éditer un Todo, il suffit de réécrire la description et de cliquer sur Save (Back to main page pour annuler).

Description	Action
Profiter des vacances	<a href="#">Edit</a>   <a href="#">Delete</a>
deployer le projet sur mac	<a href="#">Edit</a>   <a href="#">Delete</a>
<a href="#">Add a Todo</a>	<a href="#">Log Out</a>

### Error.jsp

La page d'erreur est affichée lorsque l'utilisateur rentre de mauvais identifiants au moment du login, lorsqu'un nouvel utilisateur qui veut s'enregistrer rentre un username déjà emprunté, ou alors un étudiant veut accéder aux fonctionnalités de l'instructeur (Add, Edit et Delete)



### Session + cookie

Les sessions et les cookies sont presque similaires en ce sens qu'ils enregistrent des informations à la manière de constante globale pour les afficher à un moment donné sur le site :

- Pour la session, cette dernière se crée après le login et enregistre l'id de l'utilisateur. Ensuite sur chaque page, on récupère les informations de l'utilisateur à partir de son id avec la méthode `getUserById(id)` pour récupérer le nom et prénom et afficher le message de bienvenue sur chaque page (main, add, edit).
- Pour les cookies, ils sont au nombre de 2. Un pour enregistrer le username et l'autre pour le password de l'utilisateur. Ils se créent lorsque l'utilisateur se login ou lorsqu'un nouvel utilisateur s'enregistre avec succès.