UnibetTestLive :

Version utilisées :

Java : jdk-16.0.2
Maven : 3,6,2

La version 16 de Java permet l'utilisation de stream().toList()

Librairies externes :

```
io.swagger.core.v3
javax.validation
org.openapitools
```

Ces 3 librairies sont utilisées pour la génération automatique du modèle DTO décrit dans le contrat d'interface unibetTestLiveV1.yaml.

Elles permettent aussi, via le contrat d'interface, de générer une interface pour chaque type d'API, interface qu'il suffit d'implémenter ensuite en « overridant » les fonctions générées et annotées. Cela permet d'alléger grandement le code dans les controllers.

Via la librairie `org.springdoc` la documentation spécifiée dans le contrat d'interface peut être retrouvée via l'url {server.address}/swagger-ui/ .
Dans le cas de l'application : http://localhost:8887/swagger-ui/index.html#/

L'utilisation du contrat d'interface lors du design d'API est nécessaire pour définir le modèle et les requêtes échangées entre clients et fournisseurs. L'utilisation des librairies sus-mentionnées permet d'éviter toute ambiguïté lors de l'implémentation de l'API chez les client comme chez le fournisseur car le modèle est généré à partir du document même.

Pour les codes de réponse personnalisés (600, 601,602), j'ai changé la spécification afin de les faire rentrer dans l'intervalle 452-499 en adéquation avec la doc https://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml

Librairies de mapping :

```
fr.xebia.extras
```

Cette librairie permet d'implémenter des mappers d'objet à objet qui compilent, ce qui rend le débugging beaucoup plus simple. Je l'utilise pour mapper mes objets DTO générés via openAPI en objets DAO déjà implémentés. Il permet d'ignorer certains champs via des annotations et de customiser le mapping sans alourdir le code.