

UFR Mathématique-Informatique

Université de Strasbourg



Projet eLog – État d’avancement

SOMMAIRE

I – Conception.....	2
II – Django.....	3
1 – Framework.....	3
2 – Réalisations.....	3
III – Raspberry Pi.....	6
IV – Conclusion.....	7

I – Conception

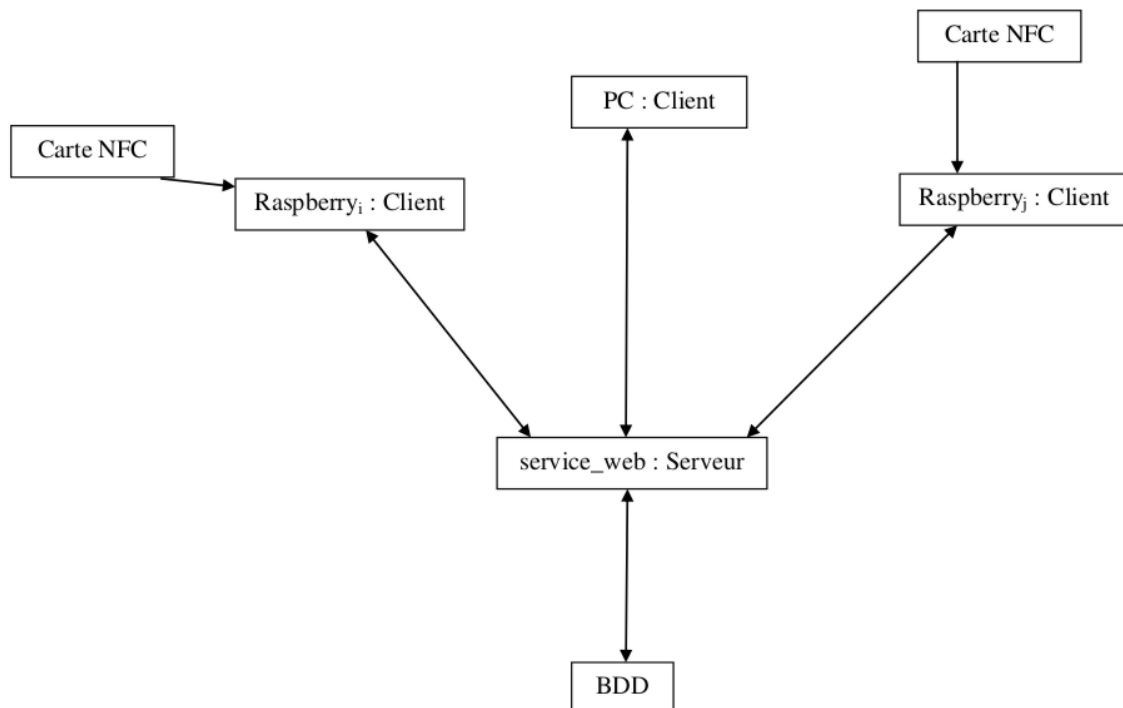
Notre projet consiste à développer une application permettant de se passer de notre fiche de présence papier actuelle. Le but est donc de passer tout cela en version numérique et d'automatiser au maximum. Nous allons voir l'état d'avancement de notre projet, deux mois après son démarrage.

Notre vision des choses, par rapport au cahier des charges, a légèrement évolué lorsque nous avons été confrontés au développement de notre solution.

Le développement de notre application web nous met face à quelques problématiques. Notre solution devra gérer plusieurs terminaux : le Raspberry Pi, le smartphone, le PC, etc. Il faut pour cela se placer en client/serveur où ces terminaux seraient des clients, ça nous permet d'avoir un service transparent et commun pour tous ces terminaux. Ce serveur recevra donc les informations provenant des différents terminaux afin d'exécuter l'opération appropriée. C'est aussi ce serveur qui sera lié à notre base de données. La scolarité aura accès à une interface de gestion pour administrer notre application. Nous devons pour cela développer un Backend.

Concernant notre lecteur de carte NFC, il sera géré par des terminaux Raspberry Pi munis de lecteurs de puces NFC. Ces derniers auront donc un script qui sera constamment actif. À la lecture d'une carte, il enverra une requête HTTP au serveur avec les informations relatives à la carte lue.

Pour mener à bien ce projet, les tâches ont été réparties. Marc et Davy se chargent de la partie serveur web, au moyen du framework de développement Django. Sylvein se charge de la gestion du terminal Raspberry Pi ainsi que son lecteur NFC, fournis par l'université de Strasbourg.



II – Django

1 - Le framework

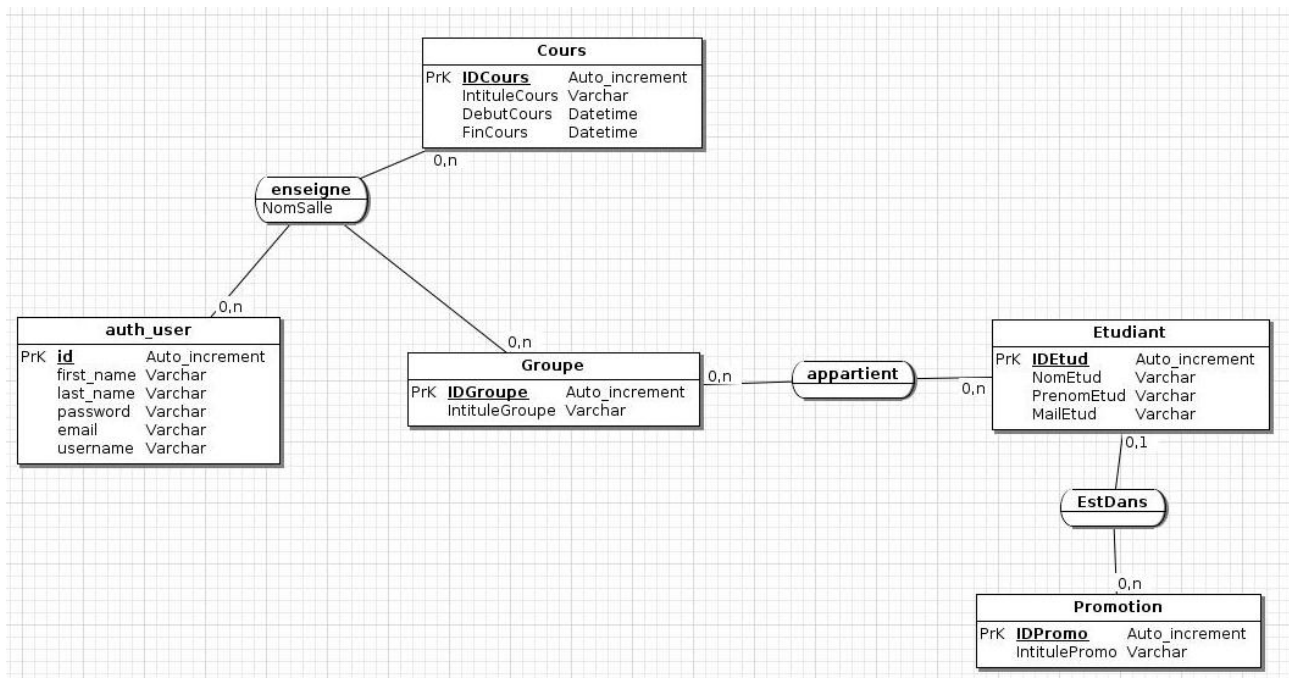
Pour développer notre serveur web, nous nous servons de Django qui est un framework de développement python. Le temps d'appréhension du framework a été assez long. Nous avons eu une longue phase d'apprentissage étant donné que le framework tout comme le langage python ne nous étaient pas familiers. Django nous permet toutefois de créer un serveur web assez simplement. Il intègre de nombreuses API pour gérer les bases de données.

Il implémente également de nombreux outils indispensables à tout développeur web, comme la gestion des formulaires, l'extension de templates, mais aussi des outils qui permettent d'inférer les fichiers modèles depuis une base de données (et vice-versa), par le biais d'un ORM spécifique.

Le code se structure sous une forme particulière, les vues permettant de renvoyer les objets HttpResponse (à l'image d'un contrôleur classique), les templates jouant alors le rôle des vues. Enfin il existe également des fichiers de modèle qui permettront d'accéder facilement à notre base de données.

2 – Réalisations

Pour gérer notre persistance de données, nous avons lié notre serveur avec une base de donnée MySQL. Nous avons créé des données fictives pour tester notre application.



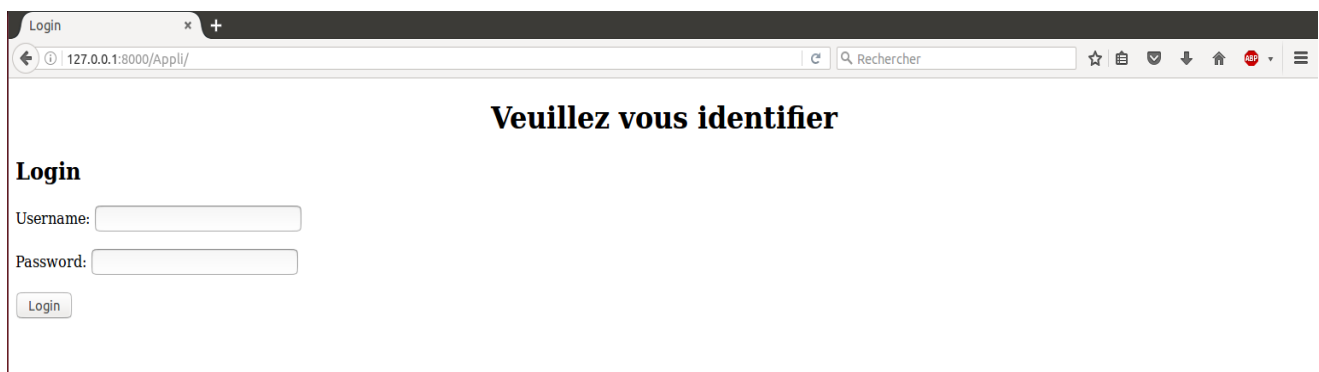
Nous avons donc implanté cette base de données, puis nous avons installé et connecté

Django à celle-ci. La table `auth_user` toutefois est générée automatiquement par Django puisque nous avons décidé d'utiliser le système d'authentification inclus dans le framework. En effet, il nous paraît offrir une sécurité satisfaisante et gère des champs appelés à contenir des informations tout à fait pertinentes pour ce qui concerne notre application, comme le champ `is_superuser` ou la gestion générale des groupes.

Nous en sommes restés à une interface d'administration simple permettant d'ajouter des utilisateurs qui soient simples utilisateurs ou super utilisateurs. Ici, les simples utilisateurs représenteront les professeurs qui seront amenés à se connecter sur le système pour valider les fiches de présence, tandis que les super utilisateurs seront amenés à insérer, modifier ou supprimer directement les différentes données depuis la base MySQL.

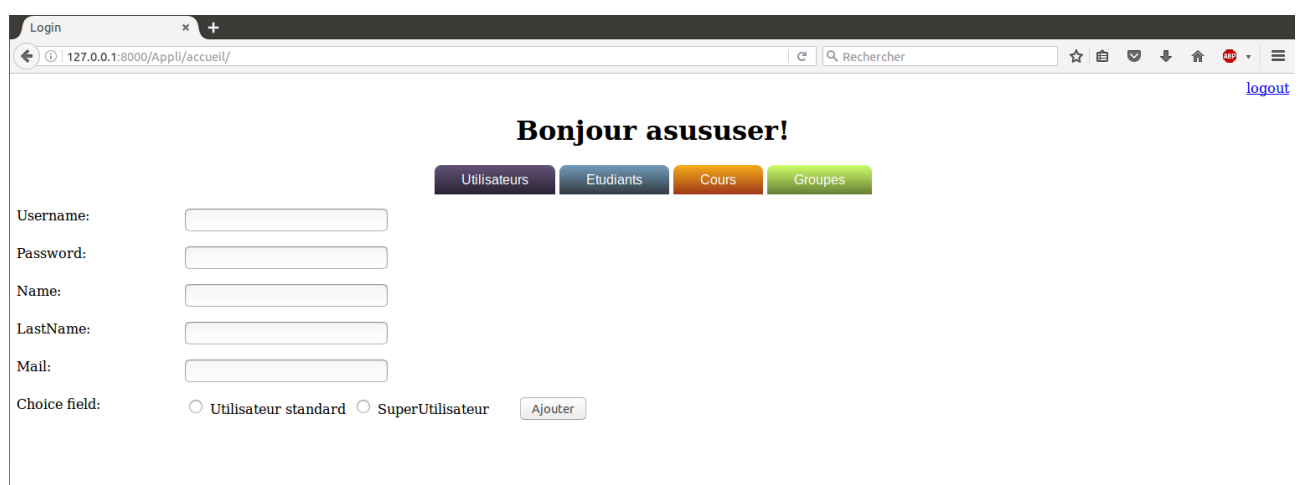
Il nous reste beaucoup à faire comme améliorer le rendu des fenêtres, rajouter les pages web manquantes et finalement installer sur la machine virtuelle qui nous sera proposée par Marc STREB cette application web.

Voici un aperçu de la page de login :



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/Appli/`. The page title is "Login". The main heading is "Veuillez vous identifier". Below this, there is a "Login" section with a "Username:" label and a text input field, a "Password:" label and a text input field, and a "Login" button.

Tout comme la page de création d'un nouvel utilisateur pour tout super-utilisateur s'appelant `asususer` :



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/Appli/accueil/`. The page title is "Login". The main heading is "Bonjour asususer!". Below this, there are four buttons: "Utilisateurs", "Etudiants", "Cours", and "Groupes". Below these buttons, there are five labels and text input fields: "Username:", "Password:", "Name:", "LastName:", and "Mail:". At the bottom, there is a "Choice field:" with two radio buttons: "Utilisateur standard" and "SuperUtilisateur", and an "Ajouter" button.

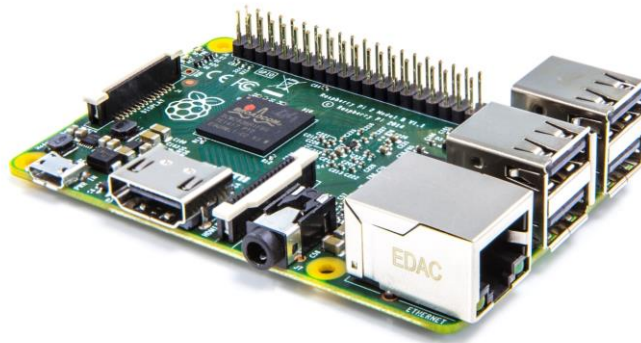
Nous avons également créé une page permettant d'afficher la fiche présumptive des étudiants lors d'un cours. Pour cet affichage, nous avons utilisé une fonctionnalité propre à Django. Il se charge d'exécuter la requête SQL et de gérer les jointures à notre place.

```
liste_etu = Etudiant.objects.filter(idgroupe__enseigne__idutil=user.id,  
idgroupe__enseigne__idcours=cours.idcours)
```

Après exécution de cette commande, liste_etu contiendra la liste des étudiants supposés être présent au cours. Pour effectuer ce filtre, on donne l'identifiant du cours concerné, ainsi que l'identifiant de son professeur. Cette liste d'étudiants "liste_etu" est ensuite passée en paramètre de notre requête HTTP. Son contenu est affiché au sein du template "fiche.html" qui se charge d'effectuer l'affichage. Ce template est sous la forme d'un formulaire que le professeur devra valider. Il doit juste sélectionner les étudiants présents ou non à son cours. L'affichage et le rendu de nos templates n'ont pas du tout été travaillés pour le moment. Par la suite, nous souhaiterons garder cette fiche présumptive validée dans notre base de données pour qu'elle soit consultable depuis la scolarité.

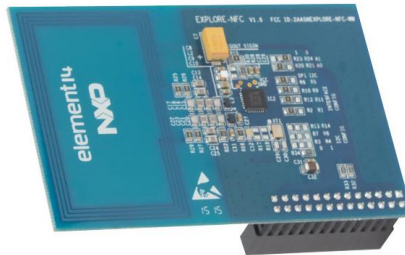
III - Raspberry Pi

En effet, afin de distinguer les étudiants entre eux, nous allons recueillir des informations uniques sur ces derniers. Un moyen sécurisé pour y parvenir est de recueillir les informations discriminantes de leur carte étudiant. Cette carte possède une puce NFC qu'on a alors besoin de lire. Pour se faire, l'utilisation d'un Raspberry Pi a été retenue, il s'agit du Raspberry Pi 2, nous avons avec ce terminal un lecteur de puce NFC.



Raspberry Pi 2 modèle B

Le Raspberry Pi dispose d'un lecteur de carte micro-SD, d'un CPU quad-core ARM Cortex-A7 cadencé à 900MHz, de 1GB de RAM, 4 ports USB, d'un connecteur GPIO à 40 broches (où l'on branchera le lecteur de puce NFC) et quelques ports d'entrées/sorties tels que le port Ethernet, une sortie HDMI, etc. Il est livré avec une carte micro-SD de 64GB.



EXPLORE-NFC-WW, lecteur de puce NFC compatible avec Raspberry Pi

Pour administrer le Raspberry Pi, l'OS Raspbian y sera installé. Le script client qui tournera sur le Raspberry Pi (qui se chargera d'envoyer des informations des cartes lues au serveur) sera écrit en java, car c'est un langage particulièrement accessible. De plus, l'OS Raspbian contient la plateforme de développement Java SE donc rend possible l'exécution de programmes JAVA.

IV – Conclusion

Pour conclure nous ajouterons que nous avons opté pour une architecture qui permettrait à terme de connecter plusieurs Raspberry Pi à notre serveur web. La principale question qui demeure consiste à découvrir une façon de réceptionner les requêtes HTTP que nous enverra le Raspberry Pi de sorte à maintenir une liste des étudiants ayant badgé au courant de la journée au sein de notre application web. De plus, nous avons rencontré des difficultés à nous mettre d'accord sur certains aspects du projet, s'entendre et s'organiser à plusieurs dans un groupe n'étant pas une chose facile. Cela a raccourci le temps qui nous était déjà compté et nous a donc mis en retard par rapport au délai que constituait ce premier rapport.