heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Gamification API
Présentation

Fabien Franchini
Sébastien Henneberger
Pascal Sekley
Rodrigue Tchuensu Pouopse

Démo de l'application gamifiée

- Démonstration
- Implémentation (Software)
- Tests
- Autoévaluation

Démo de l'application gamifiée

Scénario :

- **L'application est HeigvdOverflow**

  (forum d'entraide destiné aux étudiants de l'HEIGVD)

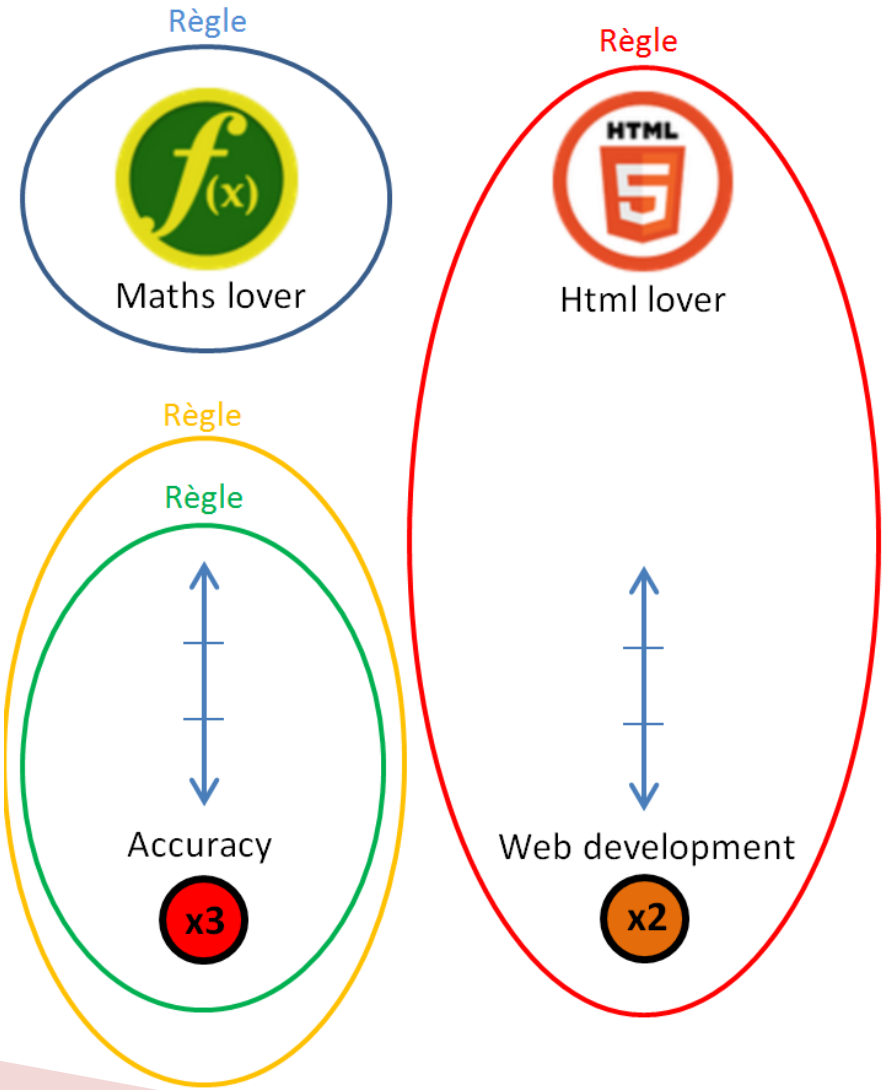
Démo de l'application gamifiée

Règle
Si eventType = solveMathsProblems ALORS
Donne le badge « Maths lover »

Règle
Si eventType = answerAccurately ALORS
+1 point sur Accurcy

Règle
Si eventType = solveHtmlProblems ALORS
+1 point sur Web development
Donne le badge « html lover »

Règle
Si eventType = answerNotAccurately
ALORS
-1 point sur Accuracy



Implémentation

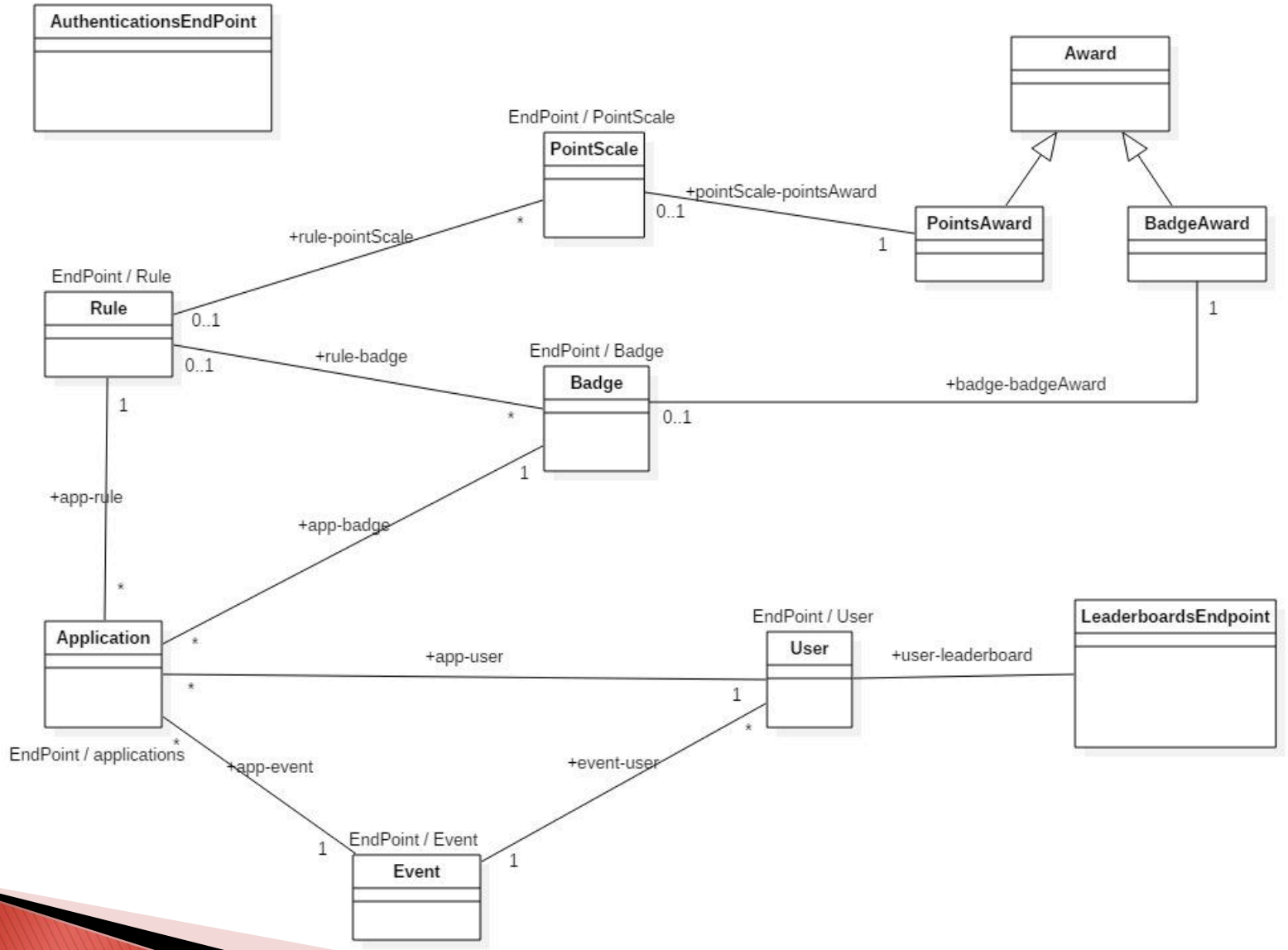
- Approche **Top-Down** avec Swagger (Editor)
- Stratégie d'héritage « **SINGLE_TABLE**»
- Utilisation des annotations **@Transactional** pour gérer la concurrence

Implémentation

Description de l'API

(fichier .yaml via http://editor.swagger.io/#/)

Implémentation



Implémentation

Sécurité implémentée :

- Mot de passe de l'application :
  - Exigence de complexité (minimum 7 caractères)
  - Transmis en clair durant l'enregistrement et l'authentification
  - Stocké en clair dans la base de donnée
  - Pas diffusé à travers l'API

Implémentation

Sécurité implémentée :

- Application gamifiée authentifiée grâce à un JWT :
  - Le token contient l'id de l'application
  - Le token à une durée de vie de 1 heure
  - Le token est signé par l'API
  - Envoyé avec chaque requête de l'application

    (placé dans l'en-tête Authorization, avec «Bearer»)

Tests

Fonctionnels :

- Implémentés en Javascript avec Mocha, Chai et Chance
- Endpoints testés complètement :
  - /Authentications
  - /Applications
  - /Badges
  - /PointScales

Tests

Fonctionnels :

- **Exemple pour /Applications :**

```
 The /applications endpoint :
   Test success for HTTP GET method :
     √ should allow an authenticated user to get the current application
(602ms)
   Test success for HTTP POST method :
     √ should allow an unauthenticated user to create a new application

   Test success for HTTP PUT method :
     √ should allow an authenticated user to completely update his application
(69ms)
   Test success for HTTP DELETE method :
     √ should allow an authenticated user to delete his application
(71ms)
   Test failures for HTTP GET method :
     √ should refuse an unauthenticated user to get his application if the authorization header is not provided
     √ should refuse an unauthenticated user to get his application if the authentication token is empty
     √ should refuse an unauthenticated user to get his application if the authentication token is not preceded by the Bearer pattern
     √ should refuse an unauthenticated user to get his application if the authentication token is not signed by the gamification API server
     √ should refuse an unauthenticated user to get his application if the authentication token is expired
     √ should refuse an authenticated user to get his application if the application does not exist
```

Tests

Fonctionnels :

- Scénario (workflow) :

```
  Scenarios :
    Authentications & applications operation :
      √ should allow an unauthenticated user to create a new application, to authenticate itself and to
get an authentication token

    The sending of an event must update the concerned user by applying the correct rule
      √ should allow an authenticated application to send an event and to see user badge update through
the application of the correct rule (54ms)
      √ should allow an authenticated application to send an event and to see user points update through
 the application of the correct rule (45ms)
      √ should allow an authenticated application to send an event and to see user badge and user points
 update through the application of the correct rule (61ms)
```

Tests

Non-fonctionnels :

(Lancement des scripts JMeter)

Auto-évaluation / critiques

Implémentation, points positifs :

- **Les fonctionnalités attendues sont implémentées**
- **Exploration de plusieurs types de tests**

(fonctionnel, charge, concurrence, sécurité)

- **L'authentification en fournissant le JWT**

Auto-évaluation / critiques

Implémentation, points à améliorer :

- Hacher le mot de passe de l'application
- Retourner des messages d'erreurs clairs pour l'utilisateur (texte)
- Mieux faire pour la gestion des exceptions
- Corriger le bug lors du lancement des tests fonctionnels avec la plateforme Dockerisée
- Compléter les tests fonctionnels pour /rules, /users/ et /events
- Passé à Cucumber pour les tests (spécification et réutilisation des étapes)

Auto-évaluation / critiques

Collaboration, points positifs :

- Communication, explication, entraide
- Répartition égale des tâches
- Bonne utilisation des outils de collaboration (Git)

Auto-évaluation / critiques

Collaboration, points à améliorer :

- Créer des règles de codage et les respecter
- Désigner un chef de groupe
- Mettre à disposition du groupe (push) du code testé, commenté et propre.