Ismail BLLADI & Oumaima Khalil

06/12/2018

RAPPORT LO54

Application web pour la gestion d’une ecole privee

Table des matières

[Introduction 2](#_Toc532926332)

[1- Technologies utilisées : 3](#_Toc532926333)

[a) Spring Data : 3](#_Toc532926334)

[b) Spring Boot : 3](#_Toc532926335)

[c) Spring Security : 4](#_Toc532926336)

[d) Thymeleaf : 4](#_Toc532926337)

[2- Environnement de développement : 4](#_Toc532926338)

[a) WampServer : 4](#_Toc532926339)

[b) IntelliJ IDEA : 4](#_Toc532926340)

[3- Architecture de l’application : 5](#_Toc532926341)

[4- Le guide d'utilisation de l'application : 7](#_Toc532926342)

[Conclusion 13](#_Toc532926343)

# Introduction

Dans le cadre de l’UV LO54, nous devions apprendre à nous servir d’une technologie liée à Java Entreprise Edition pour développer un projet de gestion d’offre de formation d’une école privée.

Nous avons choisi Spring comme étant un Framework bien connu des développeurs Java pour les nombreuses fonctionnalités qu’il apporte sur les aspects web, sécurité, batch ou encore accès aux données dans le cadre du développement d’une application.

Dans ce rapport nous allons présenter la technologie et ses avantages dans le développement des applications Web Java, ainsi on va se référencier de notre projet pour expliquer comment mettre en place Spring Boot, puis pour conclure nous parlerons de notre expérience vis-à-vis de cette technologie.

# Technologies utilisées :

## Spring Data :

Un projet créé par Pivotal, qui fait partie de Spring IO Platform, il y a quelques années pour faciliter l’écriture du code d’accès aux données, pour les différentes bases existantes, relationnelles et non relationnelles.

Il se base sur la technique du mapping objet-relationnel (ORM) qui définit des correspondances entre la base de données et les objets programmés.

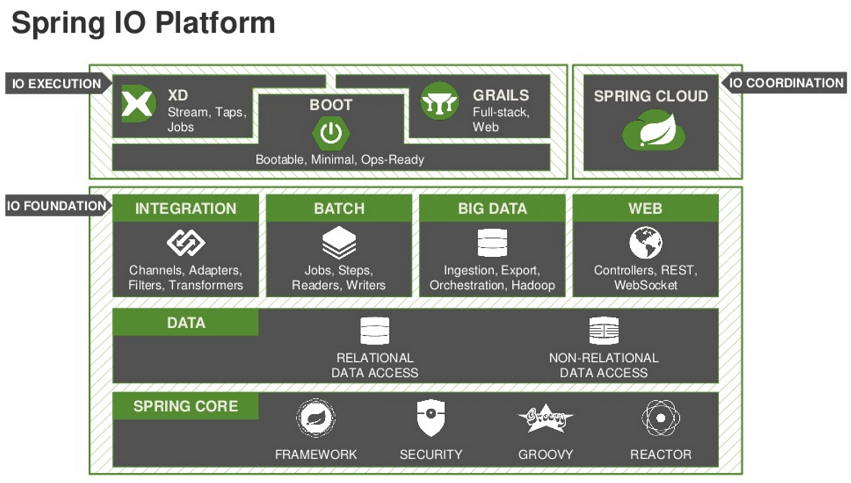


Figure 1 Spring IO Platform

## Spring Boot :

Il sert à créer facilement des applications Spring et les configurer automatiquement autant que possible. Il fournit des fonctionnalités prêtes pour la production, et un fichier POM pour simplifier la configuration Maven (gestion de dépendances). Ainsi, il permet de déboguer l’application et de cibler les erreurs de manière efficace.

## Spring Security :

C’est un Framework puissant et hautement personnalisable d’authentification et de contrôle d’accès qui prend en charge ces deux fonctionnalités de façon complète et extensible.

Il permet la protection contre les attaques informatiques telles que la fixation de session, le clickjacking (détournement de clic), et le Cross-site Request Forgery (CSRF) qui est l’une des attaques les plus répandues.

## Thymeleaf :

C’est un moteur de Template côté serveur, écrit en Java, pour les environnements Web et autonomes. Il peut générer des pages HTML5 qui peuvent être correctement affichées dans les navigateurs. Il propose un ensemble d’intégration avec Spring qui permet de l’utiliser comme substitut de JSP. Et il commence à être utilisé de plus en plus par les entreprises.

# Environnement de développement :

## WampServer :

C’est une plateforme de développement Web sous Windows pour les applications Web dynamiques à l’aide du serveur Apache2, et d’une base de données MySQL. Il possède également PhpMyAdmin pour gérer plus facilement les bases de données avec laquelle nos applications sont connectées.

## IntelliJ IDEA :

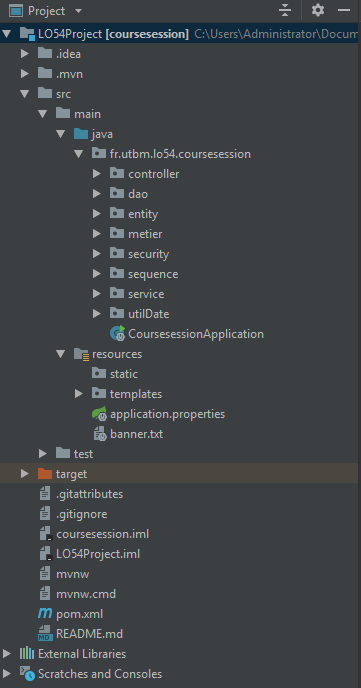
IntelliJ IDEA est un [IDE](https://fr.wikipedia.org/wiki/Integrated_Development_Environment) [Java](https://fr.wikipedia.org/wiki/Java_(langage_de_programmation)) commercial développé par [JetBrains](https://fr.wikipedia.org/wiki/JetBrains" \o "JetBrains)

# Architecture de l’application :

L’application est composée de trois couches qui sont rigoureusement séparées :

* Couche DAO (Objet d’accès aux données) : permet les accès à la base de données à travers le Framework Hibernate et le JPA Repository de Spring Data.
* Couche Métier (ou couche de service) : elle organise les accès à la couche DAO et ses aspects transactionnels qui sont gérés par Spring Framework.
* Couche Présentation : c’est la couche d’entrée dans l’application du point de vue de l’utilisateur final. Elle fait appel aux traitements de la couche Métier en fonction des actions effectuées par l’utilisateur et récupère les données retournées pour les afficher dans les pages HTML générées par Thymeleaf.

Figure 2Architecture Application

Après avoir créé le projet Spring basé sur Spring Boot, cela va nous générer un projet MAVEN :

Couche Présentation

Couche Métier

Couche DAO

Figure 3 Arborescence du projet

# Le guide d'utilisation de l'application :

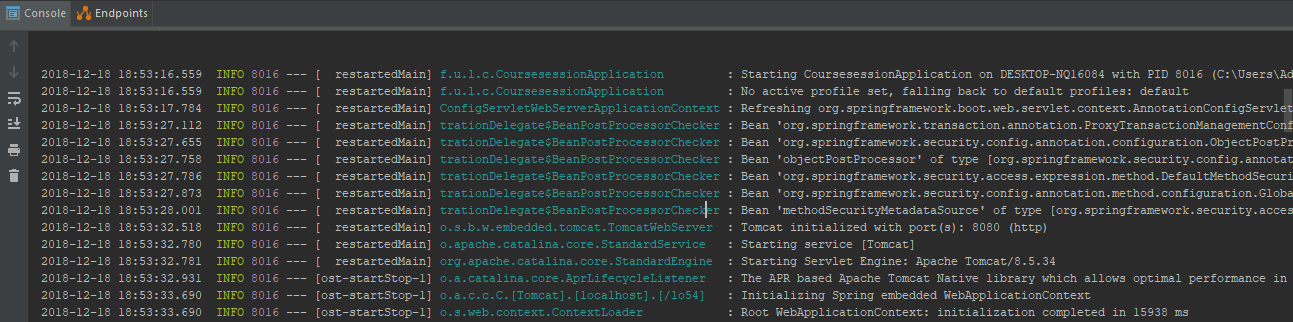
Après avoir démarré tous les services du WampServer relatifs à la base de données MySQL, on peut exécuter l’application programmée :

Figure 4 Console

On a choisi lors du démarrage de l’application d’alimenter la base avec des informations :

Course crs1 = new Course("CRS\_001","J2EE/JAVA","JAVA et JEE sont les technologies les plus " +  
 "répandues pour le développement des applications. La Formation JAVA pour devenir Développeur Java/JEE , " +  
 "proposée par LO54 est un cursus unique qui permet rapidement d'être opérationnel sur ce langage.");  
courseRepository.save(crs1);  
  
Location l2 = new Location("Belfort");  
locationRepository.save(l2);  
  
Client c = new Client("Blladi", "Ismail", "8 Rue de Ferrette", "062586487","pass", passwordEncoder.encode("pass"),"ADMIN");  
clientRepository.save(c);

ETC…

On aura le résultat suivant : -Exemple-

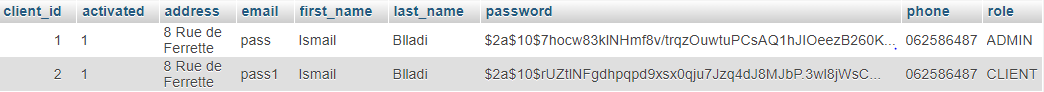


Figure 5 Table Client

L’application va générer les tables suivantes :

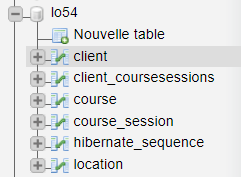
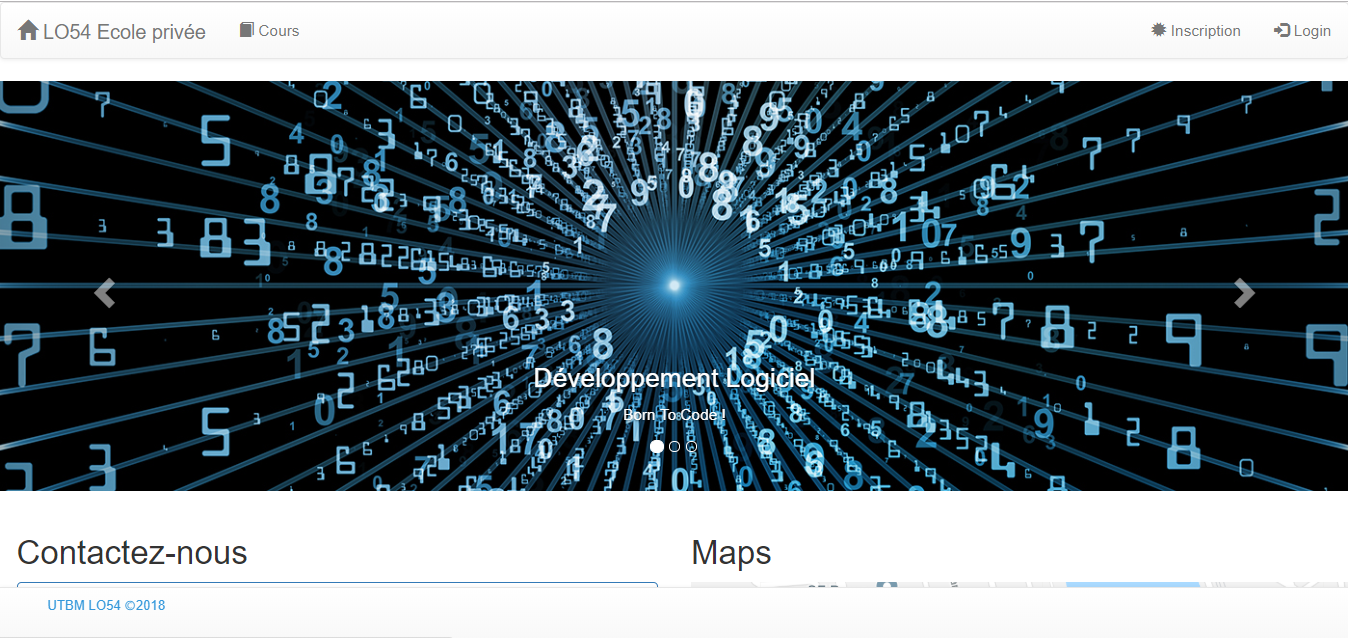


Table de jointure

Figure 6 Tables de bases de données

Passons à l’application :

* ***Page d’accueil :***

3

2

1

Figure 7 Page d'accueil

1

Le client peut voir les cours et les sessions proposées afin qu’il puisse s’inscrire dans ces derniers ainsi que dans la plateforme de l’école

2

Il peut également s’inscrire directement sur la plateforme

Comme il peut se connecter

3

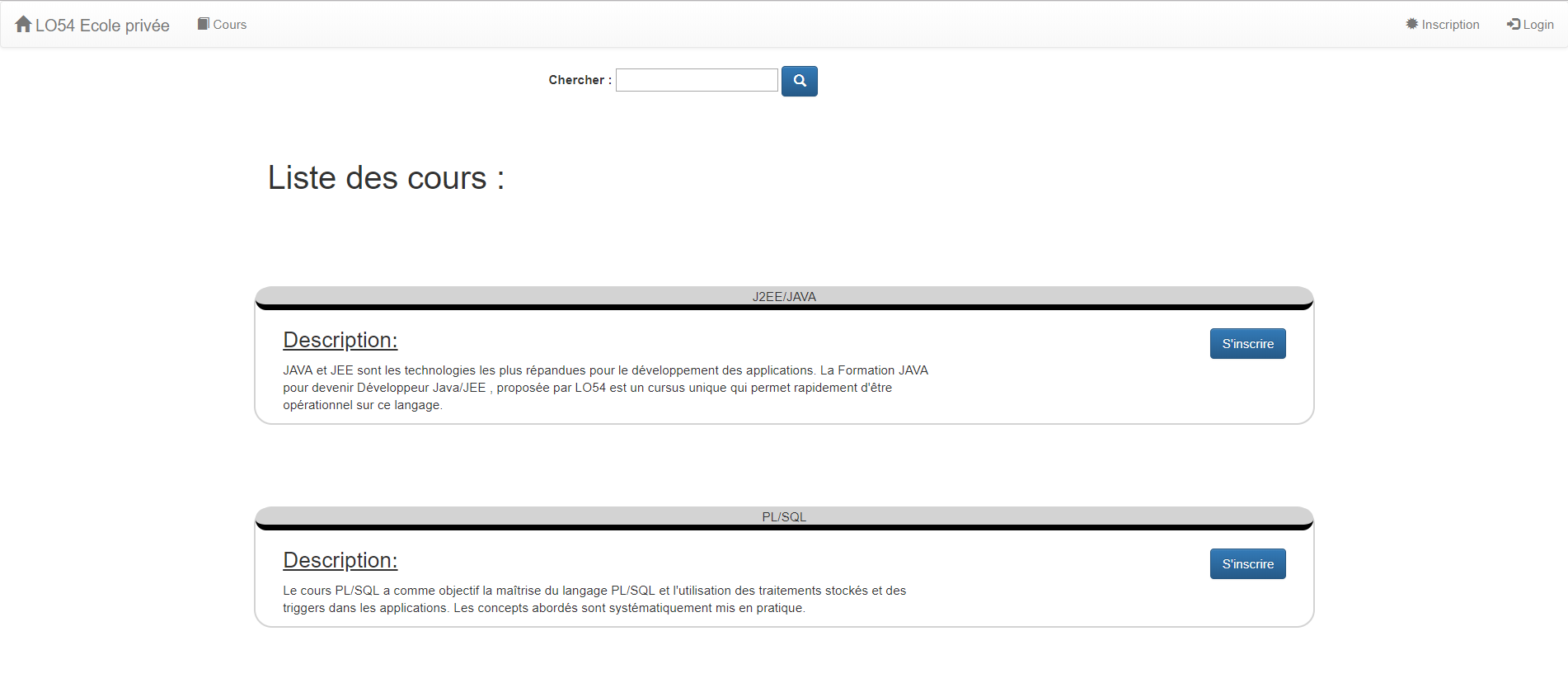
* ***Page des cours***
  + Contient tous les cours avec un filtre de recherche et une pagination limité à 4 cours

Figure 8 Page des cours

* ***Page des sessions***
  + La page des sessions affiche les sessions qui sont en relation avec le cours choisi
  + Deux filtres de recherches, qui peuvent être utilisés simultanément

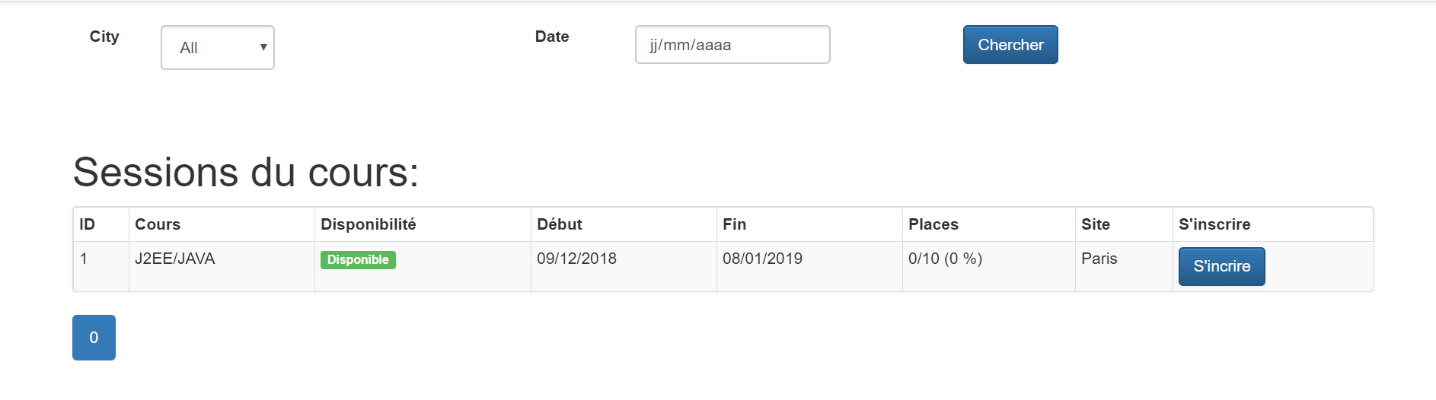


Figure 9 Page des sessions

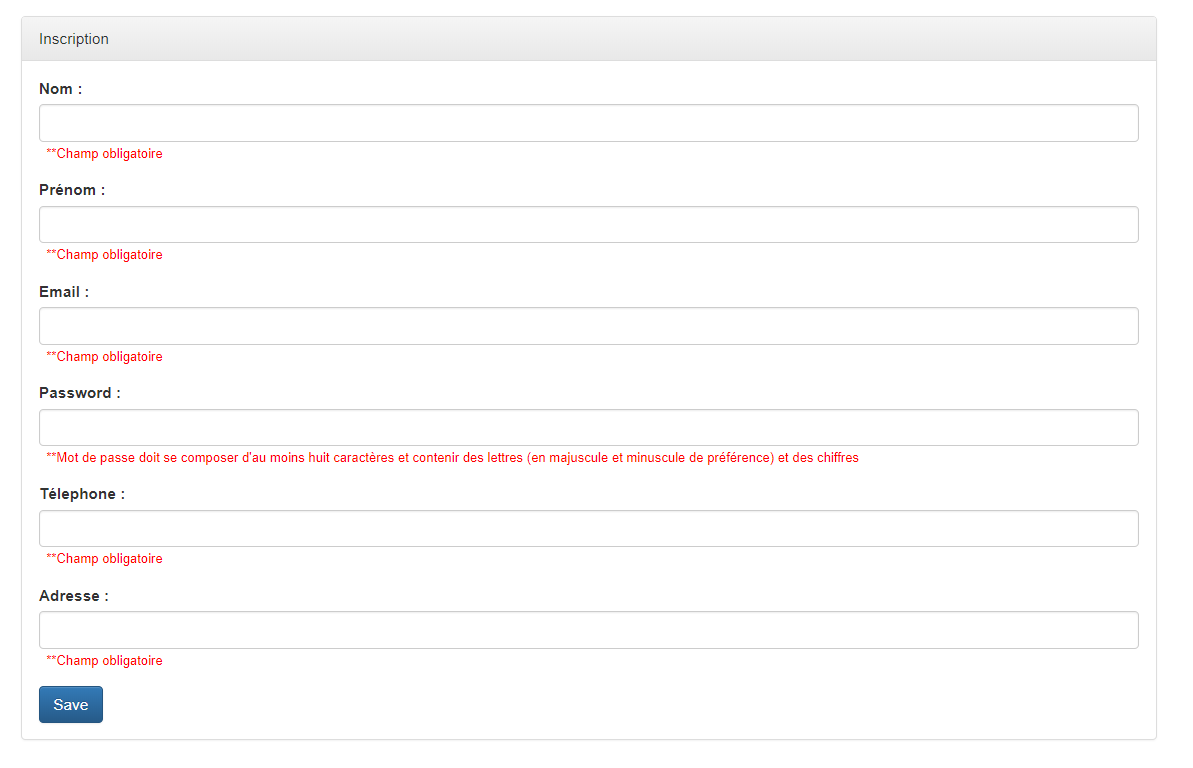
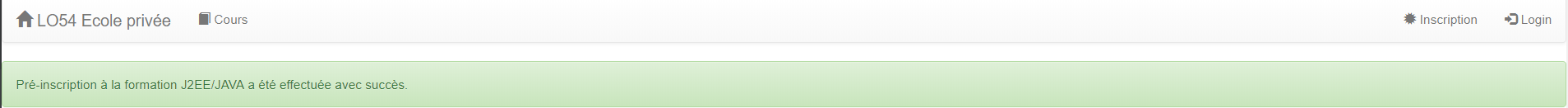
* ***Formulaire d’inscription***
  + Afin de finaliser la pré-inscription, il sera demandé à l’utilisateur de remplir un formulaire afin de garder ses informations pour que se connecter et voir les sessions sur lesquelles il s’est inscrit

Figure 10 Formulaire d'inscription

Message après pré-inscription :



* ***Page Login :***

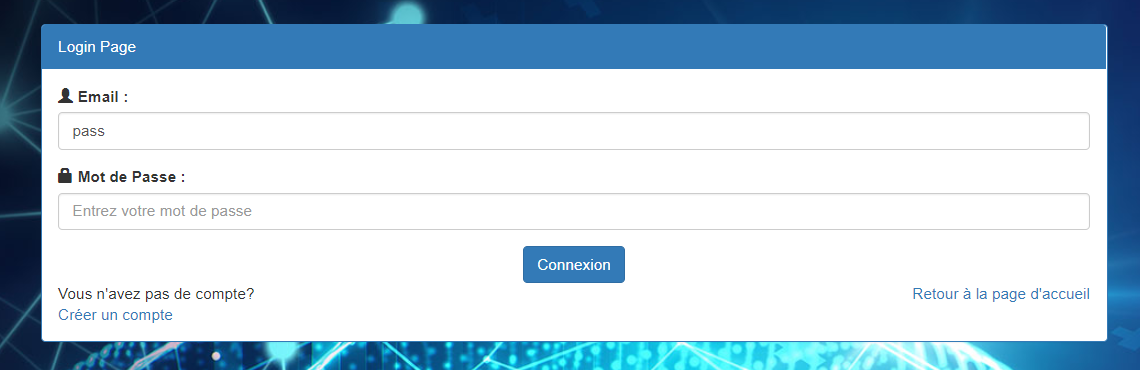
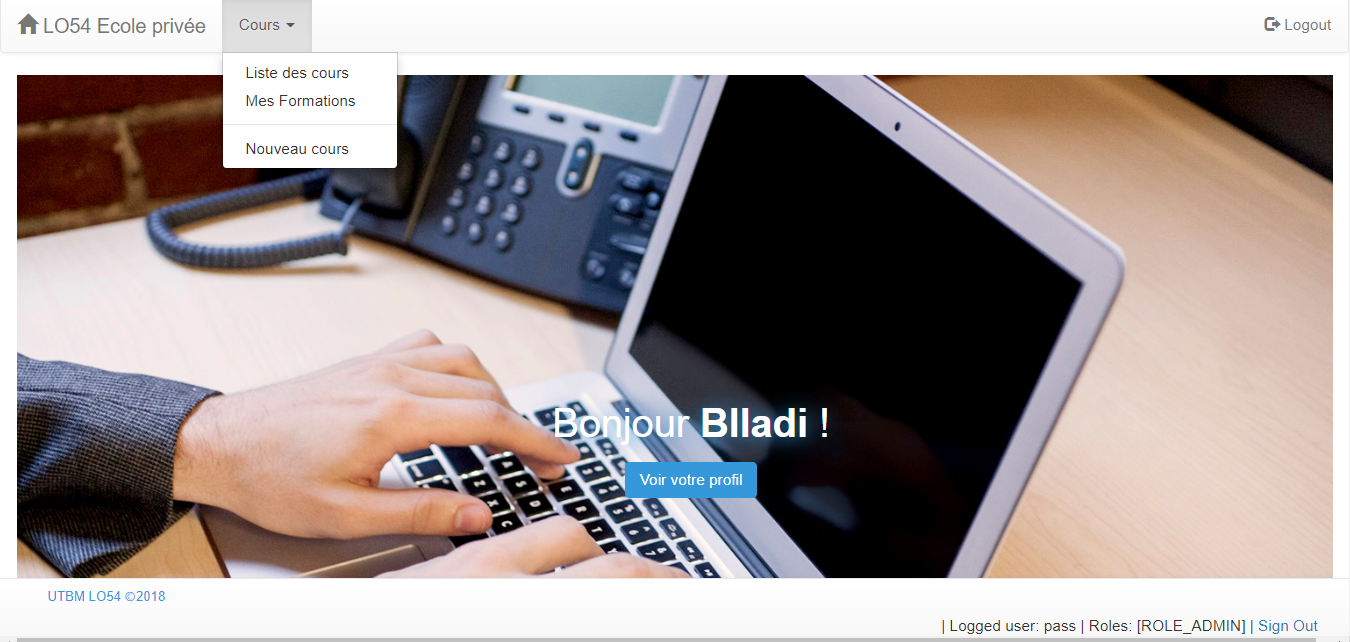


Figure 11 Page Login

* + ***Page d’administrateur :***

L’administrateur peut ajouter un nouveau cours et une nouvelle session et de modifier les données existantes (grâce à des formulaires)

* + ***Page du client :***

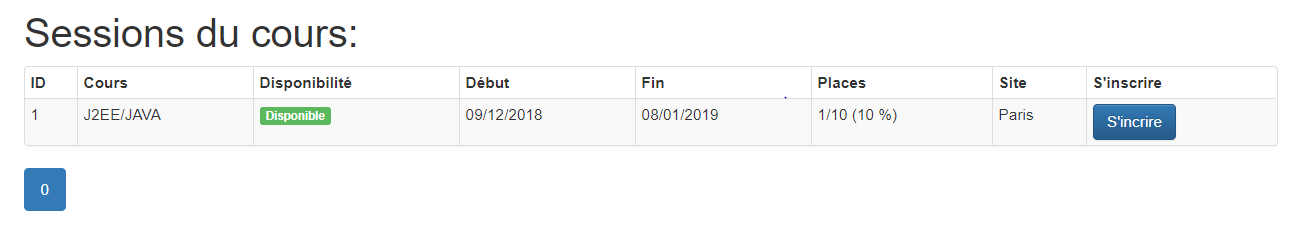
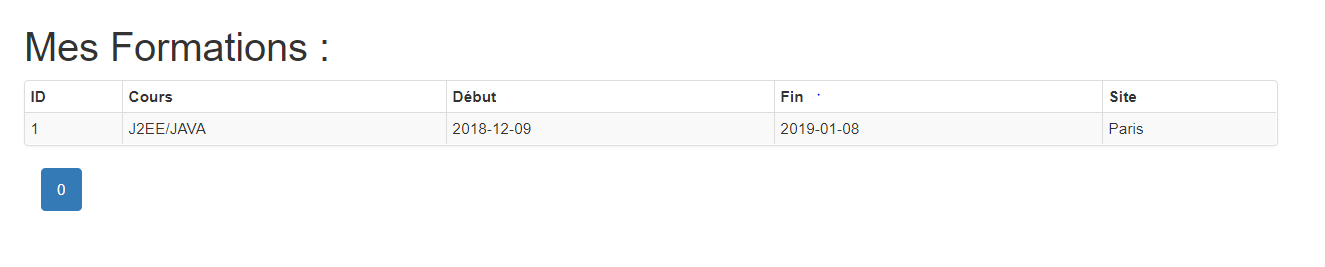
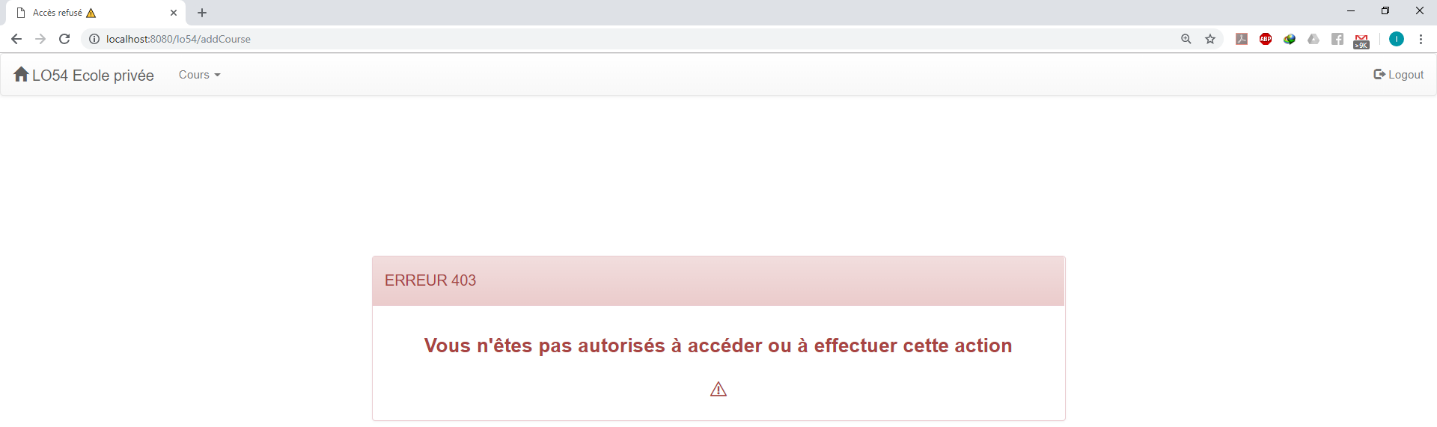
Le client ne peut que consulter son profil et les formations pour inscription ou les formations sur lesquelles il est inscrit, avec une restriction sur toute action propre à l’administrateur :

Figure 12 Restriction pour l'ajout de cours

Figure 13 Places et pourcentages des inscrits

Figure 14 Inscription personnelle

Et si le client est déjà inscrit dans une formation :



# Conclusion

Durant le développement de l’application, on n’a pas trouvé des grandes difficultés grâce aux cours et TP de l’UV qui sont bien détaillés et expliqués. Sauf au niveau de la table de jointure (qui était optionnelle). Mais après des recherches sur Internet et sur les forums du Framework ou de la technologie, on a trouvé la solution convenable à notre problème.

On s’est réparti les taches comme suit :

* Oumaima était chargée de la couche Métier et DAO (Entités / Query)
* Ismail était chargé de la couche de Service et de Présentation (Vue / Contrôleur / Sécurité)