Devoir de contrôle terminal

À rendre au plus tard le samedi 30 mars 2024 à 23h59

Consignes

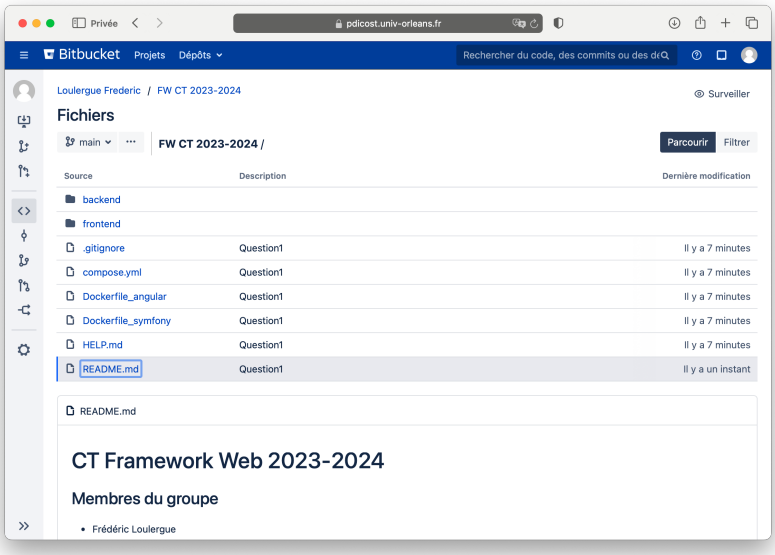
Dans ce devoir, vous serez amenés à mettre directement en application les compétences acquises dans les séances de travaux pratiques. Les dernières questions nécessiteront une recherche documentaire. Le rendu se fera **exclusivement** sur le dépôt GIT du département informatique, le même dépôt devant contenir la partie SYMFONY et la partie ANGULAR.

À chaque question du devoir, vous créerez une version sous la forme d'une étiquette (*tag*). Les étiquettes devront être de la forme `questionN` où `N` est un nombre écrit numériquement.

Pour les noms d'étiquette et tous les noms de répertoires et fichiers, la casse est importante. Vous ajouterez à votre dépôt, un fichier `README.md` au format *markdown*. Ce fichier sera mis à jour pour contenir la liste des commandes utilisées pour répondre à la question traitée. Par souci de portabilité, nous utiliserons SQLite comme système de base de données.

Le non respect des spécifications de noms de fichiers, répertoires et étiquettes, ou de la structure du dépôt GIT conduira à des malus importants pouvant aller jusqu'à 50% de la note.

La structure de votre dépôt (**à la racine**) doit être la suivante :



Vous devez utiliser DOCKER et la configuration spécifique à ce contrôle qui vous est donnée sur Célène. Comme d'habitude, il faut modifier le fichier `compose.yml` pour indiquer vos identifiants d'utilisateur, noms et email **de l'université sous la forme prenom.nom@etu.univ-orleans.fr**. **Attention** il faut pour ce fichier modifier ces données à deux endroits car deux containeurs sont définis.

Attention :

— la contribution de chaque membre du projet sera jugée sur les noms des *commits* dans le dépôt GIT. Si pour une raison légitime (par exemple mise en œuvre du *peer-programming* (programmation en bînôme)) les *commits* ne reflètent pas la répartition effective du travail, celle-ci doit être explicité très clairement dans le fichier `README.md`

— les autres valeurs du fichier `compose.yml` **ne** doivent **pas** être modifiées (en particulier les redirections de ports).

Le fichier `ct2024.zip` contient un projet SYMFONY et un projet ANGULAR à compléter. Attention l'archive a été spécialement créée pour faciliter le dépôt de votre travail avec GIT, elle est donc à utiliser.

Pour démarrer facilement :

1. Cloner le dépôt GIT de votre groupe
2. Dans le répertoire obtenu, décompresser l'archive `ct2024.zip`
3. `git add -A`
4. `git commit -m "Code fourni"`
5. `git push`
6. Vérifier que vous obtenez un affichage similaire à celui de l'image ci-dessus en vous connectant à votre dépôt GIT du département d'informatique (il n'y aura pas de fichier `README.md` et la colonne description indiquera « Code fourni »)

Questions

Question 1. Créer votre fichier `README.md` au format *markdown* contenant les noms des membres du groupes. En répondant à la question 1, n'oubliez pas de taguer votre dépôt avec l'étiquette « `Question1` » et pensez à ajouter dans le `README.md` les commandes ANGULAR utilisées.

Nous considérons un système d'information composé uniquement d'une entité *Film*. Un film est repréenté par un titre, une durée en minutes, une réalisatrice ou un réalisateur, un résumé et une date de sortie.

Question 2. Créer les composants *accueil* et *film* avec pour URL respectives `"/."` et `"/film"`. Ces composants afficheront un simple texte du type `"Bonjour l'accueil"` et `"Super film !"`.

Question 3. Ajouter une barre de navigation à votre application en utilisant MATERIAL DESIGN.

Question 4. Créer une classe *Realisateur* et un service associé *RealisateurService* gérant un tableau de réalisatrices et réalisateurs (avec leurs prénoms, noms et nationalité). Le tableau sera initialisé avec 6 valeurs.

Question 5. Créer une classe *Film* et un service associé *FilmService* gérant un tableau de films. Le tableau sera initialisé avec 12 valeurs.

Question 6. Créer des composants pour afficher une liste de film, pour ajouter ou modifier un film et enfin pour supprimer un film.

Question 7. Mettre à jour le composant de modification d'un film pour que le choix de réalisateur se fasse par un élement déroulant dont le contenu est donné par le service des réalisateurs.

Question 8. Créer un *back-end* en SYMFONY, accessible en tant que service web en utilisant API PLATFORM, pour la gestion des films avec une base de données SQLite.

Question 9. Ajouter un service *FilmWebService* qui accède au back-end SYMFONY pour la gestion d'une base de données de films, en écrivant un client HTTP pour le web service SYMFONY. Même chose pour les réalisateurs.

Question 10. Modifier les composants pour qu'ils fassent appel à *FilmWebService* et *RealisateurWebService* plutôt que *FilmService* et *RealisateurService*.

Question 11. Modifier le composant qui gère la liste de films pour n'afficher les films que d'un seul réalisateur.

Question 12. Mettre en place un système d'authentification. Un utilisateur doit pouvoir créer un compte, se connecter à l'application et se déconnecter. Un utilisateur non-connecté ne doit pas pouvoir supprimer, ajouter, ni modifier de film.

Question 13. Changer les entités et la base de données de telle sorte que l'utilisateur qui a créé un film soit le seul à pouvoir ultérieurement modifier ou supprimer ce film.