*Guilbert Simon*

*IGIRANEZA Beda*

**Plan de test**

***INFO 832 : tp1 Virtual pub***

# Le plan de test

## But

Ce plan de test est défini pour le projet **Virtual Public Bar** qui a les objectifs suivants :

* Assurer le bon fonctionnement d'approvisionnement du bar via le stock (la cave).
* Assurer l'arrivée des livraisons et la mise à jour du stockage tout en sachant que toutes les livraisons du Pub se font directement dans la cave.
* Gérer les commandes du bar “sur place ou à emporter".
* Assurer la bonne gestion de trois types de boissons proposée par le Pub (boissons chaudes, boissons alcoolisées et boissons froides).
* Assurer la bonne gestion des composants des Cocktails qui peuvent être alcoolisés ou non.

Pour ce faire, nous allons tester tous types d'entrées en vérifiant la bonne prise en compte de la valeur saisie. Nous allons tester toutes les classes, les méthodes et les fonctions présentes dans le code.

Nous allons créer les fonctions de validation génériques, et donc en faire des fonctions statiques, qui seront importables dans la classe qui contient les tests.

Nous allons créer les fonctions de validation et exécuter tous les tests.

Nous n’allons pas créer de test de performance et de stress.

Enfin, nous allons nous assurer qu’aucun des tests ne plante pas et que les exceptions sont bien gérées.

Portée

Pour tester le code, nous allons nous baser sur les différents types de test tels que :

* Test de la boîte blanche
* Tests fonctionnels ou non fonctionnels
* Tests unitaires que nous pouvons automatiser par la suite.
* Tests d'intégration pour s'assurer que toutes les classes, les fonctions et les méthodes vont bien ensemble.
* Tests de recette ou tests standard (nous allons se mettre dans la peau d’un utilisateur du Pub)
* Tests système pour s’assurer que tout va bien fonctionner quand le premier utilisateur final utilisera l’application

Nous n’allons faire qu’une simulation de tests d'acceptabilité puisque nous n’avons pas de véritable barman à notre disposition pour tester l’application dans un contexte réel.

Liste de fonctionnalités qui doivent être testées sont spécifiées ci-dessus.

Documents

|  |  |  |  |
| --- | --- | --- | --- |
| **Document** | **Version** | **Disponible** | **Notes** |
| Cas d’utilisation | - | Non |  |
| Documents d’architecture | 0.1 | Oui | La java doc fournie avec le code |
| Planning du projet | - | Non |  |
| Spécifications supplémentaires | 0.1 | Oui | Nous avons l’énoncé |
| Le code source | 0.1 | Oui |  |
| La documentation | 0.1 | Oui | La java doc fournie avec le code |
| Contact d'équipe de développement |  | Oui | Nous avons l’enseignant à qui nous pouvons poser des questions |

### Les exigences des tests

* Nous allons tester la classe **Bar** et différentes méthodes tels que :
  + Ajouter une boisson :add(Boisson boisson)
  + Ajouter une Cocktail :add(Cocktail cocktail)
  + serv (java.lang.String command)
  + toString ()

* La classe **Boisson** et sa méthode **toString**()
* La classe **Cave** et la méthode d’ajout d’une boisson **: add(Boisson b)**,  la méthode **take** () et **toString ()**
* La classe **Cocktail** et la méthode **add() , alcoolFree()** et **toString ()**

A savoir, à chaque classe nous allons tester son ou ses **constructeurs** et ses **assesseurs** (les getters et les setters) s’il y en a.

## La stratégie de test

Tableau de description des tests unitaires

|  |  |
| --- | --- |
| **Objectif du test** | Tests d'affirmation |
| **Technique** | On crée un objet puis on vérifie s’il a bien été prise en compte. |
| **Critère d’achèvement** | L’obtention de l’objet demandé à l’aide des asserts (assertEquals, assertSame, assertTrue, assertNull, assertNotNull, assertThrows) |
| **Considération spéciale** | On valide le test une fois le résultat prédit égal au résultat obtenu |

Les tests d’acceptation

Tableau de description des tests d'acceptation

|  |  |
| --- | --- |
| **Objectif du test** | Arriver à déterminer les tests qui passent avec succès et ce qui échouent. |
| **Technique** | Vérifier si l’on peut créer un objet pour chaque classe.  Vérifier le bon fonctionnement de chaque méthode et vecteurs. |
| **Critère d’achèvement** | Bien implémenter le test afin qu’il puisse être exécuté avec succès |
| **Considération spéciale** | Le résultat du test après l'exécution (failures ou non) |

Les outils

Liste des outils utilisés par les tests

|  |  |  |
| --- | --- | --- |
| **Outil** | **Version** | **Commentaire** |
| JUnit | 5 | Pour exécuter les tests |
| eclipse | Version : 2020-12 (4.18.0) | Utilisé pour implémenter et exécuter les tests |
| EclEmma | Version 3.1.3 | Couverture du code |
| Sonar Lint | Version 5.7 | Métriques de qualité |

Les ressources système

Tableau des ressources système

|  |  |
| --- | --- |
| **Ressources système** | |
| **Ressource** | **Nom/type** |
| Ordinateur | Un ordinateur suffisamment performant pour faire tourner le logiciel Eclipse |
| Logiciel | eclipse |
|  |  |
|  |  |
|  |  |

Le planning des tests

|  |  |  |  |
| --- | --- | --- | --- |
|  | **A faire** | **En cours** | **Terminé** |
| **Première séance du 3 mars 2021 (1h30)** | - Prise en main du projet  - Commencer à réfléchir aux tests  - Commencer le plan de tests | - Plan de tests |  |
| **Deuxième séance du 5 mars 2021 (4h)** | - Définir les tests à réaliser  - Continuer le plan de tests  - Implémenter les tests  - Générer les rapports de tests  - Générer le rapport des métriques du code | - Plan de tests | - Implémentation des tests  - Génération des rapports |
| **Travail post-deuxième séance** | - Terminer le plan de tests |  | - Plan de tests terminé |

Les jalons

Tableau des jalons des tests

|  |  |  |  |
| --- | --- | --- | --- |
| **Activité** | **Charge de travail** | **Début (jj-mmm-aa)** | **Fin (jj-mmm-aa)** |
| Planifier les tests | Guilbert Simon/ Beda Igiraneza | 03/03/2021` | 03/03/2021` |
| Relecture du code | Guilbert Simon/ Beda Igiraneza | 03/03/2021` | 03/03/2021` |
| Concevoir les tests | Guilbert Simon/ Beda Igiraneza | 03/03/2021` | 05/03/2021` |
| Exécuter les tests unitaires | Guilbert Simon/ Beda Igiraneza | 05/03/2021` | 05/03/2021` |
| Suivi des tests | Guilbert Simon/ Beda Igiraneza | 05/03/2021` | 12/03/2021` |
| Corriger les défauts | Guilbert Simon/ Beda Igiraneza | 05/03/2021` | 12/03/2021` |
| Exécuter les tests d’acceptation | Guilbert Simon/ Beda Igiraneza | 05/03/2021` | 12/03/2021` |

Les livrables de l’activité de test

Tableau des livrables

|  |  |  |  |
| --- | --- | --- | --- |
| **Délivrable** | **Destination** | **Origine** | **Date** |
| Une classe de test pour la classe  Boisson, Cave, Cocktail et Bar | L'équipe de développement | Créé par les testeurs | 12/03/2021` |
| Rapport de test | L'équipe de développement | Créé par les testeurs | 12/03/2021` |
| Rapport de couverture de chemins | L'équipe de développement | Créé par les testeurs | 12/03/2021` |