Nom : _____

Prénom : _____

Identifiant : _____ Groupe : _____ Enseignant : _____

/20

Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

décembre 2022
WEBG5
ARO - JLC

WEBG5 – Applications d'entreprise

Examen blanc

Partie II - Exercice
Gestion de projet Scrum

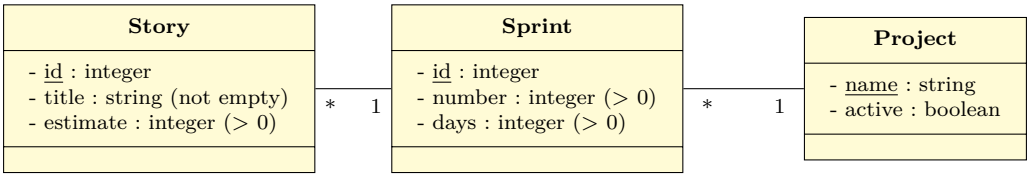
Consignes

1. Téléchargez les sources à compléter pour cette épreuve.
2. La structure des données déterminées par le fichier `data.sql` ne peut être modifiée.
3. Les versions de Spring et de Java présentes dans le `pom.xml` ne peuvent être modifiées.
4. À chaque changement d'étape, vous devez faire un **commit** et un **push** sur le dépôt gitlab créé pour vous. Testez-le pour identifier d'éventuelles erreurs à l'exécution. La cotation tient compte du style de programmation.
5. Durant le développement pensez à gérer les erreurs coté serveur **et** coté client.
6. Vous **devez** respecter l'architecture étudiée au cours.

Présentation

Vous allez coder une application permettant de gérer les histoires de projets gérés via la méthodologie Scrum.

Le schéma des données est le suivant :



Story

Représente une histoire utilisateur : son identifiant, son titre et son nombre de points estimés. Une histoire est liée à un unique sprint.

Version du document
5 décembre 2022

Sprint
: Représente un sprint : son identifiant, son numéro au sein du projet et le nombre de jour de travail qu'il comprend. Les numéros de sprint commencent à 1 pour chaque projet. Un sprint est lié à un unique projet. Un sprint peut contenir plusieurs histoires utilisateurs.

Project
: Représente un projet : son nom et son statut qui précise si il est terminé ou non. Un projet peut contenir plusieurs sprints.

Étapes

1 Persistance (4 points)

Modifiez le code fourni et transformez les classes du package `model` en entités. N'oubliez pas d'implémenter les contraintes associées à ces entités.

Créez une requête `JPQL` qui retourne pour les projets dont le nom commence par les lettres données en paramètre, le nom du projet, le nombre de sprints qu'il contient et le nombre d'histoires qu'il contient. Ces données associées à un projet sont encapsulées dans un `DTO`.

Créez un `derived query` qui retourne l'ensemble des histoires d'un projet dont le nom est passé en paramètre.

> L'absence de commit après chaque étape est considéré comme une erreur.

2 Liste des projets (2 points)

Développez une première page qui propose un tableau affichant la liste des projets. On y trouve 3 colonnes : le nom du projet, le nombre de sprints du projet et la mention *Terminé* si le projet n'est plus actif et *En cours* dans le cas contraire.

Cliquer sur le nom d'un projet permet de consulter une nouvelle page présentant les détails du projet. Le contenu de cette nouvelle page est l'objet de l'étape suivante.

> Attention à ne pas oublier le commit.

3 Détails d'un projet (2 points)

La page de détails d'un projet affiche le titre du projet accompagné d'un tableau contenant la liste des histoires de ce projet. On trouve dans ce tableau 3 colonnes : le titre de l'histoire, le numéro de l'itération associée à l'histoire et le nombre de points estimé de l'histoire. Indiquez à votre utilisateur via un message adéquat si le projet ne contient aucune histoire.

4 Ajouter une histoire (4 points)

Ajoutez à la page de détails d'un projet un bouton qui renvoie vers un formulaire permettant d'ajouter une histoire. Attention une nouvelle histoire ne peut être ajoutée qu'au dernier sprint du projet. Si un projet est clôturé, aucune histoire ne peut y être ajoutée.

Lorsqu'une histoire est ajoutée par le formulaire, l'utilisateur est redirigé vers la page de détails du projet. Cette page affichant la liste des histoires mise à jour.

5 Service Web (2 points)

Développez un service web qui permet d'obtenir la liste des projets dont le nom commence par les lettres données en paramètres. Les données retournées correspondent aux données produites par la requête `JPQL` développée lors de l'étape `Persitance`.

La réponse est au format `JSON`. Si le service rencontre une erreur le **statut de la réponse** est de type `Internal Server Error`.

6 Sécurité (2 points)

Modifiez votre application pour permettre l'ajout d'une histoire uniquement aux utilisateurs connectés avec le rôle `ADMIN`.

N'oubliez pas d'ajouter sur chaque page un bouton de connexion si l'utilisateur n'est pas connecté et un bouton de déconnexion si l'utilisateur est connecté.

Pour commencer cette étape, dé-commentez et modifiez la classe `be.he2b.scrum.security-.SecurityConfig`. Ajoutez également les dépendances nécessaires à votre projet.

Autres questions

Cette épreuve ne contient pas toute les questions possibles. On peut imaginer par exemple la question suivante :

7 Tests (0 point)

Complétez la classe `be.he2b.scrum.web.WebControllerTest` pour valider votre développement via les cas de tests suivants :

- ▷ consulter la page des projets affiche le projet `Othello` ;
- ▷ ajouter une histoire redirige l'utilisateur vers la page de détails du projet contenant la nouvelle histoire ;
- ▷ ajouter une histoire à un projet clôturé affiche un message d'erreur ;