Cours 420-5C6-LI Applications web II Automne 2025 Cégep Limoilou Département d'informatique Professeur : Martin Simoneau	TP 2 **SPA backend-frontend** **Déploiement React**

Objectifs

- Créer un frontend *SPA* en utilisant les composants *React* et le *JS.*
- Créer un backend en utilisant Spring *framework.*
- Connecter le *backend* avec le *frontend.*
- Déployer *le backend et le frontend dans Docker.*
- Utiliser *Maven* pour automatise le processus de déploiement.

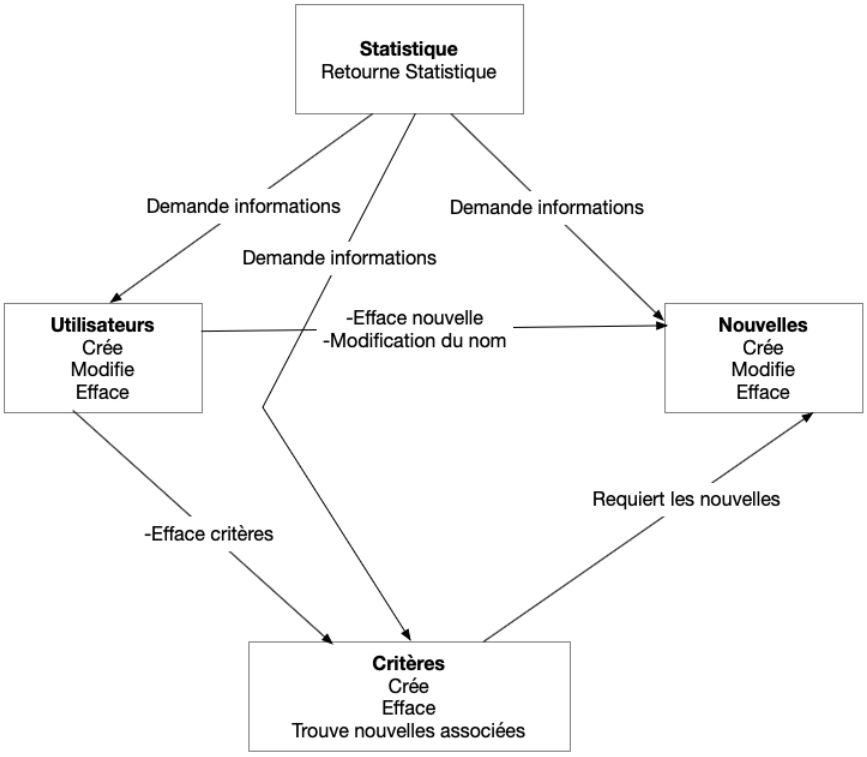
Remise:

- Le travail sera remis sur *Léa* à la date indiquée.

Contexte : **(vous devez vous assurer de bien maîtriser les éléments suivants):**

1) Ce TP est le second de 3 TPs qui impliqueront le même projet.
   a) TP1 frontend *SPA* sans *backend*
   b) TP2 avec *backend* et connexion au *frontend* et déploiement dans docker
   c) TP3 transformation du backend en *microservice avec docker compose.*
2) Le TP2 se fait en équipe de 2 ou 3 étudiants déterminée par l'enseignant.

À faire



1) Programmer 4 modèles (package) backend avec 4 package distincts (nouvelle, critère, utilisateur, statistique):
   a) Chaque modèle/package doit sauvegarder les données en BD (sauf statistique) et logger toutes les transactions qui ont lieu avec des messages significatifs et complets.

b) Chaque package/modèle gère une table indépendante. Idéalement, les tables de la BD sont liées avec des clés étrangères. On n'utilisera pas l'intégrité référentielle de la BD. Ici, on n'en utilisera pas. Les modèles vont devoir communiquer entre eux directement pour gérer les liens. Chaque modèle gère uniquement la table qui lui est associée.
   i) S'il a besoin d'interagir avec les données d'une autre table, il ne peut pas le faire directement. Il doit passer par les services des autres modèles.
   ii) Gardez les modèles indépendants les uns des autres (inversion de dépendance, observateur, événement, injection de dépendances). Utilisez les contrôleurs au besoin.

c) Le **premier** gère les informations concernant les **nouvelles** :
   i) Le titre de la nouvelle doit être unique;
   ii) Le serveur doit valider les données des nouvelles et retourner des messages d'erreur appropriés en cas d'erreur.
   iii) Le serveur doit s'assurer qu'au moins 10 nouvelles de base sont toujours disponibles dans la BD. Si toutes les nouvelles sont retirées, le serveur doit remettre les nouvelles par défaut automatiquement.

d) Le **second** modèle gère les **critères** :
   i) Notez qu'un critère réfère alors à une nouvelle par son ***id*** .
   ii) Le module doit valider les données des critères et retourner des messages d'erreur appropriés en cas d'erreur.
   iii) Le module des critères est responsable de retourner les nouvelles qui correspondent au critère.
   iv) Le module doit pouvoir effacer les critères.
      (1) Le modèle doit créer 5 critères dès le lancement du serveur. Si tous les critères sont détruits, le système doit recréer les utilisateurs de départ automatiquement.

e) Le **troisième** modèle gère les **utilisateurs** (:
   i) Les équipes de **2**
      (1) Les utilisateurs sont fixes et dans un module à part.
      (2) Le module n'a pas de persistance.
   ii) Les équipes de **3**
      (1) Créer des utilisateurs
      (2) Modifier des utilisateurs
      (3) Effacer des utilisateurs. Le module doit alors informer les modules de nouvelles et de critères d'effacer les entrées correspondant à l'utilisateur retiré.
      (4) Le système doit avoir au moins 5 utilisateurs en partant. Si le système requiert des mots de passe, ils doivent figurer dans votre rapport.
      (5) Si tous les utilisateurs sont détruits, le système doit recréer les utilisateurs de départ automatiquement. Il doit alors demander au modèle des critères de refaire les critères par défaut et au modèle des nouvelles de remettre les nouvelles par défaut.

f) Le **quatrième** modèle gère les **statistiques** (pour les équipes de 3) :
      (1) La nouvelle la plus populaire
      (2) La nouvelle la moins populaire
      (3) La nouvelle la plus longue
      (4) La nouvelle la plus courte
      (5) Le nombre de nouvelles
      (6) Le nombre de critères
      (7) … autres statistiques à votre goût
      (8) Le modèle des statistiques n'a pas accès à la persistance. Il effectue ses calculs en interrogeant les autres modules.

g) Programmez la **connexion** entre le *frontend* et les *backends*
   i) Le *frontend* affiche un message particulier s'il ne reçoit aucune nouvelles/critères/utilisateurs du serveur. Il ne doit pas présenter des nouvelles/critères qui ne sont pas dans le serveur.
   ii) Les nouvelles/critères/utilisateurs doivent être chargé(e)s automatiquement au lancement de l'application.

iii) Le frontend doit valider les données avant de les soumettre au serveur.

2) Déploiement Docker:
   a) Le projet au complet doit être déployé via Docker.
      i) Vous pouvez faire un déploiement en war ou en jar.
      ii) Tout le site doit être déployé lorsqu'on fait une installation MAVEN.
      iii) Le frontend doit être accessible directement sur le port **8080** du *localhost*.
      iv) Le projet doit être accessible avec le **débogueur** (port **5005**)
      v) La banque de données utilise un fichier **H2** et elle doit être conservée même si le conteneur est détruit
      vi) Le frontend est déployé automatiquement avec le backend.

Exigences

3) Le travail est fait en équipe sur Github.
4) Les équipes sont faites par le professeur.
5) Vous devez créer un nouveau projet/répertoire git pour le *TP2*.
6) Vous devez maîtriser et assumer tout le code que vous allez remettre. Dans le doute, le professeur pourra vous convoquer en entrevue pour le valider. Si vous ne répondez pas de façon convaincante, la note 0 vous sera attribuée et vous aurez une mention de triche à votre dossier.
7) Les scripts js sont bien différentiés des script jsx des composants React.
8) **Vous devez utiliser Jira/Github et chaque personne doit commiter son propre code. Le crédit sera automatiquement donné à celui qui fait le commit**
9) Chaque commit doit être associé à un *issue* sur Github.
10) Bref rapport contenant :
    a) Décrire ce que chacun à fait.
    b) Les utilisateurs et mot de passe s'il y en a.
    c) Comment utiliser votre site.

Critères d'évaluation

11) Pour chaque critère :
    a) **90+** l'étudiant maîtrise tous les concepts avancés du cours et ne fait pas d'erreurs.
       i) L'équipe travaille de façon continue et maîtrise les outils de gestion de projet (git/github)
       ii) Le travail fait preuve d'une grande créativité et d'une certaine originalité.
       iii) Pas de message d'erreur en console pendant l'exécution.
       iv) Pas de code inutile laissé en commentaires. Pas d'élément d'interface inutile.
       v) Pas de détour dans les algorithmes
       vi) Respect sans faute du modèle MVC (validations et statistiques font partie du modèle).
       vii) Le code est parfaitement découpé et facilement réutilisable.
       viii) L'étudiant démontre une maîtrise de tous les concepts avancés :
          (1) Respect intégral du MVC
          (2) Indépendance des modèles.
          (3) Création des composants.
          (4) Connexion automatique des composants.
          (5) Création automatique de la BD et de son contenu.
          (6) Communication entre composants.
          (7) Le projet est livré et fonctionne sans intervention. Retéléchargez votre projet sur un poste vierge pour vous assurer que tout fonctionne bien.
          (8) Avancement régulier et communications efficaces.
          (9) Respect des échéanciers.
    b) 80+
       i) Très peu de messages en console pendant l'exécution.
       ii) La majorité des validations sont effectuées dans le client et le serveur.
       iii) Les messages d'erreurs sont toujours clairs et complets.
       iv) L'application est toujours stable (surtout lorsque des données sont échangées frontend-backend).
       v) Le code est clairement découpé et facilement réutilisable.

vi) Le travail fait preuve d'une bonne créativité.

c) 70 L'étudiant conçoit une application qui fonctionne correctement, mais qui n'utilise surtout les éléments de base ou maladroitement les éléments plus avancés. Il peut y avoir des bogues mineurs ou des imperfections dans l'interface dans les communications ou dans les messages.
   i) Aucun message d'erreur important en console.
   ii) Le travail est assez générique et convenu.

d) 60+ Les fonctionnalités les plus importantes sont là et elles sont utilisables via le SPA.

FIN