|  |
| --- |
| INSA LYON || IF-3-Gl&UML |
| TP Génie Logiciel 2017/2018 |
| Cahier de spécifications des besoins |
|  |
| **CHRISTOPHE ETIENNE | LYNN GHANDOUR | MUSTAFA COREKCI** |
|  |
|  |
| **04/04/2018** |



TP1 Séance : 30/03/2018

**Introduction (présentation TRES rapide du cahier des charges puis explication résumé de ce que fait notre application)**

Le client pour lequel nous réalisons l’application est un organisme gouvernemental en charge du dépistage de plusieurs maladies. Il souhaite analyser une nouvelle empreinte ou un ensemble d’empreintes …

Préciser ici aussi ce qu’on a décidé de faire ; et ce qu’on ne fait pas ?

**Glossaire (A revoir le titre ou si il y a une partie clairement définie pour sa)**

Fichier JSON : Fichier avec l’extension «.txt» contenant des informations au format JSON.

Dictionnaire : Le dictionnaire est le nom donné au fichier rentré par l’utilisateur lors du démarrage de l’application ou durant son exécution. Les données du fichier sont au format JSON et contient toutes les paires « maladie/empreinte» que nous stockeront dans une structure de données et sur lesquels se baseront nos analyses. Une maladie peut posséder plusieurs empreintes mais dans ce cas, il y autant de paire que d’empreinte correspondante.

Exemple : Mettre plusieurs exemples sur le format des données JSON.

Fichier contenant les empreintes : Ce fichier contient l’ensemble des empreintes à analyser que nous en ayons une seule ou plusieurs. Il est au format JSON et rentré par l’utilisateur.

Fichier de configuration : Ce fichier contient un ensemble d’attribut permettant de créer aléatoirement des empreintes pour les tests.

Empreinte : Une empreinte correspond à un ensemble d’attribut. Elle est associée à une seule personne. Dans le cas où plusieurs empreintes sont analysées, chaque empreinte correspond à une personne différente et par conséquent est traitée séparément.

Attribut : Dans notre application, un attribut est identifié par une lettre suivie d’un chiffre (c’est une idée que j’ai eu, à voir si on change de solution ou non, ça permet d’en avoir beaucoup si on veut. A voir la notation en fonction du nombre d’attribut que notre application peut supporter).

Exemple : s1 ; a2 ; c6

Correspondance : Deux attributs correspondent s’ils sont identiques. Deux empreintes correspondent selon un certain pourcentage défini ci-après.

Probabilité d’avoir une maladie : C’est un pourcentage calculé à partir de la formule suivante : nombre d’attribut de l’empreinte à analysé divisé par le nombre d’attribut total de l’empreinte liée à la maladie. Nous multiplions ensuite ce résultat par 100 pour l’obtenir sous forme de pourcentage.

Catégories de probabilité : Nous avons classés les risques d’avoir une maladie dans plusieurs catégories permettant au client d’avoir une meilleure précision du risque :

* + - 100% des attributs : Risque sûr
    - Entre 70% et 100% exclus : très probable
    - Entre 50% et 70% exclus : probable (à considérer)
    - Entre 20% et 50% exclus : très peu probable
    - Entre 0% et 20% exclus : Pas de risque

Résultat : Un résultat correspond à un ensemble de risque de maladie possible suite à l’analyse d’une empreinte.

**Acteurs**

Dans notre projet, deux acteurs rentrent en jeu. D’une part, le client qui se sert de l’application afin d’analyser des empreintes et recevoir un résultat sur les différents risques de maladies possible. Le client fournit à l’application l’empreinte ou les empreintes qu’ils souhaitent analyser mais aussi le dictionnaire des maladies liées à leur empreinte. D’autres part, le serveur qui charge l’empreinte et le dictionnaire puis effectue des requêtes pour trouver une correspondance entre l’empreinte fournie et les risques possibles de maladies.

**Spécification des besoins**

Pour une spec obligatoire : utiliser « devra » et pour une spec non obligatoire mais désirable : « sera »

1. **Besoins fonctionnels (Peut-être dispatcher en catégories encore plus précises ou à l’inverse regroupé)**
2. **Analyse d’une empreinte ou d’un ensemble d’empreinte**
   1. Le système devra **détecter** les risques de maladies liés à une nouvelle empreinte ou un ensemble d’empreintes. Le nombre d’empreinte à analysé est renseigné par l’utilisateur.
   2. Le système devra analyser les empreintes à partir d’un fichier donné par l’utilisateur.
   3. Le système devra analyser l’empreinte en comparant les attributs de celle-ci avec l’ensemble des attributs des empreintes liées aux maladies.
   4. Le système devra évaluer la probabilité du risque d’avoir une maladie pour l’analyse d’une empreinte.
3. **Affichage des résultats de l’analyse d’une empreinte**
   1. Le système devra retourner après analyse d’une empreinte un affichage des 10 risques de maladies les plus probables dans l’ordre décroissant.
   2. Le système devra afficher tous les risques trouvés en cas de nombre de risques détectés inférieurs à 10.
   3. Le système devra retourner le message suivant : « Aucun risque de maladie détecté à partir des informations contenues dans notre base de données » s’il y a 0 risque détecté.
   4. Le système devra afficher tous les risques à égalité si pour plusieurs maladies les probabilités sont identiques et ceux même si cela entraine un affichage de plus de 10 résultats.
   5. Le système devra afficher une phrase supplémentaire : « Une analyse supplémentaire pour vérification est conseillée. » en cas de probabilité d’avoir une maladie supérieure à 70%.
   6. Le système ne devra pas considérer comme un risque une probabilité d’avoir une maladie inférieure à 20%.
   7. Le système devra afficher une seule fois une maladie si celle-ci est détectée plusieurs fois dans les 10 maladies les plus probables.
   8. Le système devra permettre d’afficher un seul et unique résultat, celui avec la probabilité la plus élevée. En cas d’égalité, le système devra afficher tous les risques dont la probabilité est identique.
   9. Le système devra permettre d’afficher la liste de toutes les maladies contenues dans l’unité de stockage par ordre alphabétique.
4. **Affichage des détails liés à une maladie**
   1. Le système devra permettre d’afficher les attributs d’une empreinte liés à une maladie.
   2. Le système devra permettre d’afficher l’ensemble des maladies et la totalité de ses attributs (sans doublon) si celle-ci possède plusieurs empreintes différentes.
5. **Lecture du dictionnaire : fichier contenant les informations « maladie/empreinte»**
   1. Le système devra lire un fichier JSON donné par l’utilisateur et stocker les données qu’il contient comme base pour les analyses d’empreinte.
   2. Le système devra vérifier la validité du fichier. Il devra s’assurer que c’est un fichier au format JSON.
   3. Le système ne devra pas vérifier la validité du format des données à l’intérieur du document. Si celui-ci ne convient pas à celui renseigné (indiquer la partie où on le renseigne : spec non fonctionnelle je pense) alors les résultats éventuellement renvoyés par le système ne seront pas garantis. 🡪 A préciser
   4. Le système devra supprimer les données stockées en fin d’exécution de l’application. (Préciser le terme fin d’utilisation)
6. **Lecture d’une ou plusieurs empreintes : fichier JSON contenant les informations des empreintes à analyser** 
   1. Le système devra lire un fichier JSON contenant les informations des empreintes et de leurs attributs.
   2. Le système devra vérifier la validité du fichier en s’assurant que c’est un fichier au format JSON.
   3. Le système ne devra pas vérifier la validité du format des données à l’intérieur du document. Si celui-ci ne convient pas à celui renseigné (indiquer la partie où on le renseigne : spec non fonctionnelle je pense) alors les résultats renvoyés par le système ne seront pas garantis. 🡪 A préciser
   4. Le système devra vérifier si le fichier est vide. Dans ce cas, un message d’erreur devra être renvoyé à l’utilisateur.
7. **Ajout d’informations supplémentaires de maladie associée à leur empreinte**
   1. Le système devra permettre d’ajouter, à partir d’un fichier JSON (chemin contenant le fichier indiqué par l’utilisateur), des données supplémentaires dans la structure de données de stockage.
   2. Le système ne devra pas permettre l’ajout de doublon (maladie identique et attribut de l’empreinte correspondante identique en nombre et valeur).
8. **Besoins non fonctionnels**
9. **Performance** (A définir plus précisément)

Le système devra être capable de gérer une capacité de 1000 empreintes à analyser comportant moins de 15 attributs et d’effectuer cette analyse de manière rapide.

Le système devra être capable de gérer une capacité de 1000 paires  « maladie-empreinte » dans sa structure de stockage.

Le système devra être capable de gérer une analyse de 100 empreintes contenant 40 attributs.

1. **Evolutivité et maintenabilité**

Le système devra être évolutif, c’est-à-dire tolérer les modifications futures sur le plan dimensionnel, fonctionnel et matériel. Pour ce faire, le système doit être facilement modifiable. **On le laisse ?**

1. **Généricité**
2. **Environnement de programmation**

Le système devra être développé sous l’environnement de programmation Visual Studio en langage C++.

1. **Ergonomie**

Le système devra comporter une saisie clavier en guise d’IHM pour renseigner les données et choix nécessaires et afficher les résultats.

1. **Traçabilité**

Le système ne devra pas conserver les données stockées du dictionnaire après fin de l’exécution de l’application. De même, une fois les résultats des analyses affichées, le système ne devra pas les stocker.

1. **Sécurité**

Le système ne devra pas sécurisé les informations des différentes empreintes analysées ou des dictionnaires stockées. Il ne devra pas authentifier l’utilisateur, vérifier l’intégrité des données ou chiffrer celles-ci. (En effet, par souci de temps, de priorité et de complexité, nous ne nous attarderons pas sur ce point)

1. **Organisation**

La gestion de projet devra se faire selon le modèle de gestion de projet « Cycle en V ».

**Spécifications produit :**

* **Sécurité : Les données**
* **Disponibilité : le type/moyen de stockage**
* **Dépendances : le type/moyen de stockage**
* **Performances : Capacité**

**Spécifications organisationnelles :**

* **Spécifications de l’environnement : IDE**
* **Spécification opérationnel : Cycle en V**
* **Spécification de développement : langage de programmation**

**Spécifications externes :**

* **Spécifications législatives**
  + **Confidentialité des informations et informations fournies aux bonnes personnes**

**Préciser aussi les affichages en fonction de la requête et**

**Plan de tests et validation**

1. **Tests fonctionnels**

|  |  |  |
| --- | --- | --- |
| **N°Test** | **Cas considéré** | **Réponse attendue** |
| 1 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir 10 risques. | Un affichage par ordre décroissant de probabilité des 10 risques principaux détectés pour cette empreinte. |
| 2 | Un fichier JSON contenant plusieurs empreintes valides et un dictionnaire valide permettant d’obtenir 10 risques pour chaque empreinte. | Un affichage par ordre décroissant de probabilité des 10 risques principaux détectés pour chaque empreinte. |
| 3 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir moins de 10 risques. | Un affichage par ordre décroissant de probabilité des risques détectés pour cette empreinte (nombre de risque supérieur à 0 et inférieur à 10). |
| 4 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir moins de 10 risques pour chaque empreinte. | Un affichage par ordre décroissant de probabilité des risques détectés pour chaque empreinte (avec un nombre de risque supérieur à 0 et inférieur à 10). |
| 5 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir 0 risque de maladie correspondant à l’empreinte. | Aucun risque n’est affiché, seulement la phrase d’indication : « Aucun risque de maladie détecté à partir des informations contenues dans notre base de données » |
| 6 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir une égalité et par conséquent, plus de 10 risques de maladie correspondant à l’empreinte. | Un affichage par ordre décroissant de probabilité de plus de 10 risques. |
| 7 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Une demande d’affichage des détails d’une maladie (n’ayant qu’une empreinte). | Un affichage de la totalité des attributs contenus dans l’empreinte liée à la maladie demandée. |
| 8 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Une demande d’affichage des détails d’une maladie (ayant plusieurs empreintes). | Un affichage de la totalité des attributs contenus dans les différentes empreintes liées à la maladie demandée (sans doublon). |
| 8 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Une demande d’affichage des détails d’une maladie ne se trouvant pas dans le dictionnaire. | Affichage d’un message d’information : « Cette maladie n’est pas renseignée dans notre unité de stockage. » |
| 9 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide permettant d’obtenir au moins un risque dont la probabilité d’avoir la maladie est supérieur à 70%. | Un affichage classique par ordre décroissant de probabilité des risques principaux détectés pour cette empreinte. Pour le risque supérieur à 70%, une précision supplémentaire : « Une analyse supplémentaire pour vérification est conseillée. » |
| 10 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Affichage du risque le plus probable seulement (pas de cas d’égalité). | Un affichage d’un seul risque de maladie avec les informations classiques. |
| 11 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Affichage du risque le plus probable seulement (cas d’égalité). | Un affichage des risques de maladies les plus probables étant à égalité avec les informations classiques associées. |
| 12 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Affichage du risque le plus probable seulement (pas de cas d’égalité). Vérification avec le risque le plus probable ayant une probabilité < 20%. | Aucun risque n’est affiché. Seulement la phrase d’indication : « Aucun risque de maladie détecté à partir des informations contenues dans notre base de données » |
| 13 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Insertion de maladies supplémentaires dans la base données à partir d’un autre dictionnaire valide (pas de doublon). | Toutes les maladies du nouveau dictionnaire et leur empreinte associée sont ajoutées dans notre unité de stockage. |
| 14 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Insertion de maladies supplémentaires dans la base données à partir d’un autre dictionnaire valide (contient des doublons de maladies déjà stockées). | Toutes les maladies du nouveau dictionnaire et leur empreinte associée sont ajoutées dans notre unité de stockage excepté celles déjà présentes. |
| 15 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide. Demande d’affichage de l’ensemble des maladies contenues dans notre unité de stockage. | Toutes les maladies sont affichées suivant l’ordre où elles sont stockés. |
| 16 | Un fichier JSON contenant une empreinte valide et un dictionnaire valide mais vide. Demande d’affichage de l’ensemble des maladies contenues dans notre unité de stockage. | Aucune maladie affichée, seulement une phrase d’indication : « Le dictionnaire ne contient aucune maladie. » |
| 17 | Un dictionnaire non valide (extension différente de celle attendue) (au démarrage et/ou au moment de l’insertion de nouvelles données). | Retourne une erreur précisant que l’extension du dictionnaire rentrée est invalide. |
| 18 | Un fichier JSON non valide et un dictionnaire valide (extension différente de celle attendue). | Retourne une erreur précisant que l’extension du fichier censé contenir le ou les empreintes est invalide. |
| 19 | Un fichier JSON non valide (fichier vide) et un dictionnaire valide. | Message d’erreur affiché : « Aucune empreinte à analysé dans le fichier donné. » |

1. **Tests de performances**

|  |  |  |
| --- | --- | --- |
| 19 | Un fichier JSON valide contenant 100 empreintes et 40 attributs et un dictionnaire valide. | Temps de réponse court et affichage des résultats pour les 100 empreintes. |
| 20 | Un fichier JSON valide contenant 1000 empreintes et 15 attributs et un dictionnaire valide. | Temps de réponse court et affichage des résultats pour les 100 empreintes. |
| 21 | Un fichier JSON valide contenant 100 empreintes et 40 attributs et un dictionnaire valide contenant 1000 maladies. | Stockage des 1000 maladies dans l’unité de stockage. Temps de réponse court et affichage des résultats pour les 100 empreintes. |
| 22 | Un fichier JSON valide contenant 1000 empreintes et 15 attributs et un dictionnaire valide contenant 1000 maladies. | Stockage des 1000 maladies dans l’unité de stockage. Temps de réponse court et affichage des résultats pour les 1000 empreintes. |
| 23 |  |  |

1. **Tests d’utilisabilité**

Le client vérifie si le système est ergonomique, autrement dit si son utilisation est instinctive (la saisie clavier).

1. **Tests d’exploitation**

Nous n’avons pas traité ce point la dans les spécifications donc aucun test n’est prévu à cet effet.

1. **Tests de sécurité**

Nous n’avons pas traité ce point la dans les spécifications donc aucun test n’est prévu à cet effet.

1. **Tests de sureté**

|  |  |  |
| --- | --- | --- |
| 6.1 | Vérification que le dictionnaire est vide au démarrage de l’application (avec un dictionnaire remplie à l’exécution précédente). | Message informant que le dictionnaire est vide. |

**Description des interfaces**

Forme de l’affichage classique d’un risque (X représentant sa place dans le classement) :

X. Grippe : 64% de risque - Probable

Forme de l’affichage classique de plusieurs risques :

1. Grippe : 68% de chance - Probable

2. Rhinopharyngite : 63% de chance - Probable

3. Angine blanche : 52% de chance - Probable

4. Angine rouge : 44% de chance – Peu probable

5. Grippe aviaire : 21% de chance – Peu probable

Forme de l’affichage pour 0 risque détecté :

Aucun risque de maladie détecté à partir des informations contenues dans notre base de données.

Forme de l’affichage pour un risque ayant une probabilité supérieure à 70%:

X. Grippe : 88% de chance – Très probable**.**

Une analyse supplémentaire pour vérification est conseillée.

Forme de l’affichage pour l’ensemble des maladies du dictionnaire :

Dyslexie

Grippe

Grippe Aviaire

Rhume

Forme de l’affichage pour les détails d’une maladie

Grippe: a1 / b2 / c10 / e2 / g7

Affichage du menu de saisie des options possibles

* Renseignez le dictionnaire (Chemin jusqu’au fichier texte contenant les données au format JSON)
* 1. Analyser une ou des empreintes : Affichage des 10 maladies les plus probables
* 2. Analyser une ou des empreintes : Affichage de la maladie la plus probable
* 3. Afficher les maladies du dictionnaire
* 4. Afficher le détail d’une maladie
* 5. Insérer des données supplémentaires dans le dictionnaire
* 6. Quitter l’application

Si 1 ou 2 :

* Renseignez le fichier contenant les empreintes à analyser (Chemin jusqu’au fichier texte contenant les données au format JSON)

Si 4 :

* Quelle maladie souhaitez-vous consulter ?

Si 5 :

* Renseignez le dictionnaire (Chemin jusqu’au fichier texte contenant les données au format JSON)

**Manuel utilisateur (ébauche)**

analyse\_medicale(1) Manuel d'utilisation de analyse\_medicale analyse\_medicale(1)

NOM

analyse\_medicale - Analyse un fichier d’empreinte (analyse médicale)

SYNOPSIS

./analyse\_medicale

DESCRIPTION

analyse\_medicale traite un fichier d’empreintes (renseigné par l’utilisateur) à partir d’un dictionnaire (renseigné par l’utilisateur). Le fichier d’empreinte et le dictionnaire sont au format « .txt » et contiennent des données au format JSON. analyse\_medicale analyse le ou les empreintes en les comparants à celles liées aux maladies puis affiche un résultat des risques les plus probables d’avoir la maladie. Compléter la description

AUTHOR

Binôme B3142:

Corekci Mustafa

Ghandour Lynn

Etienne Christophe

**Planning: Version 1**

|  |  |
| --- | --- |
| **Date** | **Objectifs** |
| 30 mars 2018 | Séance de TP numéro 1  Découverte du sujet – Spécifications définies – Objectifs mis en place |
| 4 avril 2018 | Rendu d’une première version du livrable attendue pour le TP1 :   * Dossier d’analyse contenant les spécifications fonctionnelles et non-fonctionnelles du système * Plan de tests de validation/recette (format textuelle) * Description des interfaces éventuelles * Manuel utilisateur (ébauche) * Planning contenant les jalons importants déterminé |
| 22 avril 2018 | Livrable numéro 1 modifié et terminé après retour des professeurs. |
| 25 avril 2018 | Séance de TP du numéro 2  Rendu du livrable numéro 1  Découverte de la phase numéro 2 du TP – Réflexion et avancement sur les différents points du livrable à réaliser – Détermination des objectifs et répartition entre les membres du trinôme pour la prochaine séance – Ajustement du planning en conséquence des avancées. |
| 25 avril 2018 – 2 mai 2018 | Avancement sur l’élaboration du livrable numéro 2. |
| 2 mai 2018 | Séance de TP numéro 3  Livrable numéro 2 terminé :   * Dossier de conception complété * Plan de test d’intégration rédigé * Première version des tests unitaires rédigée * Planning ajusté   Découverte de la phase numéro 3 du TP – Début de développement de l’application - Ajustement du planning en conséquence des avancées. |
| 2 mai 2018 - 23 mai 2018 | Avancement de la phase de développement et réalisation des tests. |
| 23 mai 2018 | Séance de TP numéro 4  Dernière modification du code concernant les détails – Réalisation de la totalité des tests – Début de la rédaction du livrable final. |
| 23 mai 2018 - 30 mai 2018 | Finition du livrable final. |
| 30 mai 2018 | Séance de validation. |