Piggy Bank

1. Initialiser un nouveau projet symfony `--version=lts`
2. Créer une entity `PiggyBank` (= tirelire) avec les propriétés suivantes :
    - `balance` : float (= somme totale dans la tirelire, par defaut = 0)
3. Créer une entity `Transaction` avec les propriétés suivantes :
    - `type` : string (= "credit" | "debit")
    - `amount` : float (= montant de la transaction)
4. Effectuer les migrations nécessaires.
5. Créer une fixture afin de n'avoir qu'une seule tirelire en base (il n y en aura jamais plus d'une)
6. Executer les fixtures
7. Créer le CRUD pour l'entity `Transaction`
    - On doit pouvoir sélectionner le type "credit" ou "debit"
    - Le montant doit forcément être positif (il sera ajouté ou soustrait à la tirelire en fonction du type)
8. Créer un service `TransactionChecker`
    - Créer une méthode `isAllowed(string $type, float $amount) : bool` qui vérifie que la transaction est autorisée
        - Conditions:
            - On ne peut pas avoir une tirelire négative
            - La somme maximum dans la tirelire ne peut pas excéder 1000 $
9. Lors de l'ajout d'une `Transaction` via le CRUD, vérifier grâce au service que la transaction est autorisée. Si ce n'est pas le cas, afficher un message d'erreur.
10. Sur la page `/transaction/` on doit visualiser le montant actuel de la tirelire, et visualiser la liste des dernières transactions

11. **Unit Tests** :
    1. Créer une classe de tests comprenant 2 méthodes, et tester à l aide de PHPUnit que la méthode `isAllowed(...)` de notre service retourne false si on débite (methode 1) ou crédite (methode 2) un montant trop élevé
12. **Functionnal Tests** :
    1. A l aide de tests fonctionnels, vérifier que selon les données remplies dans le formulaire sur la page `/transaction/new` , un message d erreur apparaisse bien dans les 2 cas cités plus haut

Ressources : https://symfony.com/doc/current/testing.html

Nb: Si vous souhaitez utiliser une autre librairie que celle(s) proposée(s) dans la doc officielle Symfony, feel free