

Projet WEB II - 2022-2023

HELBManager



POPADIUC CLAUDIU

Monsieur Riggio, Jonathan

Table des matières

Introduction	3
Description des technologies utilisées	4
Fonctionnalités de base	5
Fonctionnalités supplémentaires	15
Analyse.....	17
Limitations et développement futur	18
Conclusion	19
Source	20

Introduction

Dans le cadre du cours de WEB Q III, nous avons été demandés de réaliser individuellement une application web permettant à des utilisateurs de faire de la gestion de projets via un système simplifié de planification de tâches. Pour avoir une certaine base fonctionnelle, M. Riggio nous a proposé de regarder les vidéos de M. Corey Shafer (Source n°1), c'est ce que j'ai fait. J'ai alors codé moi-même la base fonctionnelle.

Dans ce rapport, je vais commencer par expliquer les technologies utilisées, puis les fonctionnalités qui ont été implémentées et leur aspect technique. Je continuerai en faisant la même chose pour les fonctionnalités supplémentaires. Il y aura ensuite l'analyse de mon projet présentée avec l'aide d'un diagramme. Les limitations et le développement futur du projet suivront, et je finirai par une conclusion.

Une page source sera ajoutée à la fin de ce rapport, et j'indiquerai au fur et à mesure dans mon rapport quand les sources ont été utilisées.

Description des technologies utilisées

Pour ce projet, j'ai utilisé le framework Django, qui est un cadre de développement web en Python de haut niveau. Je l'ai utilisé car c'est ce qui a été imposé par mon client. J'ai également utilisé Bootstrap, car il était présent dans les vidéos de la base fonctionnelle. Enfin, j'ai utilisé du JavaScript car j'en avais besoin pour réaliser principalement le drag & drop, ainsi que d'autres détails.

Voici leur description :

Django :

Django est un Framework de développement web en Python qui facilite la création de sites web. Il offre plusieurs outils pour les développeurs, comme la gestion de bases de données, l'authentification de l'utilisateur et la gestion de formulaires.

Django utilise le modèle MVC et met l'accent sur la séparation des tâches, ce qui signifie que chaque composant d'une application ne s'occupe que d'une seule chose. Cela rend le code plus facile à maintenir et à étendre. Django est connu pour être sécurisé et rapide, ce qui en fait un choix populaire pour les développeurs de sites web professionnels.

JavaScript :

JavaScript est un langage de programmation de scripts principalement utilisé dans les navigateurs web pour rendre les pages web interactives. Il peut être utilisé pour ajouter des effets dynamiques sur les pages, comme des animations, des formulaires interactifs et des cartes. JavaScript peut également être utilisé pour construire des applications côté client, des jeux et des applications mobiles.

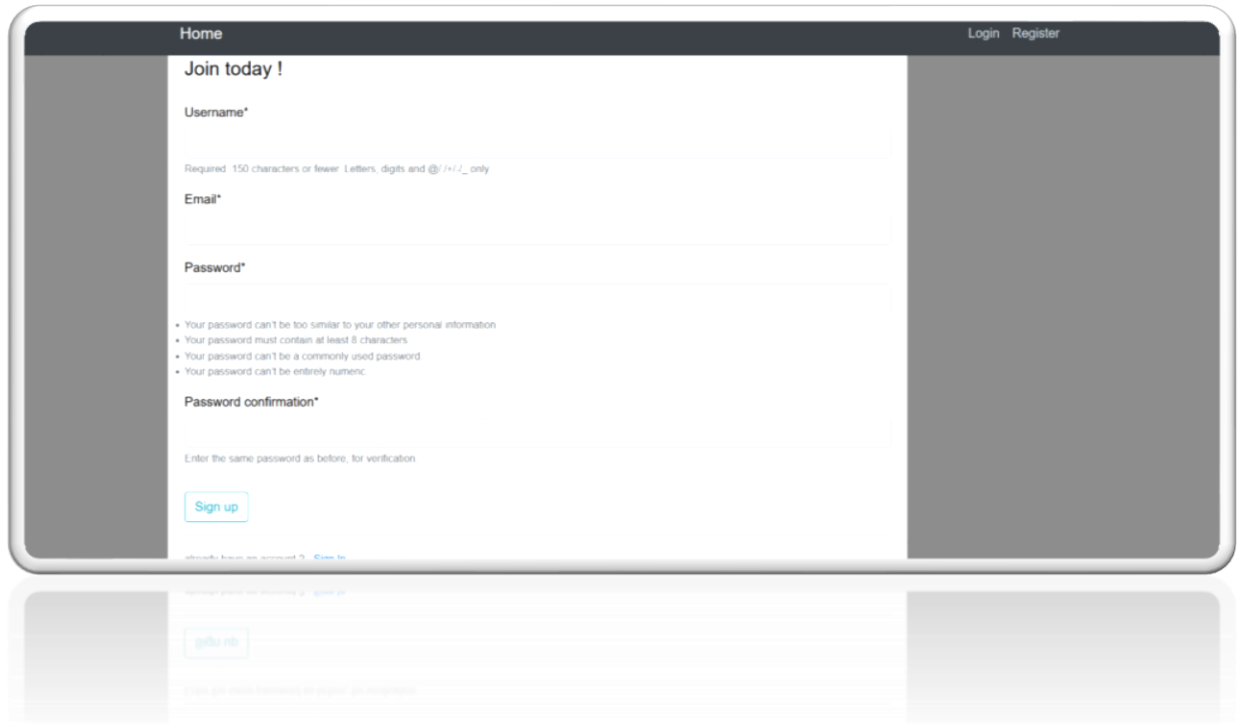
Bootstrap :

Bootstrap est une collection de styles prédéfinis pour la création de sites web et d'applications. C'est un Framework qui vous permet de mettre rapidement en place un site web ou une application en utilisant des styles de base et des composants prédéfinis qui sont prêts à l'emploi.

Fonctionnalités de base

- Un système d'inscription/connexion sur la plateforme.

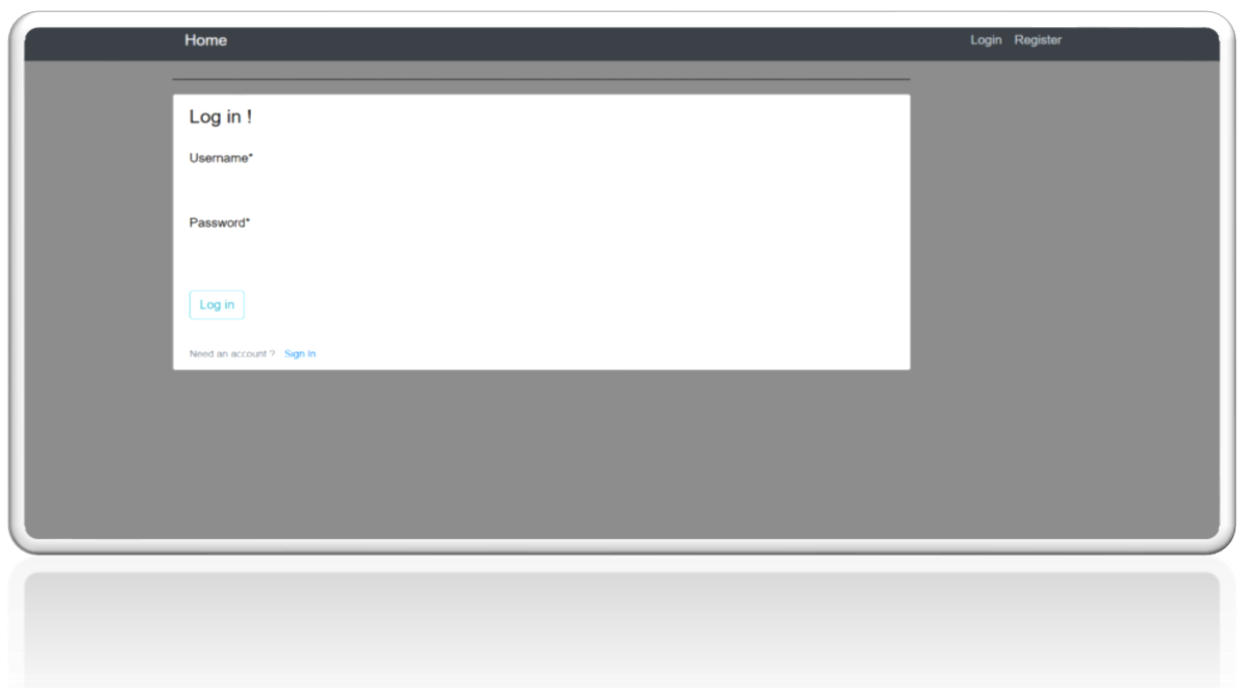
Inscription :



The registration form is displayed on a web page with a dark header. The header contains the text "Home" on the left and "Login Register" on the right. The main content area is titled "Join today !" and contains the following fields and text:

- Username***: A text input field.
- Required: 150 characters or fewer. Letters, digits and @/+/./_ only
- Email***: A text input field.
- Password***: A text input field.
- Your password can't be too similar to your other personal information
 - Your password must contain at least 8 characters
 - Your password can't be a commonly used password
 - Your password can't be entirely numeric
- Password confirmation***: A text input field.
- Enter the same password as before, for verification
- Sign up**: A blue button.
- Already have an account? [Sign in](#)

Connexion :



The login form is displayed on a web page with a dark header. The header contains the text "Home" on the left and "Login Register" on the right. The main content area is titled "Log in !" and contains the following fields and text:

- Username***: A text input field.
- Password***: A text input field.
- Log in**: A blue button.
- Need an account? [Sign in](#)

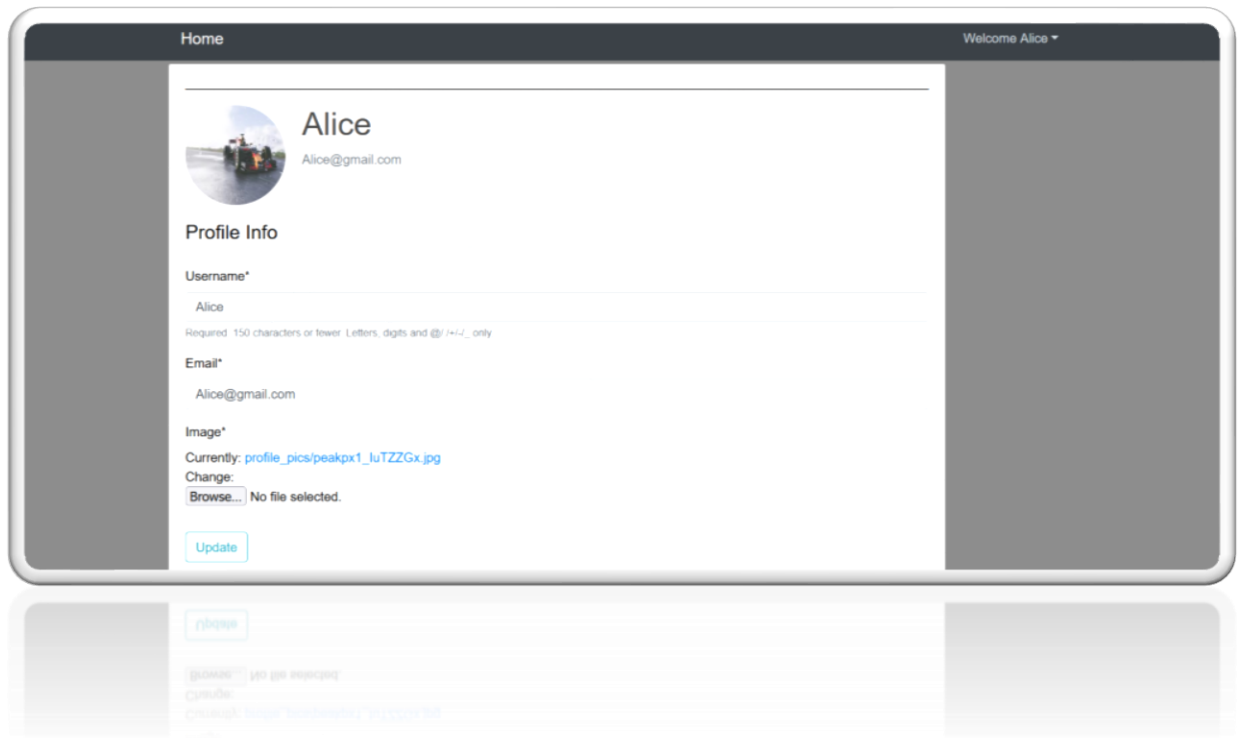
Pour que les utilisateurs puissent enregistrer leur passage sur la plateforme, un système d'inscription et de connexion a été mis en place. Dans le fichier "forms.py", il y a des champs tels que "username", "email", "password1" et "password2" afin que l'utilisateur puisse entrer ces informations de connexion, qu'il pourra modifier plus tard dans son profil s'il le souhaite.

Dans le fichier "views.py", j'ai ajouté une fonction "register" qui, si le formulaire mentionné ci-dessus est correct, enregistre les informations et redirige l'utilisateur vers la page de connexion. Tout est enregistré dans la base de données de Django, et l'administrateur a accès à toutes ces informations.

Dans le fichier "Base.html", il y a un bouton pour accéder aux pages de connexion, d'inscription et de déconnexion. Il y a aussi un fichier "login.html" qui donne à l'utilisateur une interface pour se connecter s'il n'est pas connecté, et un autre bouton qui lui permet de se déconnecter s'il est déjà connecté. Il y a également un fichier "register.html" qui donne une belle vue sur l'inscription de l'utilisateur et qui utilise le "formlcrispy" pour donner à l'utilisateur la possibilité de s'inscrire grâce aux 4 champs du fichier "forms.py".

J'ai pu réaliser tout cela grâce aux vidéos de M. Corey Shafer (Source 1). Je n'ai pas vraiment eu de problème, sauf qu'à un moment donné, un mauvais recopiage du code a été mis en place et l'inscription ne fonctionnait pas, mais tout a pu être réglé.

- Un système de profil pour les utilisateurs inscrits.

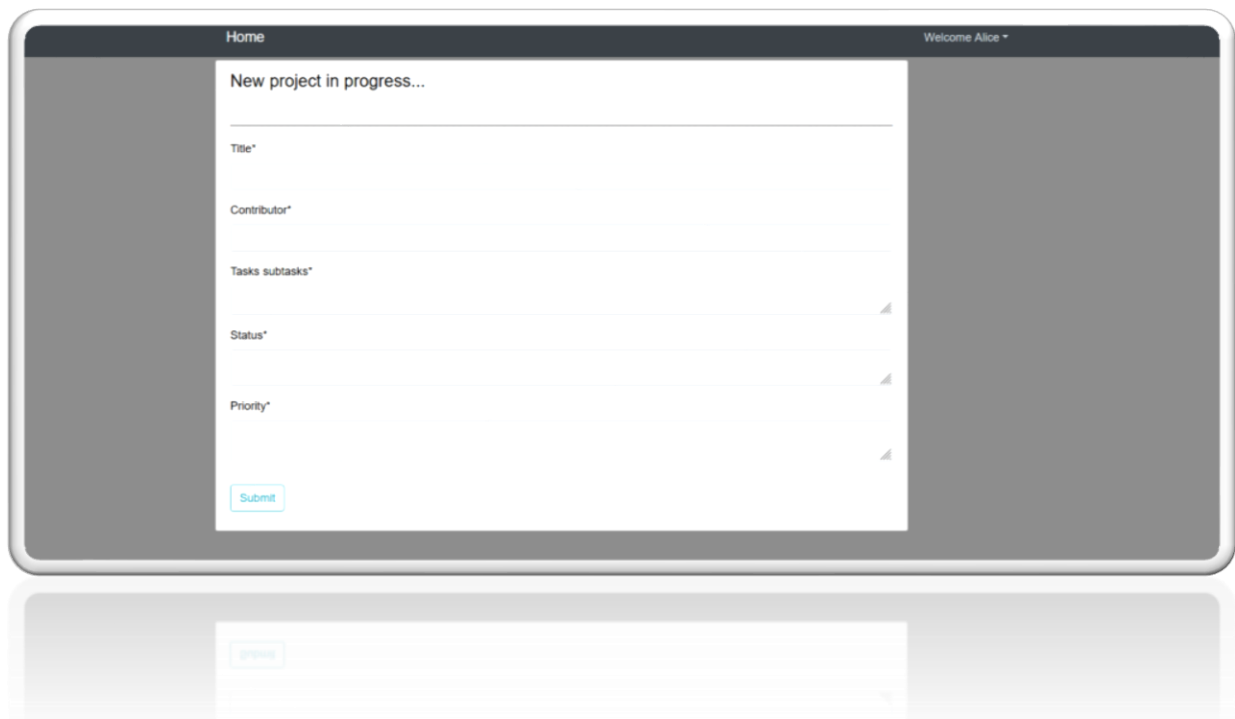
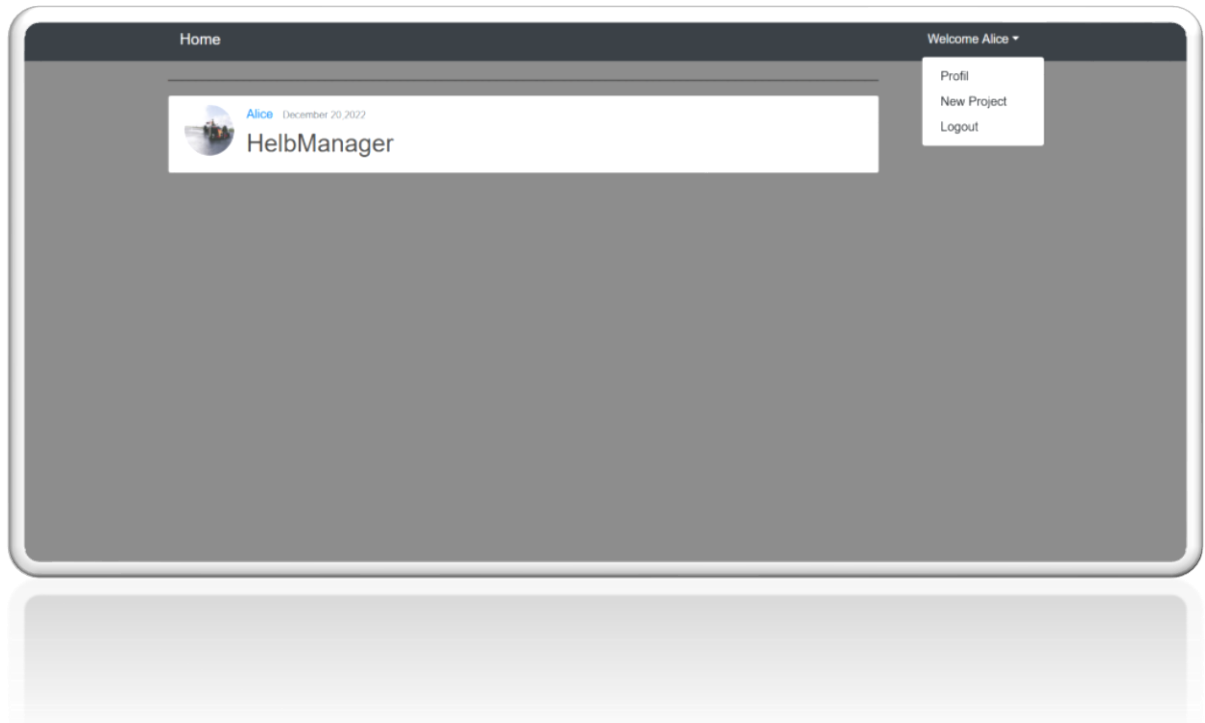


Pour que les utilisateurs puissent visualiser et modifier leurs données, comme leur email et leur nom, un système de profil a été mis en place. Il y aura également une photo de profil par défaut que l'utilisateur pourra modifier s'il le souhaite.

Dans le fichier "Base.html", comme pour l'inscription, il y aura un bouton qui permettra d'accéder au profil de l'utilisateur s'il est connecté à la plateforme. Une page "profile.html" est créée avec les données de l'utilisateur et les formulaires "u_formIcrispy" et "p_formIcrispy" sont inclus dans cette page.

Dans le fichier "forms.py", une classe "UserUpdateProfil" permet la modification de l'email et du nom d'utilisateur, et une autre classe "ProfileUpdateForm" permet la modification de l'image de profil. Dans le fichier "models.py", une fonction permet de sauvegarder ces modifications et de mettre toute image à une hauteur et une largeur convenable, quelle que soit l'image importée. Dans le fichier "views.py", une fonction vérifie, comme pour l'inscription, si le formulaire de modification est valide et, s'il l'est, enregistre les modifications dans la base de données.

- Un système de création et de gestion de projets conforme à la description donnée dans cet énoncé (Project, task, subtasks, status).



Pour pouvoir créer un projet, j'ai simplement repris la fonction de création de post qui était déjà implémentée pour cela, j'ai modifié les postes afin que ce soit un projet qui soit créé. Lorsque l'utilisateur connecté voudra créer un projet, il devra simplement appuyer sur le bouton prévu pour cela (New Project).

Une fois que cela sera fait, l'utilisateur arrivera sur une page où il devra ajouter le titre, les contributeurs, les tâches, sous-tâches ainsi que les statuts souhaités pour son projet dans des champs « Fields » ou « Tex Fields ».

(J'expliquerai le système de priorité dans la section prévue pour cela)

Je définis les champs titre, contributeurs, statuts ainsi que les tâches à l'intérieur du « models.py », puis j'ajoute ces champs dans le « views.py », plus exactement dans une fonction « PostCreateViews » qui prendra en compte les champs nécessaires pour la création du post et les affichera. L'utilisateur qui créera le projet deviendra automatiquement le chef du projet (l'auteur du projet). Ceci sera fait encore une fois dans une autre fonction de « PostUpdateViews » afin de pouvoir modifier ou supprimer les éléments du projet, une fois qu'il a été créé. Pour donner vie à cela, les migrations vont être effectuées via le PowerShell.

Si l'utilisateur souhaite ajouter plusieurs contributeurs, tâches, sous-tâches ou statuts, il devra simplement terminer par un « ; » à chaque ajout supplémentaire. Voici un exemple :

1. Rédaction de l'introduction-BOB;
2. Recherche de documentation sur les app de vente de produits-ALICE;
3. Implémentation de l'application mobiles-BOB;
 - 3.1 Analyse des besoins-BOB;
 - 3.2 Conceptions-CESAR;
 - 3.3 Développements-BOB;
4. Tests de l'application sur des utilisateurs réels-CESAR;
5. Rédaction des résultats des tests-CESAR;
6. Rédaction de la conclusions-BOB

Ceci est possible grâce aux codes que j'ai ajoutés dans « models.py », une fonction définie « get_all_tasks », qui prendra tout ce que j'ai écrit dans l'image ci-dessus, à l'aide d'une boucle for il ajoutera dans un tableau toutes les tâches et sous-tâches une par une dès que l'élément « ; » est retrouvé. Quand cela sera fait, il stockera tout cela dans le tableau et dans le « post-detail.html », une simple boucle for parcourra toutes les tâches et les affichera. Le même principe est appliqué pour les statuts ainsi que les contributeurs.

Une fois que tout cela sera effectué correctement, un simple bouton « submit » sera affiché afin de sauvegarder le projet avec la description donnée dans l'énoncé et le projet s'affichera dans le « Home.html » qui est la page sur la première photo ci-dessus.

Faire tout cela a été très compliqué pour moi, cela m'a pris plusieurs semaines. Au début, j'avais tout hard codé et j'avais mis seulement un certain nombre de champs suffisant pour répondre à l'énoncé du projet, mais si Alice voulait supprimer une tâche, elle n'aurait pas pu le faire. Alors j'ai généré ce système de tableau qui m'a demandé énormément de temps, et c'est mon client M. Riggio qui m'a proposé de faire en sorte que dans un seul Text Field il y ait toutes nos tâches. Cette conversation a eu lieu lors de notre dernière itération.

- **Un système d'inscription des utilisateurs à un projet créé par un chef de projet.**

Comme expliqué précédemment, il y aura un champ pour les contributeurs. Le chef de projet pourra lui seul choisir les utilisateurs qu'il souhaite, afin qu'eux seuls puissent accéder au projet. Cela donnera un effet d'inscription automatique à un projet si un utilisateur fait partie des contributeurs du projet en question. Si un utilisateur ne fait pas partie des contributeurs, il n'aura pas accès aux informations du projet et ne pourra pas changer le drag & drop.

Le traitement des contributeurs et de leur affichage sera similaire à celui des tâches expliqué dans la fonctionnalité précédente

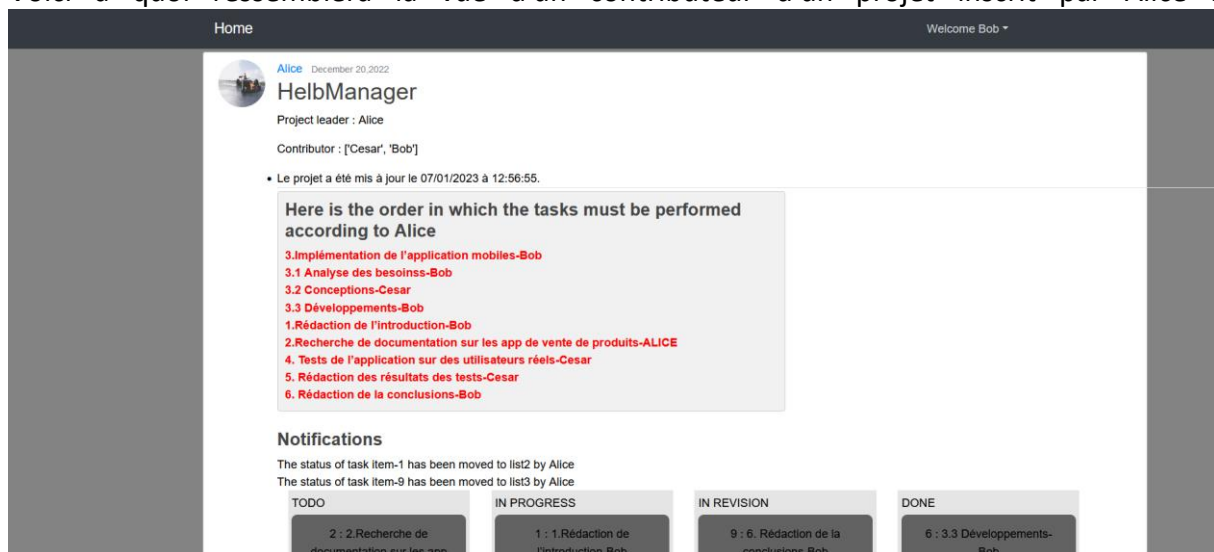
HelbManager

Project leader : Alice

Contributor : ['Cesar', 'Bob']

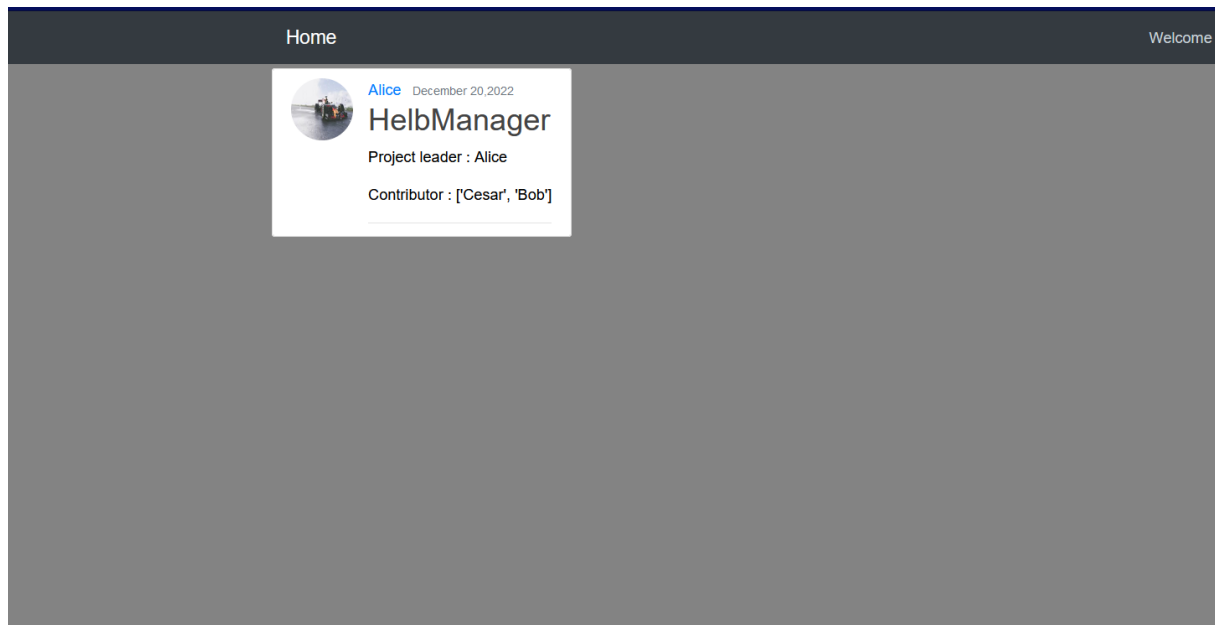
Pour que seuls les contributeurs puissent voir le drag & drop, j'ai ajouté une boucle for dans mon "post-detail.html" qui parcourera le tableau des contributeurs et vérifiera pour chaque contributeur s'il s'agit de l'utilisateur connecté en ce moment même. Si c'est le cas, il affichera toutes les tâches (drag & drop), sinon il n'affichera rien.

Voici à quoi ressemblera la vue d'un contributeur d'un projet inscrit par Alice :



The screenshot shows the HelbManager web application interface. At the top, there's a navigation bar with "Home" and "Welcome Bob". The main content area displays the project details for "HelbManager", including the project leader "Alice" and contributors "Cesar" and "Bob". A message states: "Le projet a été mis à jour le 07/01/2023 à 12:56:55." Below this, a box titled "Here is the order in which the tasks must be performed according to Alice" lists tasks in a specific order: 3. Implémentation de l'application mobiles-Bob, 3.1 Analyse des besoins-Bob, 3.2 Conceptions-Cesar, 3.3 Développements-Bob, 1. Rédaction de l'introduction-Bob, 2. Recherche de documentation sur les app de vente de produits-ALICE, 4. Tests de l'application sur des utilisateurs réels-Cesar, 5. Rédaction des résultats des tests-Cesar, and 6. Rédaction de la conclusions-Bob. Below the task list, there's a "Notifications" section with two messages: "The status of task item-1 has been moved to list2 by Alice" and "The status of task item-9 has been moved to list3 by Alice". At the bottom, there's a task management section with four columns: "TODO", "IN PROGRESS", "IN REVISION", and "DONE". Each column contains a task item with its ID and description: "2 : 2. Recherche de documentation sur les app", "1 : 1. Rédaction de l'introduction-Bob", "9 : 6. Rédaction de la conclusions-Bob", and "6 : 3.3 Développements-Bob".

Vu d'un utilisateur qui n'est pas contributeur :



Cependant, pour le chef de projet, il y aura une autre condition qui vérifiera que si c'est l'utilisateur connecté, alors il affichera en plus la possibilité de modifier les tâches, sous-tâches, contributeurs, statuts ou même de supprimer complètement le projet. L'option "update" permettra d'accéder à la même page que pour la création de projet, mais avec les informations du projet en question afin de pouvoir le modifier.

Voici à quoi cela ressemble :

Alice December 20,2022

[Update](#) [Delete](#)

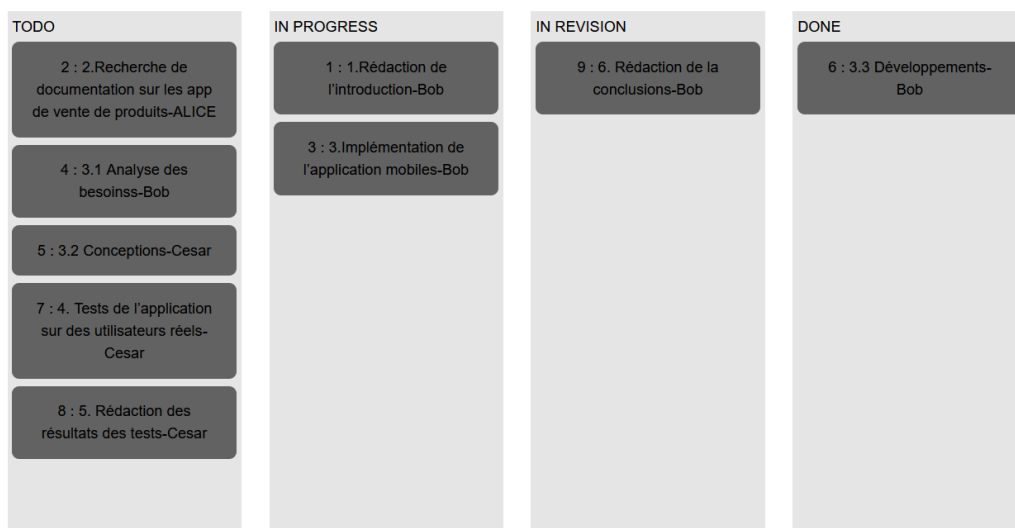
HelbManager

Project leader : Alice

Contributor : ['Cesar', 'Bob']

- Une interface de type « drag & drop » permettant de notifier le statut de tâches et sous tâches du projet.

(Source n°2) Pour effectuer l'interface de drag & drop, je me suis inspiré de cette vidéo YouTube et j'ai ajouté un script.js qui effectue le drag & drop. J'ai ajouté la possibilité de rendre les tâches draggable dans le div qui affichait déjà les tâches. Ces tâches devaient être ajoutées dans un "div app" et ensuite dans un "div lists". Ensuite, dans le CSS, j'ai ajouté les modifications nécessaires pour arranger cela. J'ai fait la même chose pour les statuts, sauf que les statuts ne seront pas draggable, mais seront les cadres dans lesquels on pourra ajouter les tâches. À noter que le chef de projet peut ajouter autant de tâches et de statuts qu'il le souhaite.



(Source 3,4) Malheureusement, le script fourni dans la vidéo n'était pas complet et lorsque j'ai effectué un drag & drop et rafraîchi la page, la page est revenue à son état initial. J'ai eu ce problème pendant une dizaine de jours et cela a été compliqué pour moi de réussir cela, mais j'ai pu comprendre comment faire grâce au stockage local de JavaScript. J'ai donc dû modifier le script afin de pouvoir enregistrer les changements effectués avec le drag & drop, même si je quittais la page.


Finalement, il a fallu ajouter des "ID" à chaque tâche et sous-tâche, pour cela j'ai utilisé un compteur de boucle "forloop" et j'ai dû également modifier une partie du code "script.js" pour le drag & drop. Il y a 2 variables qui sélectionnent tous les éléments de la liste et toutes les listes de la page (list_items, lists).

La fonction "saveDraggedItemLocation" empêche le comportement par défaut de l'événement de se produire, puis récupère l'élément cible de l'événement "drop" et le stocke dans la variable "targetList". Elle vérifie ensuite que l'élément glissé-déposé (draggedItem) et l'élément cible (targetList) sont bien définis avec une condition. Pour finir, la condition enregistre l'emplacement de l'élément déplacé dans le stockage local.

La fonction "restoreDraggedItemLocations" récupère tous les éléments de liste et les stocke dans la variable "list_items". Chaque élément de liste est ajouté avec une boucle "for". La variable "location" récupère l'emplacement enregistré dans le stockage local. Si un emplacement a été enregistré, la variable "targetList" récupère l'élément de liste. Elle vérifie ensuite que l'élément cible est bien une liste. Pour finir, elle ajoute l'élément de liste courant à l'élément de liste cible avec la méthode "append".

Pour assigner les tâches aux contributeurs, Alice ajoutera un "-" suivi du nom du contributeur à la fin de chaque tâche. Si Bob est connecté, il ne pourra déplacer que les tâches qui lui sont assignées en vérifiant si son nom est dessus. Pour les autres tâches, il pourra les voir mais ne pourra pas en changer le statut. Le chef de projet peut modifier toutes les tâches. Cela est effectué dans le "post-detail.html", lorsque j'affiche les tâches, je vérifie s'il y a le nom de l'utilisateur connecté dedans. Si c'est le cas, les tâches suivies de l'utilisateur connecté seront "draggables" et le reste des tâches ne le seront pas. Ainsi, Bob pourra effectuer un changement de statut uniquement pour ses propres tâches. En ce qui concerne le chef de projet, s'il est connecté, il pourra directement modifier le statut de toutes les tâches.

- Des notifications en cas de changement du statut d'une tâche ou sous tâche du projet.



Alice December 20, 2022

HelbManager

Project leader : Alice

Contributor : ['Cesar', 'Bob']

[Delete the project](#) [Update](#)

- The project was updated on 07/01/2023 at 14:08:03.

The status of task item-1 has been moved to list2 by Alice
The status of task item-9 has been moved to list3 by Alice

[Deletes notifications](#)

TODO	IN PROGRESS	IN REVISION	DONE
<p>2 : 2.Recherche de documentation sur les app de vente de produits-ALICE</p> <p>4 : 3.1 Analyse des besoins-Bob</p>	<p>1 : 1.Rédaction de l'introduction-Bob</p> <p>3 : 3.Implémentation de l'application mobiles-Bob</p>	<p>9 : 6. Rédaction de la conclusions-Bob</p>	<p>6 : 3.3 Développements-Bob</p>

(Source n°5) Des notifications ont été ajoutées lorsqu'un changement de statut d'une tâche a eu lieu. Comme vous pouvez le voir ci-dessus, on peut y lire "The status of task item-1 has been moved to list2 by Alice". Nous avons le nom de la personne qui a fait le changement, ainsi que l'ID de la tâche (qui est le premier numéro de la tâche) et l'endroit où la tâche a été déposée (chaque statut correspond à une liste qui a chacune un ID). Pour la première notification, on peut y lire que la rédaction de l'introduction de Bob a changé de statut et est passée à "IN PROGRESS".

Une notification est prévue également lorsque le chef de projet modifie le projet (ajoute, modifie, supprime une tâche). En dessous du bouton "update", on peut lire "The project was updated on 07/07/2023 at 14 :08 :03", ce qui donne la date et l'heure à laquelle le chef de projet a modifié le projet.

Pour créer la notification du drag & drop, dans le "script.js", j'ai ajouté à ma fonction de mouvement de drag & drop "saveDraggedItemLocations" un "InnerHTML", qui effectue un affichage sur une page HTML depuis un script de la valeur donnée. Je reprends donc simplement les ID des tâches, ainsi que celui des statuts, et je les affiche avec une phrase. J'ajoute également le nom de la personne connectée sur le site en ce moment même. De manière similaire à ce que j'ai fait pour les tâches, j'effectue un stockage local pour que les changements sur la page HTML soient enregistrés. J'ajoute également un bouton qui supprime ce stockage local et cela ne sera possible que par le chef de projet.

Pour créer la notification du "project update", j'ai ajouté un script qui faisait la même chose que pour le drag & drop, mais cette fois-ci seulement lorsque le bouton "update" était cliqué. Ainsi, une notification apparaîtra seulement lorsque le chef sera sur la page de l'update du projet.

Fonctionnalités supplémentaires

- Un système de priorité pour les tâches, sous tâches.

Priority*

3.Implémentation de l'application mobiles-Bob;
3.1 Analyse des besoins-Bob;
3.2 Conceptions-Cesar;
3.3 Développements-Bob;
1.Rédaction de l'introduction-Bob;
2.Recherche de documentation sur les app de vente de produits-Alice;
4. Tests de l'application sur des utilisateurs réels-Cesar;
5. Rédaction des résultats des tests-Cesar;
6. Rédaction de la conclusions-Bob

Submit

Comme mentionné précédemment, il y avait un champ qui s'appelait "Priority" lors de la création d'un projet. Dans ce champ, le chef de projet indiquera l'ordre exact dans lequel les tâches devront être exécutées et les collaborateurs devront suivre ses instructions.

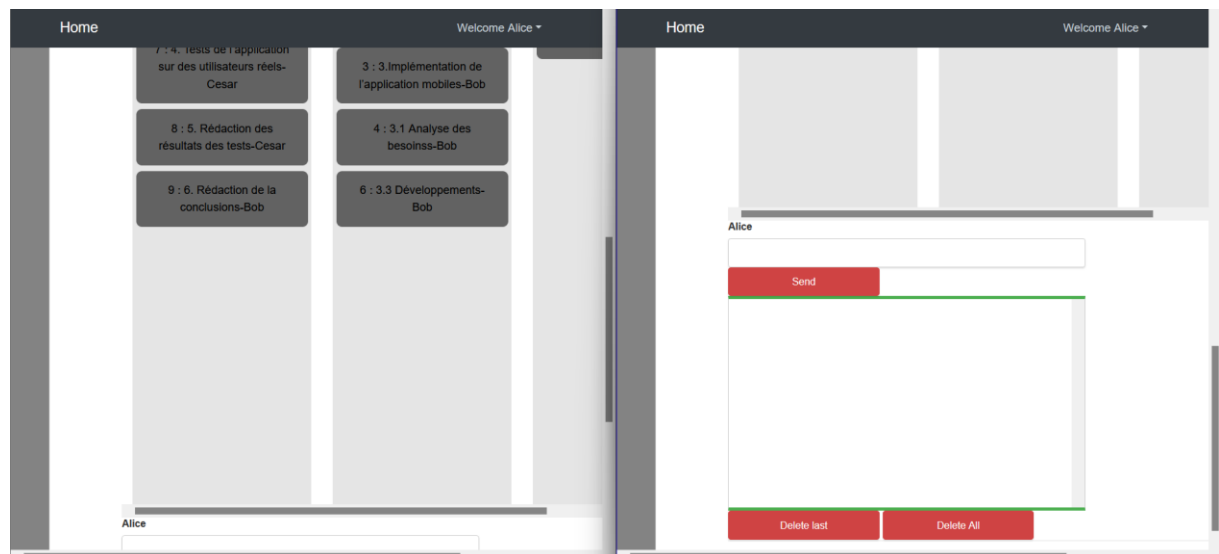
Voici comment les priorités seront affichées dans la page du projet en question :

Here is the order in which the tasks must be performed according to Alice

3.Implémentation de l'application mobiles-Bob
3.1 Analyse des besoins-Bob
3.2 Conceptions-Cesar
3.3 Développements-Bob
1.Rédaction de l'introduction-Bob
2.Recherche de documentation sur les app de vente de produits-Alice
4. Tests de l'application sur des utilisateurs réels-Cesar
5. Rédaction des résultats des tests-Cesar
6. Rédaction de la conclusions-Bob

Le nom du chef de projet sera indiqué en prenant l'auteur du projet. Pour créer cela, j'ai fait la même chose que pour le système de création de tâches, mais pour les priorités. J'ai juste ajouté les priorités dans un cadre avec un affichage simple au lieu de les mettre dans le drag & drop.

- Un système de message entre utilisateurs de projet.



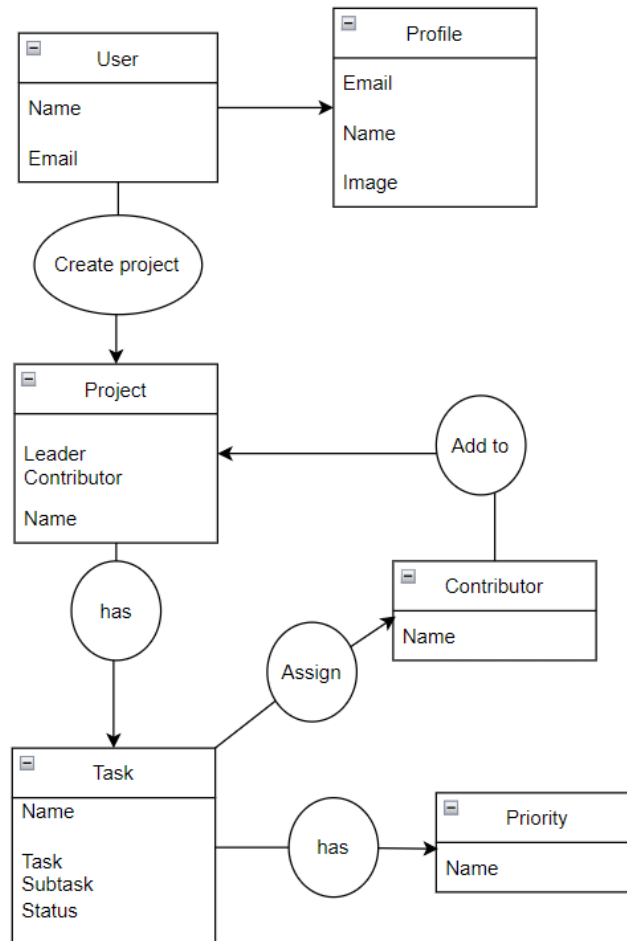
(Source n°6) En dessous du drag & drop, un système de message est implémenté pour le chef de projet et les collaborateurs. Deux boutons seront implémentés et seul le chef de projet pourra les utiliser : l'un permet de supprimer le dernier message et l'autre de supprimer toute la conversation. Pour les boutons, ils s'affichent seulement si l'auteur du projet est sur le projet. Pour le système de chat, j'ai ajouté un container pour l'affichage avec une bande de texte dans l'HTML. Mais tout se passe dans le "script.js". Il y aura, comme pour le drag & drop, un système d'enregistrement des messages envoyés dans le stockage local (comprendre cela m'a énormément aidé pour tout le projet). Et tout simplement, lorsqu'un utilisateur écrit et envoie un message, il est affiché dans la zone prévue pour cela. Le nom de l'utilisateur est affiché avant le message pour savoir qui

Alice : salut cava Bob?
Bob : oui et toi Alice ?
Alice : cava tres bien !

envoie le message.

Pour ce qui est de la suppression des messages, c'est simple : on vide le stockage local en ajoutant à la place de tous les messages une donnée vide et il n'y aura plus rien. Pour supprimer le dernier message, on va simplement prendre dans une fonction le dernier message envoyé et le supprimer de l'affichage.

Analyse



Tous les utilisateurs auront un nom et un email. Ils auront un profil avec leurs données ainsi qu'une photo de profil. Tous les utilisateurs peuvent créer un projet. Une fois qu'ils en créent un, ils deviennent l'auteur du projet (chef de projet). Le projet aura un nom et le chef de projet pourra attribuer à ce projet des collaborateurs.

Chaque projet pourra avoir des tâches et sous-tâches et chacune d'entre elles aura un statut. Chaque tâche et sous-tâche sera assignée à un contributeur. Un système de priorité des tâches est aussi implémenté et chaque tâche aura une certaine priorité que les utilisateurs devront respecter.

Limitations et développement futur

La fonctionnalité de l'interface graphique de la chronologie de l'avancement du projet n'a pas été effectuée. Si j'avais eu plus de temps, je l'aurais faite. Mon site est censé fonctionner très bien selon l'énoncé de mon client, mais malheureusement il y a quelques soucis techniques tels que :

Notification :

J'ai créé ce système de notification comme une sorte d'historique du drag & drop car j'ai essayé de vraiment faire en sorte que les membres du projet reçoivent une notification par email lorsqu'un changement a été effectué, mais après avoir passé beaucoup de temps dessus, j'ai abandonné cette idée et je suis passé à ce type de notification qui a été plus simple à réaliser car j'ai compris le système de stockage local en JavaScript.

L'un des problèmes du site : Les mêmes notifications sont affichées pour tous les projets, même s'il y a un autre projet. Les notifications des autres projets seront affichées dans le projet du scénario de mon client. Cependant, s'il n'y a pas d'autre projet, il n'y a pas ce souci. Un autre petit souci est que lorsque le chef de projet supprime les notifications, il doit faire un drag & drop après cela sinon les notifications ne sont pas supprimées lors du rafraîchissement de la page. Si le drag & drop est effectué, il y aura donc une seule notification qui est celle qui vient d'être faite et, lorsque la page est rafraîchie, les notifications sont supprimées et seul le dernier reste. Si le drag & drop n'a pas été effectué, toutes les anciennes notifications réapparaissent. C'est un petit problème qui normalement ne pose pas de souci technique au site web, de plus, la suppression des notifications n'est pas demandée, c'est un plus pour les utilisateurs du site.

Cependant, les notifications du bouton "update" ne pourront pas être supprimées car je n'ai pas réussi à le faire sans que le stockage local du drag & drop ne soit pas effacé aussi. J'ai donc abandonné cette suppression.

Message :

Le problème des notifications qui se voit dans l'autre projet est le même pour les messages : tous les messages se voient à travers le site, donc même si un autre projet existe, les utilisateurs des autres projets pourront écrire dans le même chat que celui des utilisateurs du projet du scénario.

Assigination : par exemple, lorsque Bob est connecté, il est censé pouvoir modifier le statut de ses tâches, cela fonctionne mais lors de mes tests il s'avère que rarement (une fois sur 15) Bob peut modifier le statut assigné à un autre utilisateur. Ceci est un comportement totalement anormal et ne devrait pas arriver, mais je préfère le signaler dans les limitations.

Si j'avais eu plus de temps, j'aurais réglé ces soucis et j'aurais ajouté plus de fonctionnalités et amélioré l'aspect visuel du site.

Conclusion

Il est vrai que j'ai pu énormément apprendre lors de ce projet, l'utilisation de Django et Python sont des choses très intéressantes et surtout très importantes pour mon développement en tant que programmeur.

Je pense, contrairement au projet de Java, que celui-ci aurait dû être fait en groupe car ce projet a été, à mon avis, beaucoup plus compliqué que celui de Java. Énormément de choses ne fonctionnaient pas, j'ai pris beaucoup de temps à effectuer des recherches et souvent je passais plusieurs heures sur quelque chose qui, à la fin, ne fonctionnait pas. Du coup, même si j'apprenais quelque chose de nouveau, je perdais beaucoup de temps.

Je pense donc que ce projet, avec une autre personne, aurait pu être vraiment beaucoup plus simple et tout aussi intéressant à faire car la complexité aurait aussi été là, mais un appel avec un collègue et qu'on travaille en même temps sur quelque chose jusqu'à ce que le premier trouve une solution et, une fois que le premier a trouvé, on s'explique mutuellement ce qu'on a fait. Je ne me demande pas si c'est même mieux de travailler à deux sur un projet, il y a des chances d'apprendre encore plus que seul.

Quoi qu'il en soit, je suis fière de ce travail. J'ai pu apprendre énormément de JavaScript, alors que je ne l'avais jamais utilisé auparavant, et grâce à cela j'ai pu développer mes performances en tant que programmeur.

Source

M.Corey Shafer:

1: <https://www.youtube.com/watch?v=UmljXZlypDc&list=PL-osiE80TeTtoQCKZ03TU5fNfx2UY6U4p>

Drag & Drop :

2: <https://www.youtube.com/watch?v=tZ45HZAkblc>

Local Storage:

3: <https://openclassrooms.com/fr/courses/7697016-creez-des-pages-web-dynamiques-avec-javascript/7911201-sauvegardez-les-donnees-dans-le-localstorage>

4: <https://grafikart.fr/tutoriels/javascript-local-storage-2077>

Notification:

5: <https://www.youtube.com/watch?v=6WUJdEfftK8>

Message:

6: https://www.youtube.com/watch?v=SF1k_Twr9cg