Projet WEB II - 2022-2023

HELBManager



POPADIUC CLAUDIU

Monsieur Riggio, Jonathan

**Table des matières**

[Introduction 3](#_Toc123758790)

[Description des technologies utilisées 4](#_Toc123758791)

[Fonctionnalités de base 5](#_Toc123758792)

[Fonctionnalités supplémentaires 11](#_Toc123758793)

[Analyse 12](#_Toc123758794)

[Limitations et développement futur 13](#_Toc123758795)

[Conclusion 14](#_Toc123758796)

[Source 15](#_Toc123758797)

# **Introduction**

Dans le cadre du cours de WEB Q III, il nous a été demandé 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 ensuite 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 l’aspect technique. Je continuerais par faire la même chose mais cette fois-ci pour la fonctionnalité supplémentaire. 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 finirais par une conclusion.

# **Description des technologies utilisées**

Pour ce projet j’ai utilisé le Framework « Django » qui est un cadre de développement web Python de haut niveau. Construit par des développeurs expérimentés, Django est gratuit et open source.

Voici 6 motivations d’utilisé Django :

1. Facilité d’utilisation : Django est conçu pour être simple à utiliser et à apprendre, avec une structure claire et une documentation détaillée.
2. Stable, fiable : Django a été utilisé pour développer de nombreuses applications de grande envergure. Il est bien établi et stable, ce qui signifie que vous pouvez vous fier à lui pour exécuter votre application de manière fiable.
3. Sécurité : Django prend la sécurité très au sérieux et inclut de nombreuses mesures de sécurité intégrées pour protéger votre application contre les attaques courantes.
4. Évolutif : Django est conçu pour être facilement évolutif et peut prendre en charge des applications de toutes tailles. Il vous permet de développer votre application étape par étape et de l'élargir au fur et à mesure que vos besoins évoluent.
5. Grande communauté : Django a une communauté active qui contribue à son développement et à sa documentation. Cela signifie que vous avez accès à une grande quantité de ressources et de soutien en ligne.
6. Django inclut également une interface d'administration intégrée qui vous permet de gérer vos données et de surveiller l'état de votre application.

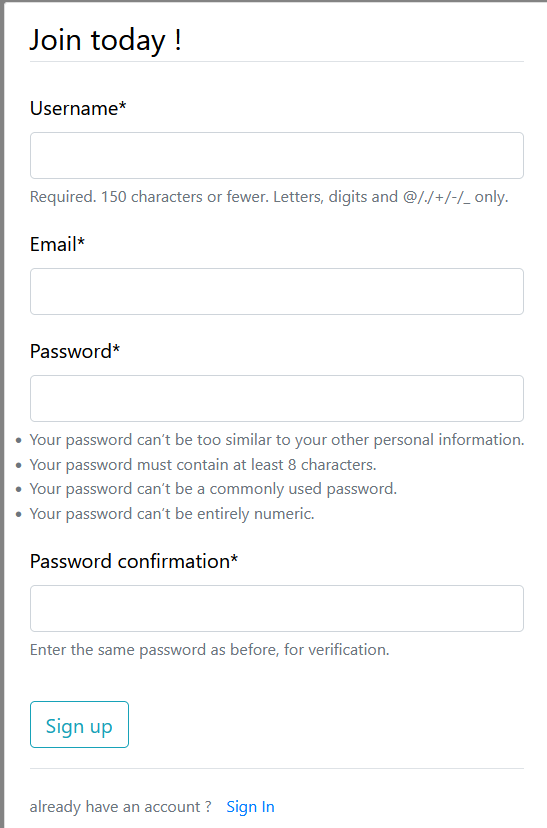
Django est un choix populaire pour le développement web car il vous permet de développer et déployer rapidement des applications fiables et évolutives. Il est utilisé par de nombreuses grandes entreprises, y compris Instagram et Pinterest, et convient parfaitement pour la création de site web.

# **Fonctionnalités de base**

**Mon site web fourni les fonctionnalités suivantes :**

* **Un système d’inscription/connexion sur la plateforme.**

**Inscription :**

****

**Connexion :**

**Une image contenant texte

Description générée automatiquement**

Pour que les utilisateurs puissent enregistrer leur passage sur la plateforme, un système d'inscription et une connexion ont 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 "formIcrispy" 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. Je n'ai pas vraiment eu de problème, si ce n'est 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.**

**Une image contenant texte

Description générée automatiquement**

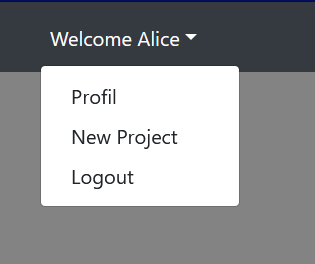
Pour que les utilisateurs puissent visualiser et modifier leurs données, comme leur email et leur nom, un système de profil a été mis en place. Il y aura également une photo de profil par défaut que l'utilisateur pourra modifier s'il le souhaite.

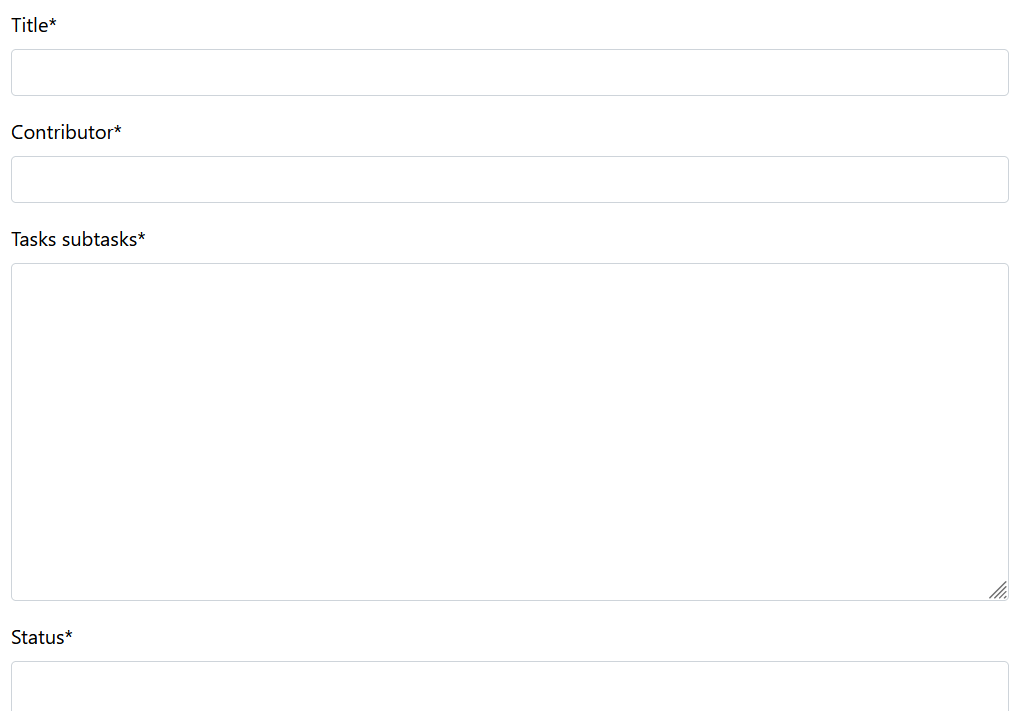
Dans le fichier "Base.html", comme pour l'inscription, il y aura un bouton qui permettra d'accéder au profil de l'utilisateur s'il est connecté à la plateforme. Une page "profile.html" est créée avec les données de l'utilisateur et les formulaires "u\_formIcrispy" et "p\_formIcrispy" sont inclus dans 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 crée un projet, j’ai simplement repris la fonction de création de post qui étais déjà implémenter pour la création de post, j’ai modifié cela afin que ce soit un projet qui soit créé. Lorsque l’utilisateur connecte voudra créer un projet, il devra simplement appuyer sur le bouton prevu pour cela.



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

Je défini les champs titre, contributeurs, statuts ainsi que les taches à l’interieur du « models.py » ensuite 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 effectué via le PowerShell.

Si l’utilisateurs souhaite ajouter plusieurs contributeurs, taches, sous taches ou statuts, il devra simplement terminer par un « ; » à chaque ajout supplémentaire. Voici un exemple :

**Une image contenant texte

Description générée automatiquement**

Ceci est possible grace aux codes que j’ai ajouté dans « models.py », une fonction def « get\_all\_tasks », qui prendra tout ce que j’ai écrit dans l’image ci-dessus, à l’aide d’un for il ajoutera dans un tableau toutes les taches et sous tache une par une dès que l’élément « ; » est retrouve, quand cela sera fait, il stockera tout cela dans le tableau et dans le « post-detail.html », un simple for parcourra toutes les taches 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 »



Faire tout cela a été très compliqué pour moi, cela m’a pris plusieurs semaines, je n’arrivais à faire en sorte de crée un nombre de tache définit par l’utilisateur seulement, a la base j’avais tout hard coder et j’avais mi seulement un certain nombre de champs suffisant pour répondre à l’énonce du projet, mais si Alice voudrait supprimer une taches elle n’aurait pas pu le faire. Alors générer ce système de tableau 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 on est tous nos taches 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 puisse accéder au projet. Ce qui 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, il ne pourra pas changer le drag & drop.

Pour réalise cela, ce qu’il en est des contributeurs et de son affichage il sera similaire qu’aux tâches explique lors de la fonctionnalité précédente.

Une image contenant texte

Description générée automatiquement

Pour faire en sorte que seul le contributeur puisse voir le drag & drop, j’ai dans mon « post-detail.html » une boucle for qui parcourra le tableau des contributeurs et qui vérifiera pour chaque contributeur si c’est l’utilisateur connecte en ce moment même, si c’est le cas il affichera toutes les taches (drag & drop) sinon il n’affichera tout simplement rien.

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 taches, sous taches, contributeur, statuts ou même de supprimer le projet complètement. Ici « l’update » permettra d’accéder a la même page que pour la création de projet mais avec les informations du projet en question afin de pouvoir le modifier.

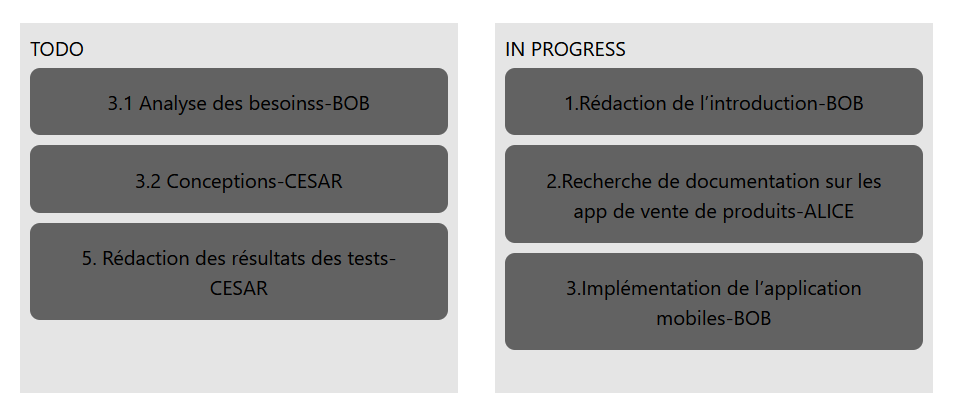
Une image contenant texte

Description générée automatiquement

* **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, j’ai pu donc ajouter un script.js qui effectue le drag & drop.

J’ajoute la possibilité d’être draggable, cela dans le div qui affichait déjà les taches. Et ces taches en question je dois les ajouter dans un « div app » et ensuite dans un « div lists ». Ensuite dans le css j’ajoute les modifications nécessaires afin d’arranger cela. Je fais ensuite la même chose pour les statuts sauf que les statuts ne seront pas draggable mais seront les cadres dans lequel on pourra ajouter les taches.



Malheureusement le script fourni de la vidéo n’était pas complet, lorsque j’effectuais un drag & drop et que je rafraichissais la page, la page revenait à ce qu’elle était au début.

Alors le script a du être modifier afin de pouvoir enregistrer tout les changements même si je quittais la page. J’ai passé 2 semaine sur ce problème, finalement j’ai réussir il fallait ajouter des « ID » à chaque tache et sous taches, pour cela j’ai utilisé un forloop.counter, et il a aussi fallu modifier une partie du « script.js » je vais alors expliquer le code final :

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, ensuite, elle 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 varaible « location » récupère l’enplacement enregistré dans le stockage local. Si un emplacement a été enregistré. La variable « targetList » récupère l'élément de liste. Vérifie ensuite que l'élément cible est bien une liste. Pour finir il ajoute l'élément de liste courant à l'élément de liste cible avec la méthode « append() ».

La première boucle for parcourt tous les éléments de liste et ajoute des événements pour gérer le début et la fin du drag & drop d'un élément de liste. Lorsque l'événement « dragstart » se produit sur un élément de liste, la fonction enregistre cet élément dans la variable « draggedItem » et masque l'élément de la. Lorsque l'événement « dragend » se produit sur un élément de liste, la fonction affiche à nouveau l'élément de liste et réinitialise la variable « draggedItem » à « nul ».

La seconde boucle for parcourt aussi tous les éléments de liste et ajoute les evenements comme pour la premiere boucle. Lorsque l'événement « dragover » se produit sur un élément de liste, la fonction empêche le comportement par défaut de l'événement.

Lorsque l'événement « dragenter » se produit sur un élément de liste, la fonction empêche le comportement par défaut de l'événement.

Lorsque l'événement « drop » se produit sur un élément de liste, la fonction « saveDraggedItemLocation » est appelée. Lorsque l'événement « load » se produit sur la fenêtre (la page est chargée), la fonction « restoreDraggedItemLocations » est appelée pour restaurer l'emplacement de chaque élément de liste en se basant sur les données enregistrées dans le stockage local comme expliqué precedement.

* **Des notifications en cas de changement du statut d’une tâche ou sous tâche du projet.**
* **Une interface permettant de visualiser graphiquement (à l’aide de diagrammes) la chronologie de l’avancement du projet.**

Je n’ai pas fait cette fonctionnalité.

# **Fonctionnalités supplémentaires**

Cette section devra énoncer et

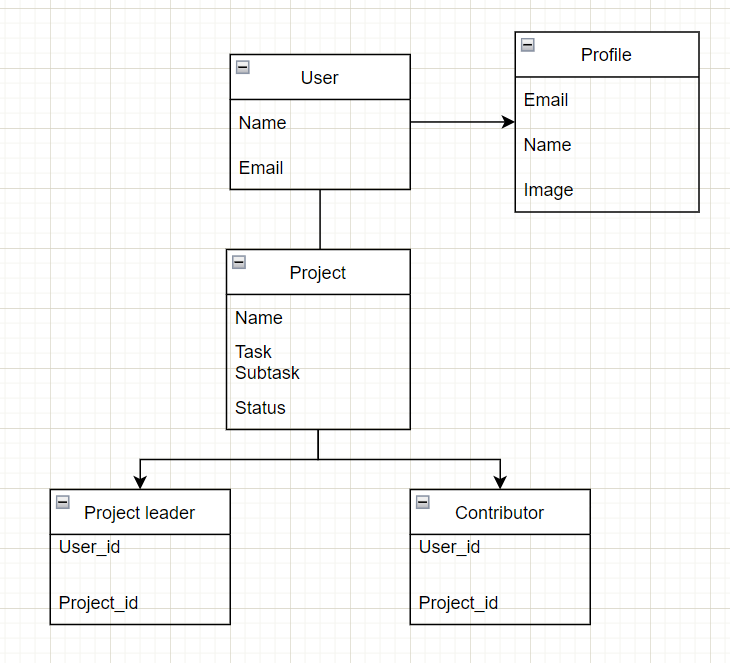
expliquer les fonctionnalités supplémentaires offertes par votre application. Si

il n’y en a pas vous pouvez laisser cette section vide. N’hésitez pas à

également mettre l’accent sur les défis techniques et les solutions apportées

afin de valoriser votre travail.

# **Analyse**



Tous les utilisateurs auront un nom et un email, ils auront un profil, et ils pourront avec ses données ainsi que leur image de profil par défaut.

Ils peuvent tous crée un projet avec un nom, des taches, des sous taches ainsi que des statuts, et en fonction de cela, un utilisateur est soit un chef de projet soit un contributeur, ils peuvent être les deux ou aucun des deux en fonctions de chaque projet crée. Ils peuvent donc participer à plusieurs projets. Chaque chef de projet aura son id enregistré en tant que chef de projet et assignera des contributeurs et enregistra leur id comme tel. Chaque utilisateur aura donc un accès pour leur projet auquel ils font partie.

# **Limitations et développement futur**

Les limites de votre application web,

par exemple : dans quels cas d’utilisation votre application pourrait ne pas

fonctionner comme prévu ? Y a-t-il des aspects techniques du site qui n’ont

pas été traité ? Si vous aviez plus de temps pour le projet, qu’auriez-vous

amélioré ? Plusieurs points de vue sont possibles, il revient au groupe

d’étudiant de choisir les points qu’il considère les plus pertinents pour réaliser

son autocritique.

# **Conclusion**

Il est vrai que j’ai pu énormément appris lors de ce projet, l’utilisation de Django et python, ce sont des choses très intéressante et surtout très importante pour mon développement en tant que programmeur.

Mais 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 chose ne fonctionnait pas, j’ai pris beaucoup de temps a faire des recherches et finalement je passais plus de temps a cherché quelque chose qui fonctionne que coder moi-même. Je pense donc que ce projet avec une autres personne aurait pu être vraiment beaucoup plus simple et tout aussi intéressant à le faire.

# **Source**

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

2 : https://www.youtube.com/watch?v=tZ45HZAkbLc