Projet 2

Ce projet compte pour 10% de votre note.

Dans votre répertoire de travail, créez le ***Projet2*** en lançant la commande suivante dans un terminal

$ npx express-generator --no-view Projet2

Ouvrez le répertoire ***Projet2*** qui vous est créé avec Visual Studio Code et observez l'ensemble des fichiers qui vous ont été créés par le générateur ***Express***.

Ouvrez le fichier ***app.js***. Quels sont les modules qui ont été importés par ce fichier dans votre projet?

Effectuez une recherche sur internet et commentez dans votre fichier ***app.js*** ce que font ces modules de façon succincte.

Lancez les commandes suivantes à partir d'un terminal dans le répertoire ***Projet2***

$ npm install

Modifiez le fichier ***package.json*** pour avoir ce qui suit pour l'objet scripts:

```
"scripts": {
    "start": "npx nodemon ./bin/www"
  }
```

Lancez la commande **$ npm start** et testez votre application dans un navigateur avec l'url http://localhost:3000. Vous devriez voir le contenu de la page ***index.html*** affiché.

Récupérez les fichiers de votre ***projet 1*** utilisant fetch et ajoutez-les dans le répertoire public de votre **projet2**. Modifiez les noms des fichiers rapatriés de ***projet1*** vers ***projet2*** pour avoir un contexte plus significatif.

Quand on lance le projet présentement, le fichier par défaut qui est lancé est le fichier ***index.html***. Modifions le fichier app.js pour que le fichier ***projet2.htm*** (que vous avez surement mis dans le répertoire public) soit lancé par défaut. L'appel à la fonction ***express.static()*** peut être ainsi modifié pour ce faire. Faites la modification suivante pour modifier l'instruction d'utilisation de ***express.static()***

```
app.use(express.static(path.join(__dirname, 'public'), {"index": "projet2.htm"}));
```

Le répertoire public vous permet d'exposer la partie ***FrontEnd*** de votre application.

Testez à nouveau votre application dans le navigateur avec l'url ***http://localhost:3000***. Vous devriez voir maintenant affiché la page d'accueil du ***projet 2*** vous invitant à établir la connexion avec le serveur.

Pour ce projet, vous devez toujours utiliser la fonction ***fetch()*** pour communiquer avec le serveur. Vous devriez remplacez tout appel à la fonction ***requeteServeur*** par des appels à la fonction ***fetch()***.

Sauvegardez et zippez votre projet ***sans le répertoire node_modules*** dans un fichier ***Projet2v1.zip***.

Rapatriement du serveur en local

Dans cette étape, vous allez programmer le serveur en local sur votre ordinateur. Il ne doit plus y avoir d'appel au serveur du collège. Tous les appels utilisant la fonction ***fetch()*** doivent être dirigés sur votre serveur web local.

Vous devriez créer un répertoire ***data*** avec pour contenu deux fichiers ***students.json*** et ***succursales.json***. Votre fichier **students.json** aura comme contenu les informations sur les étudiants. Votre fichier ***succursales.json*** aura comme contenu les informations sur les succursales. Un objet étudiant va contenir les mêmes champs que dans le ***labo 5*** en plus d'un champ ***password***. Pour pouvoir se logger dans votre application, un étudiant doit fournir son matricule et son mot de passe comme au ***projet 1***. Vous devez valider ces informations et vérifier sur votre **serveur local** que ces informations sont correctes. Si tel est le cas, l'utilisateur accédera à la fenêtre de gestion des succursales.

Dans le cadre de ce projet, le routage va être fait en utilisant l'objet ***router*** de ***Express*** pour éviter une explosion de gestion de route dans votre fichier ***app.js***.

Ajoutez les lignes suivantes au début de votre fichier ***app.js*** pour la gestion des routes.

```
const succursalesRouter = require('./routes/succursales')
const studentsRouter = require('./routes/students')
```

Ajoutez les lignes suivantes avant la ligne ***module.exports = app***

```
app.use('/succursales', succursalesRouter);  //1
app.use('/students', studentsRouter);         //2
```

Pour accéder aux différentes routes que vous allez définir dans le fichier ***succursales.js***, vous devez les préfixer par ***/succursales***. (1)

Pour accéder aux différentes routes que vous allez définir dans le fichier ***students.js***, vous devez les préfixer par ***/students***. (2)

Créez les deux fichiers ***succursales.js*** et ***students.js*** dans le répertoire ***routes***. Modifiez ces deux fichiers en suivant le même principe que le fichier ***users.js***. Ces fichiers doivent contenir les différentes routes amenant à la manipulation des succursales et des étudiants, l'objet ***router*** jouant le rôle de l'objet app pour accéder aux méthodes ***get, post, put et delete*** dans les fichiers ***succursales.js*** et ***students.js***. Vous aurez à modifier le fichier ***projet2.js*** un peu plus tard pour tenir compte de cette importante modification.

Créez un répertoire ***models*** avec les fichiers ***students_persistence_module.js*** et ***succursales_persistence_module.js*** et modifiez leur contenu de façon appropriée pour la gestion des étudiants et des succursales dans le répertoire ***data***. Vous devez suivre les mêmes principes qu'au laboratoire 5. Le fichier ***students_persistence_module.js*** sera similaire à celui du laboratoire 5. Vous devriez lui ajouter au moins une fonction ***login*** pour vérifier les étudiants qui veulent se logger dans le système. Accommodez votre fichier si nécessaire. Le fichier ***succursales_persistence_module.js*** vous permettra de faire la gestion des succursales. Les objets succursales définis dans le fichier ***succursales_persistence_module.js*** doivent avoir au moins les champs ***ville, budget, matricule***. Le matricule est celui de l'étudiant ayant ajouté la succursale.

Retour sur la partie client (Répertoire public)

Modifiez les fichiers ***projet2.js*** et ***projet2.htm*** de façon appropriée pour la manipulation des succursales et la sécurité du site en conservant les mêmes fonctionnalités que le projet 1. Le site doit fonctionner de la même façon en communiquant avec le serveur Node que vous aurez développé localement.

Sauvegardez votre projet **sans le répertoire node_modules** dans un fichier ***Projet2v2.zip***.

Le projet sera corrigé sur 100 et vous pouvez aller chercher un bonus additionnel de 20 points.

Bonus (20 points)

Modifiez le projet dans une nouvelle version (***Projet2v3.zip***) pour que la production et la consommation des données se fasse avec des objets ***JSON*** complètement.

À remettre

Les fichiers ***Projet2v1.zip*** et ***Projet2v2.zip*** (possiblement ***Projet2v3.zip***) compressé en un sur Léa.

Au plus tard, **Vendredi, le 13 mai 2022 à 15:00**.

Pénalité de 10% par jour de retard.