**DOCUMENTATION TECHNIQUE**

**SERVICE DE COURSE**

Le service se présente sous la forme d'une API REST qui comporte 3 endpoints :

* + **/horse/create** : crée un cheval en base avec son nom
  + /**race/createOrUpdate** : crée une course non-existante avec minimum 3 chevaux ou modifie une course avec un nombre indéterminé de chevaux
  + **/race/show** : affiche les informations relatives à une course
  + **/kafka/send** : envoie les informations relatives à une course sur un topic Kafka.

La fin du document est consacré aux tests JUnit.

**/horse/create**

**entrée** : JSON contenant le nom du cheval à créer

exemple :

{

"name":"horse5"

}

**sortie** : JSON représentant le cheval créé.

{

"name":"horse5",

"id": 5

}

**/race/createOrUpdate**

**entrée**: JSON contenant le jour, le nom et le numéro unique pour ce jour

exemple :

{

"day":"2023-05-25",

"name":"race2",

"daily\_id":3

}

**sortie**: JSON représentant la course créée ou modifiée.

RaceId : clé primaire composite de la course. Cette clé est composé du jour, du nom et du numéro unique pour ce jour.

horseAndPositionList : liste des paires Cheval/Position relative à cette course. Un cheval peut être affecté à une course un jour donné et à une autre course un autre jour ou le même jour avec un autre numéro pour ce jour. Chaque élément de cette liste de paires Cheval/Position contient :

* "id" : clé primaire de la paire, élément uniquement à titre technique, il n'a pas de valeur métier.
* "racePosition" : la position du cheval sur la ligne de course
* "horse" : cheval lui-même composé de :

"name": son nom

"id" : clé primaire du cheval, élément uniquement à titre technique, il n'a pas de valeur métier.

exemple :

{

"raceId": {

"day": "2023-05-25T00:00:00.000+00:00",

"name": "race2",

"daily\_id": 3

},

"horseAndPositionList": [

{

"id": 5,

"horse": {

"name": "horse2",

"id": 2

},

"racePosition": 1

},

{

"id": 6,

"horse": {

"name": "horse4",

"id": 4

},

"racePosition": 2

},

{

"id": 7,

"horse": {

"name": "horse5",

"id": 5

},

"racePosition": 3

}

]

}

**/race/show**

**entrée**: JSON contenant le jour, le nom et le numéro unique pour ce jour.

exemple :

{

"day":"2023-05-25",

"name":"race2",

"daily\_id":3

}

**sortie**: JSON représentant la course .

{

"raceId": {

"day": "2023-05-25T00:00:00.000+00:00",

"name": "race2",

"daily\_id": 3

},

"horseAndPositionList": [

{

"id": 5,

"horse": {

"name": "horse2",

"id": 2

},

"racePosition": 1

},

{

"id": 6,

"horse": {

"name": "horse4",

"id": 4

},

"racePosition": 2

},

{

"id": 7,

"horse": {

"name": "horse5",

"id": 5

},

"racePosition": 3

}

]

}

**/kafka/send**

**entrée** : JSON représentatif de la course avec toutes ses informations.

exemple :

{

"raceId": {

"day": "2023-05-25T00:00:00.000+00:00",

"name": "race2",

"daily\_id": 3

},

"horseAndPositionList": [

{

"id": 5,

"horse": {

"name": "horse2",

"id": 2

},

"racePosition": 1

},

{

"id": 6,

"horse": {

"name": "horse4",

"id": 4

},

"racePosition": 2

},

{

"id": 7,

"horse": {

"name": "horse5",

"id": 5

},

"racePosition": 3

}

]

}

**sortie**: envoie le JSON relatif à la course sur le bus Kafka.

**Tests JUnit**

Les tests JUnit se trouvent dans le package src.test.java.com.pmu.races.

Dans Eclipse, exécuter les tests en cliquant sur Run As → JUnit.

La base de données de test est *embedded,* ce qui signifie qu'il n'y a pas besoin d'installer le schéma de la base en local pour tester. Spring Boot et JPA créent le schéma automatiquement à l'aide des annotations du code sur cette base embarquée. Un fichier de configuration spécial /races/src/test/resources/application.properties a été crée pour les tests.