Circus Air Hockey

Plan de tests logiciels

Version 1.4

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2013-04-01 | 1.0 | Élaboration du squelette. Début de rédaction de tableaux | François |
| 2013-04-02 | 1.1 | Rédaction de tableaux | Félix, François, Alexandre, Simon, Philippe, Émile |
| 2013-04-03 | 1.2 | Section 1, 4 | Félix |
| 2013-04-04 | 1.3 | Section 2 | Philippe |
| 2013-04-04 | 1.4 | Rédaction, Relecture | Félix |

Table des matières

1. Introduction 4

1.1 But 4

1.2 Mise en contexte 4

1.3 Documentation 4

2. Exigences à tester 4

3. Stratégie de test 10

3.1 Types de test 10

3.1.1 Tests de fonction 10

3.1.2 Tests d’interface usager 11

3.1.3 Tests d’intégrité des données 11

3.1.4 Tests de performance 12

3.1.5 Tests de charge 12

3.1.6 Tests de stress 13

3.1.7 Tests de volume 13

3.1.8 Tests de sécurité et de contrôle d’accès 14

3.1.9 Tests d’échec/récupération 15

3.2 Outils 16

4. Ressources 16

4.1 Équipe de test 16

4.2 Système 17

4.2.1 Client Lourd 17

4.2.2 Client Léger 17

5. Jalons du projet 17

Plan de tests logiciels

# 

# Introduction

## But

Ce plan de tests pour le *Circus Air Hockey* a pour but d'identifier les composantes à tester, d'expliciter les stratégies de tests utilisées pour les tester ainsi que les ressources utilisées pour y arriver, et enfin de détailler les jalons relatifs à la discipline de tests.

## Mise en contexte

Dans le cadre du projet 3, nous sommes partis d'un jeu interactif 3D programmé en Java/C++ pour le rendre multi-joueurs en ligne et changer toute la couche Java pour du C#. Les changements majeurs furent surtout la reconstruction des interfaces et bien sûr l'ajout de la couche réseau et d'un serveur, ainsi que l'ajout d'une base de données. C'est pourquoi un effort accru fut porté aux tests concernant ces éléments nouveaux, la logique C++ du jeu ayant moins changé.

## Documentation

Les documents suivants étaient disponibles pour l'identification, l'exécution et la validation des tests:

* SRS
* Document d'architecture (cas d'utilisation, diagrammes de séquences)

# Exigences à tester

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonctionnalité** | **Interface** | **Base de données** | **Calcul** |
| 1.1 L’utilisateur aura à sa  disposition différents  menus selon son contexte  courant. | S'assurer que les choix du  client sont clairs et peu  complexes. | Aucun | Aucun |
| 1.1.1 L'utilisateur pourra  redimensionnement la  fenêtre du client. | S'assurer que les  éléments et les menus se  repositionnent de  manière ergonomique. | Aucun | Aucun |
| 1.1.2 Le menu d'option  permettra à l'utilisateur  de modifier les touches  fonctionnelles  lors des parties. | S'assurer que l'entrée de  l'utilisateur est  correctement enregistrée. | Aucun | Aucun |
| 1.1.3 Les menus  accessibles lors des parties  permettront de modifier  la vue de l'utilisateur. | S'assurer que les  différentes vues sont  fonctionnelles ainsi que le  « replay ». | Aucun | Aucun |
| 1.2 Le menu de  connexion en ligne  permettra à l'utilisateur  d'entrer son nom dans un  champ de texte normal et  son mot de passe dans un  champ de texte caché  pour se connecter. | S'assurer que le mot de  passe de l'utilisateur n'est  pas visible. | Accède à la table  contenant les utilisateurs  pour vérifier que  l'information fournie est  présente et correcte. | Aucun |
| 1.2.1 L'utilisateur peut  aussi décider de créer un  nouveau profil. | S'assurer que le mot de  passe choisi est aussi  dans un champ de texte  caché et qu'il est le même  dans le champ  « confirmation ». | Ajoute à la table  contenant les utilisateurs  le nom choisi avec le mot  de passe si celui n'existe  pas déjà. | Aucun |
| 1.2.2 Un nouveau profil  aura des statistiques  partant de zéro et aura  l'avatar par défaut. | S'assurer que les  nouveaux profils restent  constants. | Ajoute les statistiques de  base au profil crée. | Aucun |
| 1.3 L'utilisateur pourra  accéder à son profil du  menu principal du mode  en ligne et modifier les  éléments de son avatar et  son maillet. Il enregistra  les changements en  appuyant sur  « enregistrer ». | S'assurer que l'utilisateur  puisse voir les choix qui  lui sont présentés sans  devoir enregistrer les  modifications pour le  savoir. | Vérifie si l'utilisateur a  suffisamment de victoire  pour certains éléments des  avatars et enregistre les  changements si c'est le  cas. | On considère que le nombre de victoires du client est égal ou supérieur au nombre de victoires nécessaires d'un élément. |
| 1.3.1 L'utilisateur pourra  voir les avatars des autres  joueurs en ligne dans les  lobby des parties. | S'assurer que les avatars  de tous les autres joueurs  sont conformes aux  choix qu'ils ont  enregistrés. | Recherche dans la base  de données les éléments  de chaque utilisateur  dans la partie pour dire  au client lesquels  afficher. | Aucun |
| 1.3.2 L'utilisateur pourra  accéder à un tableau  contenant les meilleurs  scores. | S'assurer que le tableau  affiche les scores  adéquatement et en  ordre décroissant (Du  meilleur au moins bon). | Recherche tous les  statistiques nécessaires  des profils d'utilisateur. | Aucun |
| 1.4 L'utilisateur pourra  crée des parties et des  tournois localement. Il  n'a pas à se connecter  pour jouer au mode local. | S'assurer que l'option de  jouer se mode local est  présent dans les premiers  choix de l'utilisateur  lorsqu'il part le client. | Aucun | Aucun |
| 1.4.2.1 L'utilisateur  pourra voir les tables qu'il  a créé et les tables de  d'autres joueurs rendu  public selon l'onglet. | S'assurer que les tables se  retrouvent dans l'onglet  respectif. | Aucun | Aucun |
| 1.4.3 Les joueurs virtuels  contourneront les murs  placés sur les tables ainsi  que naviguer sur  n'importe quelle table  crée par un utilisateur. | S'assurer que le maillet  des joueurs virtuels  puisse reconnaître le bon  chemin pour se rendre  vers la rondelle. | Aucun | Utilise un algorithme  semblable à Dijkstra pour calculer les déplacements. Le chemin calculé doit être optimal. |
| 1.5 L'utilisateur connecté  pourra se joindre au  lobby principal puis  accéder à des parties et  des tournois en ligne  contre d'autre utilisateur. | S'assurer que l'option  partie rapide et tournoi  rapide mène vers la  création de leur type  respectif s'il n'en n'existe  pas présentement sur le | Aucun | Aucun |
| 1.5.1 L'utilisateur pourra  visionner les parties non pleines  une fois dans le  lobby principal. | S'assurer que les parties  affichées ont leur nom  correctement affiché et  que le nombre de joueur  présent se met à jour  dynamiquement. | Aucun | Aucun |
| 1.5.2 L'utilisateur aura la  possibilité de clavarder  dans le lobby principal  et le lobby de la partie ou  tournoi dans lequel il se  trouve. | S'assurer que l'utilisateur  puisse voir le message  qu'il écrit et que les  messages envoyer à un  lobby n'apparaisse que  pour les autres joueurs  présents dans ce lobby. | Aucun | Aucun |
| 1.5.2.1 Le clavardage du  jeu contiendra des mots  censurés par le client. | S'assurer que les mots  choisi soit correctement  remplacé par d'autre mot  plus approprié. | Contient la liste des mots  censurés et leur  transformation. | Aucun |
| 1.5.3 L'utilisateur qui crée  une partie ou un tournoi  pourra choisir le nom de  la partie, le nombre de  joueur, la table et le  activer le mode soufflé  ou non. | S'assurer qu'une nouvelle  parties ou un nouveau  tournoi s'ajoute à la liste  des parties existantes  dans le lobby principal  pour les autres  utilisateurs. | Aucun | Aucun |
| 1.6 Les vues disponibles  lors les parties seront  orthogonales, libre,  orbite et ciel. | S'assurer que l'utilisateur  puisse changer sa vue une  fois la partie commencée. | Aucun | Aucun |
| 1.6.1 La vue pourra se  déplacer avec la souris et  selon le mode actif. | S'assurer que les valeurs  d'entrée de la souris soit  adéquate pour les  mouvements de la vue. | Aucun | L'entrée de la souris sera  pris en compte pour les  calcul de la nouvelle  position de la caméra. |
| 1.7 Lors des parties, la  physique du jeu  respectera une physique  dit 2.5D qui inclura la  friction, les collisions et  les autres éléments  particuliers du jeu. | S'assurer que la rondelle  et les maillets de  l'utilisateur soiten limités par  les contraintes établies et  que tout objet autre est  l'effet désiré sur la  rondelle. | Aucun | La rondelle prendra en  compte la plupart des  calculs nécessaires. Elle  devra gérer ses actions  selon les objets qu'elle  rencontre en prenant en  compte sa propre vitesse  et direction. |
| 1.7.1 Il sera possible  d'observer le même type  de physique lors des  parties en ligne. | S'assurer que le serveur  effectue correctement les  calculs ainsi que les  clients. | Aucun | Le serveur fera les mêmes  calculs que les clients  pour transmettre les  informations aux joueurs. |
| 1.7.2 L'utilisateur pourra affecter les jets d’air avec du bruit si le soufflage est activé. | S’assurer que faire du bruit dans le micro affecte correctement les jets d’air et donc la rondelle. | Aucun | La valeur d’entrée du micro affectera le calcul de temps de vie de nouvelles particules ainsi que sa force. |
| 1.7.2.1 L'utilisateur  pourra calibrer son micro  au préalable dans les  options. | S'assurer que le calibrage  du micro influence  correctement le micro  pour l'utilisateur. | Aucun | Calibration de micro selon  le son ambiant. |
| 1.7.3 Un muret pourra  être destructible. | S'assurer qu'après un  certain nombre de coup  de rondelle, un muret  disparaisse. | Aucun | Calculer la vie restante  d'un muret suite à un  impact. Prendre en  compte la vitesse de  l'impact. |
| 1.7.4 L'utilisateur  pourra « tilt » la table  pour décoincé la rondelle. | S'assurer qu'un utilisateur  ne puisse abuser de cette  l'option. | Aucun | Imposer un délai entre  chaque utilisation de  manière incrémentale. |
| 1.7.5 L'utilisateur pourra  voir certaines annonces durant la partie. | S'assurer que l’affichage  de ses éléments ces  éléments soient visibles  peu importe la vue. | Aucun | Aucun |
| 1.7.6 Il y aura la présence  d'animation de caméra et  de reprise vidéo. | S'assurer que la reprise  fonctionne et que les  mouvements de caméra  sont fonctionnels. | Aucun | Aucun |
| 1.7.7 Il sera possible  d'observer de l'éclairage  sur la table de jeu et sur  les objets 3D. | S'assurer des résultats des  lumières avec les surfaces  et normales des éléments  du jeu ne cause pas  d'incohérence graphique. | Aucun | Aucun |
| 1.7.7.1 La « skybox » ne  sera pas affectée par le  lumière pour assurer une  belle continuité sur sa  texture. | S'assurer que la  « skybox » et les lumières  n’interagissent pas. | Aucun | Aucun |
| 1.7.8 L'utilisateur pourra  changer de musique lors  de la partie avec l'aide  d'une radio virtuelle. | S'assurer la possibilité de  plusieurs postes de radio  et que la musique joue en  même temps que les sons  provenant des éléments  physiques du jeu. | Aucun | Aucun |
| 1.7.8.1 Les utilisateurs de  parties en ligne pourront  changer la radio. | S'assurer que tout joueur  puisse changer le poste  de radio. | Aucun | Aucun |
| 1.7.9 S'il s'agit d'une partie en ligne, l'utilisateur pourra être  un spectateur. | S'assurer qu'un joueur  puisse se joindre à une  partie en cours pour  l'observer. | Aucun | Aucun |
| 1.7.9.1 Un spectateur  pourra parler dans le  clavardage de la partie en  cours, changer le poste de  radio et bouger sa vue  mais rien d'autre. | S'assurer que les  spectateurs n'aient accès  qu’aux fonctionnalités de  clavardage, de radio, et  de vue. | Aucun | Aucun |

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonctionnalité** | **Interface** | **Base de données** | **Calcul** |
| 2.1 L’utilisateur pourra  créer une nouvelle table ou modifier une table  sauvegardée dans le  menu principal. | S'assurer que les tables  sauvegarder soit  reconnaissables en  vignette d'aperçu. | Aucun | Aucun |
| 2.2 L'utilisateur pourra  changer de mode de vue  dans le mode édition. | S'assurer que le mode  courant est clairement  indiqué. | Aucun | Aucun |
| 2.2.1 L'utilisateur pourra  se déplacer de différentes  manière selon son mode  de vue. | S'assurer que le  glissement de doigts sur  l’iPad fonctionne comme  dicté dans le SRS. | Aucun | Aucun |
| 2.2.1.1 L'angle de la vue  libre pourra être modifié  selon l'angle de l’iPad. | S'assurer que le contrôle  de l'angle est intuitif et  agréable. | Aucun | Calcul de l'angle de la  camera en vue libre selon  les données provenant du  iPad. |
| 2.2.1.2 L'utilisateur devra  utiliser des boutons de  l'interface pour avancer  et reculer en vue libre. | S'assurer que la vitesse  rend les contrôles  manœuvrables. | Aucun | Incrémentation de la  position de la camera  dans les axes appropriés. |
| 2.3 L'utilisateur pourra  modifier les objets  présents sur la table et en  ajouter. | S'assurer que les objets  ne puissent être en dehors  de la zone d’édition ou de  la table dans le cas des  maillets et de la rondelle. | Aucun | Calcul des positions des  objets et s'ils dépassent  les valeurs limites ou non. |
| 2.3.1 Il sera possible de  modifier la taille et  l'orientation d'un objet  en glissant deux en ayant  un objet sélectionné. | S'assurer que les maillets  et la rondelle ne puissent  pas être modifier de cette  manière. | Aucun | Calcul de la nouvelle taille  selon la variation de  distance entre les doigts  et l'angle selon la  variation d'alignement. |
| 2.3.2 L'utilisateur pourra  aussi ajouter des objets  en glissant son doigt des  boutons d'objet vers la table. | S'assurer que l'objet  apparaît dès que le doigt  de l'utilisateur glisse hors  de la zone du bouton. | Aucun | Aucun |
| 2.3.3 L'utilisateur aura la  possibilité de supprimer  des objets en le sortant  de la zone d'édition. | S'assurer que l'objet hors  de la zone d'édition  paraisse un peu rouge, le  marquant comme objet  allant être supprimé. | Aucun | Aucun |
| 2.3.4 Un objet pourra être  duplicable. | S'assurer que le nouvel  objet à la même taille et  angle que son parent. | Aucun | Aucun |
| 2.3.5 Il sera possible de  sélectionner multiple  objet en même temps. | S'assurer que peser avec  deux doigts sur un objet  ne désélectionne pas un  autre objet. | Aucun | Aucun |
| 2.3.5.1 Toutes  fonctionnalités applicables  aux objets peuvent  s'appliquer à une sélection  multiple. | S'assurer que les maillets  et la rondelle reste  toujours la même taille et  angle. | Aucun | Aucun |
| 2.3.6 La table pourra être  modifiée mais restera  tout de même symétrique  pour des raisons d'égalité  entre joueurs. | S'assurer que toutes  formation possible de  table s'affiche  correctement. | Aucun | Calcul des limites des  coins de la table pour  bien effectuer l'affichage  de la table utilisant un  stencil. |
| 2.4 En brassant l’iPad,  l'utilisateur pourra  bouger de manière  aléatoire les objets sur la  table. | S'assurer que les objets  reste dans leur zone  limite respective. | Aucun | Aucun |
| 2.4.1 En revirant l’iPad  de bord, l'utilisateur  supprimera tous les  objets sauf les maillets et  la rondelle. | S'assurer que les maillets  et la rondelle restent sur la  table. | Aucun | Aucun |
| 2.5 Il sera possible  d'accéder à une fenêtre  de configuration pour  limiter la taille maximale  de la table, sa friction et  le coefficient de rebond. | S'assurer que le mode de  configuration applique  bien les variables entrées  par écrit ou avec la bar de  choix. | Aucun | Aucun |
| 2.6 L'utilisateur aura  accès aux fonctions  « undo » et « redo ». | S'assurer que la méthode  enregistre au moins 10  dernières modifications. | Aucun | Aucun |
| 2.7 L'éclairage de l’iPad  devrait fonctionner  comme dicté en 1.7.7 | Voir 1.7.7 | Aucun | Aucun |
| 2.8 Lors de retour vers le  menu principal de l’iPad  ou de l'application, il  faudra sauvegarder la  table en cours. | S'assurer que la sauvegarde de table sur iPad fonctionne correctement. | Aucun | Aucun |
| 2.8.1 Il faudra aussi que  l'application sauvegarde  un aperçu de la table  pour que celle si soit  reconnaissable pour  l'utilisateur à son retour. | S'assurer que peut  importe l'orientation du  iPad lors de la  sauvegarde, les vignettes  doivent être  pratiquement identiques. | Aucun | Aucun |
| 2.9 Il sera possible pour  l'utilisateur d'envoyer sa  table au serveur. | S'assurer qu'un utilisateur  sache immédiatement s'il  est capable de rejoindre  le serveur. | Aucun | Aucun |
| 2.9.1 L'utilisateur peut  utiliser son compte s'il en  possède lorsqu'il envoie sa  table. | S'assurer que la table soit  présente sur son compte  sur le client lourd dans  l'option « plus de choix » | Enregistre le nom de la  table, si elle est publique  ou no et le nom de son  créateur. | Aucun |
| 2.9.1.1 Un utilisateur  peut aussi envoyer une  table anonymement. | S'assurer que la table soit  automatiquement  publique. | Enregistre le nom de la  table, qu'elle est publique  et le créateur est  anonyme. | Aucun |

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonctionnalité** | **Interface** | **Base de données** | **Calcul** |
| 3.1 Le serveur devra au  moins avoir une interface  en ligne de commande. | S'assurer que les  commandes disponibles  puissent remplir les  fonctions nécessaires. | Aucun | Aucun |
| 3.2 Le serveur devra être  capable d’administrer au  moins 5 parties  simultanées. | S'assurer que le nombre  de joueurs en ligne en  même temps peut être  plus haut que la limite  prévu. | Aucun | Aucun |
| 3.3 Le serveur devra gérer  les différents messages  qu'on lui envoie. | S'assurer qu'un nombre  adéquat de message  puisse être traité. | Aucun | Aucun |
| 3.3.1 Certains de ces  messages demanderont  des calculs pour le jeu. | S'assurer que la physique  calculer dans le serveur  est la même que celle du  jeu. | Aucun | Calcul la physique de la  rondelle selon les  informations fournies par  les clients. |

# Stratégie de test

## Types de test

### Tests de fonction

|  |  |
| --- | --- |
| Objectif de test: | S’assurer que tous les comportements observables décrits dans le SRS ont été implémentés. S’assurer que tous les cas d’utilisation décrits dans le document d’architecture peuvent être réalisés. |
| Technique : | Pour chaque comportement observable décrit dans le SRS :   * Observer le comportement dans l’application. S’assurer qu’il correspond exactement à la description du SRS. * Lorsque c’est possible, tenter d’effectuer des actions invalides en lien avec le comportement observable. S’assurer que les actions invalides sont signalées par un message d’erreur clair. * S’assurer qu’il n’y a aucun «crash» ou comportement inattendu.   Pour chaque cas d’utilisation décrit dans le document d’architecture :   * Réaliser le cas d’utilisation dans l’application. |
| Critère de complétion : | * Tous les comportements observables décrits dans le SRS ont pu être observés et correspondent exactement à leur description (effectuer les corrections au besoin). * Tous les cas d’utilisation peuvent être réalisés (effectuer les corrections au besoin). |
| Considérations spéciales: |  |

### Tests d’interface usager

|  |  |
| --- | --- |
| Objectif de test: | S’assurer que l’interface utilisateur est uniforme et respecte les règles d’utilisabilité enseignées dans le cours LOG2420. |
| Technique: | Pour chaque écran de l’application, un testeur s’assure que :   * L’écran respecte les standards de navigation (titre, bouton retour, boutons vers les prochains écrans). * L’écran respecte les standards de navigation entre les champs avec le clavier et la souris. * Des messages standards sont affichés pour toutes les actions invalides identifiées.   Pour chaque écran de l’application, un testeur observe un néophyte tenter d’exécuter les comportements observables décrits dans le SRS et s’assure que :   * Les comportements observables peuvent être atteints dans des délais raisonnables spécifiés avant l’exécution des tests. * Le néophyte obtient les résultats voulus suite à ses actions. |
| Critère de complétion: | Chaque écran respecte les standards fixés dans l’équipe. Un néophyte est capable d’utiliser l’application selon les attentes. |
| Considérations spéciales: | La participation d’un néophyte est nécessaire pour exécuter ces tests. |

### Tests d’intégrité des données

|  |  |
| --- | --- |
| Objectif de test: | S’assurer que la base de données ne contienne pas de données corrompues et que les actions de base fonctionnent correctement (retournent les résultats attendus). |
| Technique: | Reproduire tous les types de requêtes pouvant être effectués sur la base de données sans avoir recours à l’interface usager. Pour cela, on enverra des messages simulés au serveur à l’aide de scripts.  À chacune des requêtes, vérifier, en regardant dans la base de données, si les données sont correctes.  Il faut aussi faire ces requêtes en utilisant des données non valides pour voir qu’il n’y ait pas de corruption de données et que la base de données retourne des messages d’erreur. |
| Critère de complétion: | Chaque action fonctionne bien et la base de données ne contient aucune information corrompue. |
| Considérations spéciales: | Pour pouvoir effectuer ces tests, il faut avoir accès directement à la base de données et il faut connaître toutes les étapes que comporte chaque action. |

### Tests de performance

|  |  |
| --- | --- |
| Objectif de test: | S'assurer que les temps de communication TCP et UDP soient adéquat au bon fonctionnement du jeu dans un contexte d’utilisation moyen. |
| Technique: | Tester sur quelques machines les différents modes de jeu en ligne, avec différentes tables de jeu.  Tester le déroulement d'une partie en ligne avec plusieurs joueurs et spectateurs |
| Critère de complétion: | Dans tous les cas le temps de réponse devra être négligeable, voir instantané du point de vu de l'utilisateur |
| Considérations spéciales: |  |

### Tests de charge

|  |  |
| --- | --- |
| Objectif de test: | S'assurer que les exigences non fonctionnelles du SRS ayant rapport à la performance soient comblées. S’assurer que l’application réagit bien lorsque le nombre maximal de clients supportés est atteint. |
| Technique: | Mémoire vive   * Sur une même machine, exécuter plusieurs instances du client lourd et léger puis analyser l'état de la mémoire et la fluidité du jeu.   Réseau   * Connecter le plus grand nombre requis de clients à un même serveur (sur le réseau de l'école). Avec plusieurs testeurs, effectuer plusieurs requêtes simultanées à partir des clients. (TCP) * Analyser la fluidité lors du plus grand nombre requis de parties en réseau simultanées. (UDP) |
| Critère de complétion: | Dans tous les cas, le jeu devra rester fluide et cohérent pour tous les joueurs.  Les comportements indiqués dans le SRS devront être respectés  La mémoire utilisé par une instance de jeu devra respecter les valeurs définies dans le SRS. |
| Considérations spéciales: |  |

### Tests de stress

|  |  |
| --- | --- |
| Objectif de test: | Découvrir les points critiques de l'application, surtout sur la couche réseau, et prendre des mesures préventives pour éviter les plantages |
| Technique: | Réseau   * Connecter un grand nombre de clients à un même serveur (sur le réseau de l'école). Avec plusieurs testeurs, effectuer plusieurs requêtes simultanées à partir des clients. Augmenter le nombre de clients et de requêtes jusqu'à instabilité ou défaillance (TCP) * Exécuter un grand nombre de parties en réseau simultanées. Analyser la fluidité avec de plus en plus de parties jusqu'à l'obtention de pertes de synchronisation ou défaillances.(UDP) |
| Critère de complétion: | Ici, on doit simplement s'assurer que les points critiques obtenus sont définitivement au-dessus des requis inscrits dans le SRS |
| Considérations spéciales: | Il sera intéressant d'augmenter la charge de stress pour atteindre les points critique du serveur et éventuellement développer des mesures préventives contre les "plantages" |

### Tests de volume

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que le logiciel fonctionne correctement lorsque beaucoup de clients se connectent au serveur et quand la base de données reçoit un nombre élevé de requêtes en même temps. |
| Technique: | Serveur   * Il faut d’abord se connecter au serveur avec plusieurs clients et exécuter des fonctions demandantes envers le serveur sur tous les clients. * S’assurer que le serveur répond bien à une charge élevée et que les informations sont toujours bien relayées entre les clients et le serveur.   Base de données   * Se connecter d’abord au serveur avec plusieurs clients puis exécuter des fonctions qui utilisent la base de données sur chaque client. * Vérifier que la base de données répond bien lorsque plusieurs requêtes sont exécutées simultanément et quand la taille de la base de données devient grande. |
| Critère de complétion: | Il faut s’assurer que le logiciel fonctionne de façon normale lorsque les niveaux de charges maximum tel qu’inscrit dans le SRS sont atteints. |
| Considérations spéciales: |  |

### Tests de sécurité et de contrôle d’accès

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que les utilisateurs ont seulement accès aux fonctions auxquelles ils devraient avoir accès tel que défini dans le SRS. S’assurer que le système et les applications ne sont accessibles que par les acteurs qui y ont droit (tel que défini dans le SRS). |
| Technique: | Niveau applicatif   * Démarrer l’application en tant qu’un acteur donné. * Vérifier que l’on n’a accès qu’aux fonctions tel que défini dans le SRS. * Répéter pour tous les acteurs/types d’utilisateurs.   Niveau système   * Il suffit de s’assurer que seuls les utilisateurs ayant le droit d’accéder au système peuvent y accéder. |
| Critère de complétion: | Tous les acteurs n’ont accès qu’aux fonctions auxquels ils ont le droit d’accéder et seul les acteurs ayant le droit d’accéder au système y ont accès. |
| Considérations spéciales: | Les tests niveau système ne sont probablement pas nécessaires pour le cas présent. |

### Tests d’échec/récupération

|  |  |
| --- | --- |
| Objectif de test: | S’assurer que le logiciel se comporte de façon adéquate lors d’une perte de connexion, d’une défaillance matérielle ou logicielle. C'est-à-dire que les autres clients ne sont pas trop affectés par cette défaillance et que les données ne sont pas corrompues. |
| Technique: | Réseau   * Connecter deux clients et débuter une partie, puis arrêter le serveur * Connecter plusieurs clients. Certains seront dans le lobby et d’autres seront en train de jouer, puis arrêter le serveur * Connecter un client qui modifiera son profil, puis arrêter le serveur   Client   * Connecter deux clients et débuter une partie, puis quitter un client de façon inappropriée (tuer le processus ou fermer la fenêtre) * Connecter deux clients et débuter une partie, puis fermer une seule connexion * Connecter deux clients et débuter une partie, puis un des clients appuie sur le bouton *quitter* de l’interface usager   À chaque étape, nous allons aussi vérifier si les données de l’utilisateur n’ont pas été corrompues. De plus, nous allons vérifier si les messages d’erreur sont compréhensibles pour l’utilisateur. |
| Critère de complétion: | Le client qui a une défaillance doit avoir tous les messages d’erreur et qu’ils soient compréhensibles. De plus, le client fautif ne doit pas affecter les autres clients. |
| Considérations spéciales: |  |

### 

## Outils

Les outils suivants seront utilisés au sein de la discipline de test:

|  |  |
| --- | --- |
| **Type de test** | **Outil** |
| Test de charge | Task Manager,  SRS |
| Tests d’interface usager | SRS |
| Tests d’intégrité des données | Visual Studio |
| Tests de performance | - |
| Tests de stress | Instruments (Xcode),  Task Manager,  SRS |
| Tests de volume | Visual Studio |
| Tests de sécurité et de contrôle d’accès | Visual Studio |
| Tests d’échec/récupération | Task Manager |
| Tests de fonction | - |

# Ressources

## Équipe de test

|  |  |  |
| --- | --- | --- |
| **Rôle** | **Membre de l’équipe** | **Responsabilités** |
| Testeur en chef  (client lourd) | Alexandre | Planifier et distribuer les tests. Exécuter des tests fonctionnels sur le client lourd |
| Testeur  (client léger) | Philippe | Exécuter les tests fonctionnels sur le client léger |
| Testeur  (client lourd, server) | Simon | Exécuter les tests concernant la communication client lourd - serveur. |
| Testeur  (client lourd) | Félix | Exécuter les tests non fonctionnels (fiabilité, mémoire, performance, etc.) sur le client lourd. |
| Testeur  (client léger, server) | François | Exécuter les tests non fonctionnels sur le client lourd et les tests concernant la communication iPad - serveur |
| Testeur  (client lourd) | Émile | Exécuter des tests fonctionnels sur le client lourd |

## Système

### Client Lourd

Les tests du client lourd et du server seront exécutés sur les ordinateurs d'évaluation, c'est-à-dire ceux du laboratoire de l'école. En cas de besoin, les ordinateurs personnels des développeurs seront utilisés également. Les machines de tests devront être configurées avec les éléments suivant:

* Windows 7 (préféré), XP ou Windows 8
* 2 Go de mémoire vive sous XP, 4Go sous Windows 7 ou Windows 8 (8Go préféré)
* Visual Studio 2010 ou 2012
* Microsoft SQL Server 2005 (préféré) ou 2008

### Client Léger

Les tests du léger seront exécutés sur l'iPad 2 exécutant iOS 6 fourni par l'école et utiliseront également les ordinateurs du laboratoire. En cas de besoin, les ordinateurs personnels des développeurs seront utilisés également.

# Jalons du projet

|  |  |  |  |
| --- | --- | --- | --- |
| **Jalon** | **Effort** | **Date de début** | **Date de fin** |
| Plan de tests | 5h | 01/04/13 | 02/04/13 |
| Design de tests | 5h | 02/04/13 | 03/04/13 |
| Exécution de tests | 10h | 03/04/13 | 05/04/13 |
| Évaluation de tests | 5h | 05/04/13 | 07/04/13 |