Ce travail pratique compte pour 25 points de la note finale du cours. Ce travail est à remettre dans Moodle. La date de remise est indiquée dans Moodle.

**Section I : (5 points) Création du projet dans gitHub et Atlas**

Commencez par créer un projet privé dans gitHub que vous devez nommer en suivant ces règles : « WEB2\_FAM\_TP2 » où FAM sont les 3 premières lettres de votre nom de famille. Par exemple, pour moi mon projet se nomme WEB2\_PIL\_TP2.

Ajoutez un fichier README.md pour décrire le projet. Inscrivez un titre et votre nom ensuite vous pouvez copier-coller l’énoncé du travail comme description. Vous devez ajouter un fichier dans votre git pour ignorer les fichiers se trouvant dans les dossiers : « node\_modules ». Ajoutez tous les fichiers nécessaires au bon fonctionnement de votre projet dans git et ensuite publiez le tout dans votre projet gitHub (en faisant un push, vous devriez tester que votre projet est bien publié en faisant un git clone dans un autre dossier sur votre ordinateur). Une fois votre travail terminé vous devrez le publier sur un serveur Web gratuit, je vous donnerai les détails un peu plus tard dans le cours.

Assurez-vous de mettre les commentaires appropriés (qui décrivent vos actions) dans tous vos fichiers Node JS. (il n’est pas nécessaire de mettre des commentaires dans les fichiers vues .ejs).

Dans Moodle vous allez me remettre le présent fichier qui va contenir ces trois informations :

1. Votre répertoire gitHub comme ceci :

git remote add origin https://github.com/OctazGitHub/TP2\_WEB2\_DORR.git

1. Votre token d’accès privé comme ceci:

ghp\_Y8RLc8p7KVPCI5Hg7IHAY65b5t03cU3TZxR9

URL du site publié comme ceci: https://tp2-dorr-web.herokuapp.com/

Il sera important dans votre remise de remettre le présent fichier ET un dossier compressé contenant tout votre projet (soit tous les fichiers nécessaires à l’exécution du projet, excluant le dossier node\_modules). Votre fichier principal pour que je puisse démarrer votre projet doit se nommer « tp2.js » (c’est lui qui sera exécuté lorsque j’utiliserai la commande npm run dev).

Vous devez, ensuite, utilisez la même base de données MongoDB dans Atlas que pour votre TP1 soit « web2\_TP1 ». Ajoutez une collection nommée « usagers2 ». La collection usagers2 va contenir les données de connexion des utilisateurs (c’est le même format que dans le TP1, en fait vous pourriez copier la collection que nous avions utilisée) et vous lui ajouterez deux informations (roles qui est un tableau de chaînes de caractères et fichierImage qui est une chaîne de caractères) :

La collection usagers2 est dans le format suivant :

**[ { \_id : identifiant de l’usager (une chaîne),  
 nom : le nom de l’usager (une chaîne),  
 email : le courriel de l’usager (une chaîne),  
 password : le mot de passe de l’usager en format bcrypt (une chaîne),  
 date : la date que l’usager a été mis dans la BD (format Date),  
 roles : ['normal', 'admin', 'gestion'] (la valeur normal est présente pour tous),  
 fichierImage : une chaîne de caractères représentant un fichier image   
 }, {…} ]**

**Section II : (20 points) Projet nodeJS d’application Web transactionnelle**

Pour cette partie, vous devrez faire un programme en vous servant des fichiers et exemples que j’ai fait pendant le cours. Il s’agit d’écrire un programme qui démarre un serveur Web afin de réaliser une application complète de gestion de notre collection de livres et d’usagers.

Votre serveur Web doit écouter sur le port 8000 et afficher la page d’accueil permettant de se connecter au programme. La page d’accueil redirige le lecteur sur la page d’identification de l’usager (la page login). Une fois identifié, il faut afficher le menu du site (la page d’accueil), qui permet à l’usager soit de gérer des utilisateurs (dans la BD) ou de gérer ou d’afficher la collection de livres ou encore de se déconnecter (logout) (la page d’accueil affiche 3 choix). Aucune page Web ne doit s’afficher (autre que login) tant que l’usager ne s’est pas identifié correctement avec un mot de passe valide. Voici la page d’accueil à reproduire :



Je vous suggère de commencer avec votre travail pratique #1, la page de connexion (login) que nous avons faite pendant les laboratoires (ou dans le TP1) peut très bien être utilisée sans changement.

Dans ce travail nous allons ajouter la notion de rôles et nous allons accepter trois rôles distincts, soit ‘normal’, ‘admin’ et ‘gestion’. Tous les utilisateurs auront au moins le rôle normal qui est automatiquement ajouté lors de la création. Vous devez modifier les pages de gestions des usagers afin d’y inclure cette nouvelle donnée, le rôle (dans des checkbox). De plus, nous allons inclure une image qui peut être téléversée dans le profil de chaque usager. Il faut permettre d’ajouter et de modifier le rôle et l’image de profil en plus des autres informations du TP1.

La gestion des usagers n’est disponible qu’aux utilisateurs possédant le rôle ‘admin’. Toutes les pages de la section gestion des usagers ne doit être accessibles qu’aux utilisateurs possédant ce rôle.

Pour ce qui est du 2e choix du menu « Affichage/Gestion des livres », il permet d’afficher le contenu de la collection livres pour tous les utilisateurs (dans une liste). Les options de gestion des livres ne seront accessibles qu’aux usagers qui possèdent le rôle ‘gestion’. Il faudra donc afficher les livres à tout le monde et permettre de modifier le contenu seulement si l’utilisateur possède le rôle ‘gestion’ (ajout, suppression, modification). Il faut aussi afficher dans une balise img le contenu de url\_image de la collection livres. Vous n’avez pas à valider que le fichier image est disponible sur l’URL (juste l’afficher). Vous devriez l’afficher en miniature sur la liste des livres et en plus gros lors de l’affichage des détails. (il faut permettre de changer l’URL lors de la modification d’un livre).

Comme pour le TP1, l’affichage et la conception des pages de gestion des usagers (de même que pour les livres) est laissé à votre discrétion. Vous devez permettre l’affichage, l’ajout, la modification et la suppression des usagers. J’imagine très bien une liste des usagers avec des boutons pour modifier et supprimer (avec confirmation). Utilisez votre imagination et « fontawesome » pour afficher d’élégants boutons. Les différents contenus seront chargés au fur et à mesure selon les besoins que nous avons.

Naturellement la section route est le point central de votre application Web, car tout se décide dans cette section. Voici les routes que vous devriez avoir :

* GET « /usagers/login » pour la page de connexion (authentification)
* GET de la page d’accueil qui va accepter « / » ou « /index » ou encore « /index.html »
* GET « /usagers/logout » pour déconnecter l’usager
* GET « /usagers/menu » (afin d’afficher une liste des usagers)
* GET « /usagers/editer » (afin d’afficher un usager pour le modifier)
* GET « /usagers/ajouter » (afin d’afficher un usager pour l’ajouter dans la BD)
* GET « /usagers/supprimer » (afin d’afficher un usager pour confirmer la suppression)
* GET de la page des livres « /livres/ » (afin d’afficher la liste des livres)
* GET « /livres/editer » (afin d’afficher un livre pour le modifier)
* GET « /livres/ajouter » (afin d’afficher un livre pour l’ajouter dans la BD)
* GET « /livres/supprimer » (afin d’afficher un livre pour confirmer la suppression)

Aucune de ces pages (sauf login) ne devraient être accessibles si l’utilisateur n’est pas authentifié. Les sections qui se ressemblent d’une page à l’autre devraient se retrouver dans un « layout partiel » (comme par exemple, l’affichage d’un usager ou d’un livre). Vous pouvez ajouter d’autres routes selon vos besoins, mais les routes identifiées ci-dessus sont obligatoires.

Votre application doit être facile à utiliser et vous devez soigner la présentation. Lors de la remise du travail vous devrez faire une présentation de l’exécution de votre travail au professeur.

De façon à ce que je puisse me connecter à votre application vous devez me créer un utilisateur dans votre BD qui se nomme ‘Alain Pilon’ et utilise un email ‘alain@gmail.com’ et mot de passe ‘alain9’ (et possède les privilèges : admin, normal et gestion).