Contrôle continu - TP / mini projet

Le code doit être soumis sous la forme d'un repository git (sur Github, Gitlab ou Bitbucket). Pour les repository privés, merci de bien vouloir donner un accès en lecture respectivement à `tdutrion`, `tdutrion` et `thomas_dutrion`.

Date limite de soumission : dimanche 7 janvier 2017 à 23h59.

Envoyer le lien vers le repository à `tdutrion@myges.fr` avec votre nom, prénom et classe.

Le projet est à faire **individuellement**.

Vous pouvez utiliser le `README.md` pour tout message concernant un éventuel problème d'implementation ou toute autre remarque.

Le projet

Sur une base de Zend Framework 3 et Doctrine (comme vu en cours), vous devez gérer une liste de meetups.

Un meetup possède un titre, une description, une date de début, une date de fin.

La date de fin ne doit pas être antérieure à la date de début (pour les validations dépendantes, les validateurs prennent la `$value` en premier paramètre et `$context` en second).

Le titre doit être proprement limité en nombre de caractères, de même pour la description (afin d'eviter une erreur SQL non user friendly).

Un meetup peut optionnellement avoir des participants, organisateurs, être rattaché à une organisation...

Le site doit permettre de lister, ajouter, mettre à jour et supprimer les meetups, et consulter le détail d'un meetup spécifique.

L'utilisation de docker et docker-compose est grandement appréciée.

Notation

Les fonctionnalités CRUD sont obligatoires, et doivent être réalisées dans les règles de l'art (séparation des classes de formulaire, repository, etc, utilisation de validateurs sur les formulaires). Elles permettront d'obtenir 12/20 si tout est bien fait.

Tout code mort non utilisé sera pénalisé. Votre projet doit uniquement contenir le code requis pour

les fonctionnalités exposées ci-dessus (dans le cas où le projet utilisé en cours est réutilisé notamment).

Les `child_routes` et routes `Segment` devront être utilisées à bon escient.

Les fonctionnalités en plus, notamment l'utilisation de champs/validateurs/filters plus complexes que dans les exemples du cours seront valorisées. De même, l'utilisation du Zend Event Manager et des listeners associés permettra de remonter la note de façon conséquente (si le reste est bien fait).

Vous êtes encouragés à utiliser les bonnes pratiques modernes, notamment pour l'utilisation d'un code style, avec des outils comme `php-cs-fixer` ou `phpcs` / `phpcbf` .

Les dépendances en terme de librairies doivent être organisées correctement dans `composer` (i.e. `symfony/var-dumper` va dans les `require-dev`).