

**Le logiciel ScoutLog**

**Pour la flotte Scout**

***Document confidentiel***

**Auteur : R. Daneel Olivaw**

**Version : 1.0**

**Date : 19 janvier 2021**

C’est la fin de la vie sur Terre…

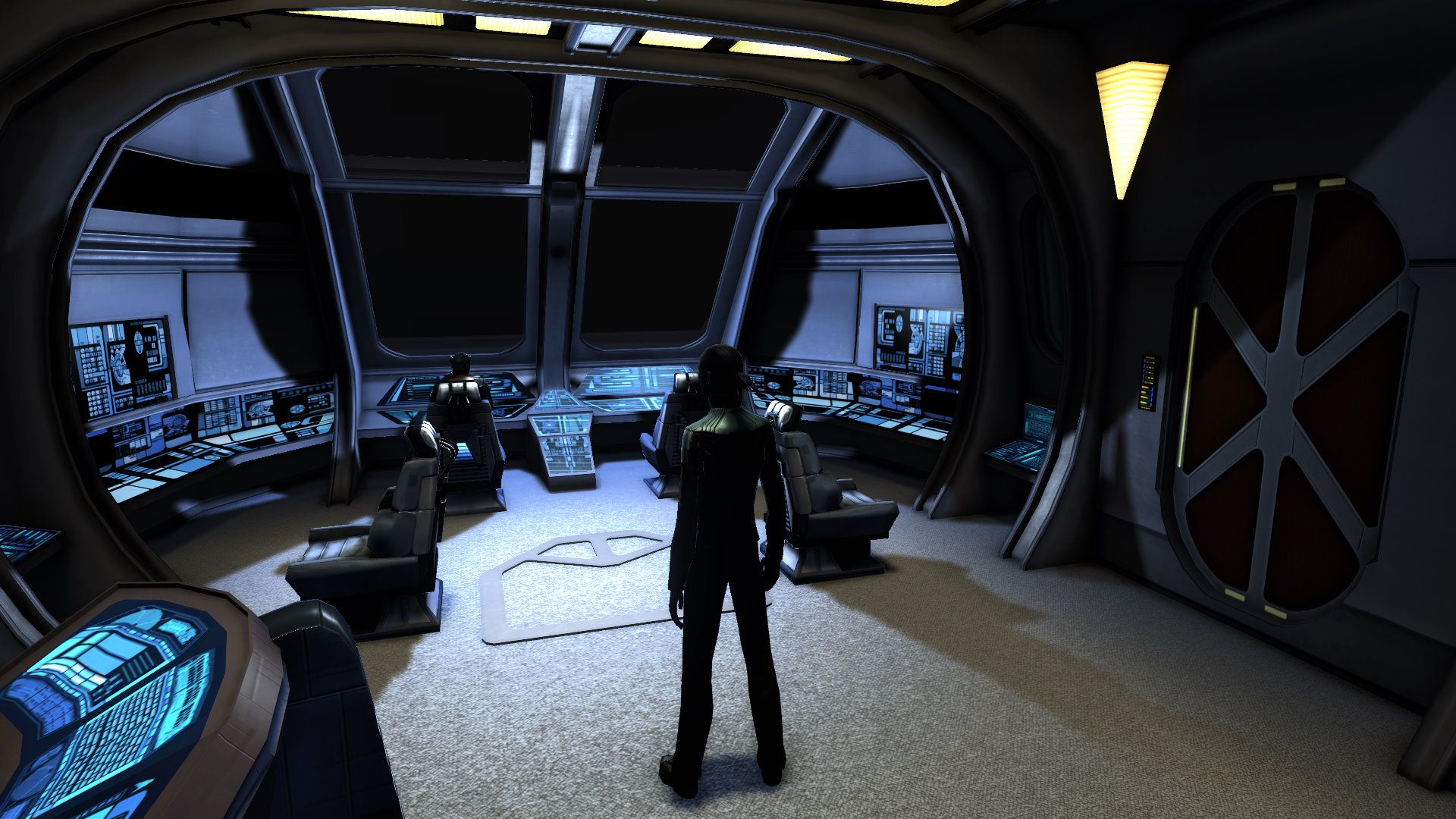


C’est l’année 2098 et il n’y a plus de ressources sur Terre. Tout a été utilisé jusqu’à épuisement. La surpopulation, la mauvaise gestion des matières premières et la révolte des geeks de 2050 ont mortellement affaibli notre planète. Malgré les investissements et tous les efforts des dernières années, la colonie sur Mars a échoué et a été laissée à l’abandon. Les citoyens des 12 pays commencent à désespérer et il devient indéniable qu’il faut se tourner vers l’espace afin d’y trouver une planète habitable. Heureusement, un regroupement des plus grands scientifiques de la planète y travaillent déjà et de nouveaux vaisseaux permettant la découverte et la colonisation de nouvelles planètes habitables sont déjà en construction. Ces nouveaux modèles de vaisseaux se nommeront « Scouts ». Ils sont conçus pour l’exploration et peuvent loger un grand nombre de personnes.

## Un nouvel espoir!

Cette flotte de vaisseaux (nommée « Scout Fleet ») est pratiquement prête et tous les citoyens qui le désirent pourront y embarquer. Actuellement, tous les gouvernements, le conseil général de la planète ainsi que la colonie de Solaria optent pour quitter la Terre à bord de ces vaisseaux. Ceux qui voudront rester sur Terre seront laissés à eux-mêmes.

Pour diriger la flotte et guider l’exploration, le commandant désigné ne sera nul autre que R. Daneel Olivaw. Au moment d’écrire ces mots, M. Olivaw se trouve dans le laboratoire souterrain en train d’analyser la première galaxie à explorer.



Pourtant, les défis restent nombreux. Une équipe est actuellement en train de développer une intelligence artificielle qui sera installée dans chaque vaisseau. Celle-ci servira d’aide et de ressource pour la prise de décision de chaque vaisseau. Une autre équipe doit développer un logiciel facilitant l’exploration et l’écriture d’un journal de bord.

On vous a chargé de créer ce logiciel, nommé ScoutLog. L’humanité dépend de vous!

## Le logiciel ScoutLog 1.0

Cette application servira à conserver un journal de bord et à répertorier les planètes visitées.

### Module 1 : le journal de bord (*Mongodb?*)

#### 1.1 Entrée du journal

Chaque jour, le commandant du vaisseau se doit de créer une nouvelle entrée dans le journal de bord du vaisseau. Une entrée dans le journal de bord contient toujours ces informations :

* Date (unique)
* Statut de la journée : Normal, Anormal, Exploration
* Créateur de l’entrée (nom du commandant)

Dans le cas où le statut est « Anormal », les informations suivantes doivent être ajoutées :

* Raison(s)

Dans le cas où le statut est « Exploration », ce qui veut dire que le vaisseau a exploré une planète, il faut ajouter :

* Le nom de la planète
* Si elle est habitable ou pas
* Le nom de sa galaxie
* Le nom de la ou des planètes à proximité (parmi celles déjà explorées)
* Une photo de la planète (qui sera affichée lorsqu’on la visionne)

Veuillez noter que toutes ces informations sont obligatoires, sauf le nom des planètes à proximité. Aussi, il est impossible que 2 planètes possèdent le même nom.

#### 1.2 Visionnement des entrées

Une section du logiciel doit permettre de voir les entrées du journal de bord. Une page afficherait une entrée du journal de bord, et il serait possible de choisir une autre entrée via le bouton « Suivant » ou « Précédent ». La dernière entrée du journal est affichée en premier (selon la date). En cliquant sur « Précédent », l’entrée précédente est affichée, etc.

Cette section possèdera également un bouton « Supprimer », permettant de supprimer une entrée du journal de bord.

### Module 2 : le répertoire des planètes et les données du système

Afin de mieux diriger l’exploration de la flotte, le logiciel doit être en mesure de fournir certaines informations importantes sur les planètes :

#### 2.1 Trajectoire (*neo4j?*)

Ce sous-module permet de fournir à l’IA du vaisseau le chemin pour arriver à une planète destination. Le chemin est constitué d’une liste de planètes.

Par exemple, si l’on demande le chemin entre la planète « Terre » à la planète « Aurora », le logiciel sortirait : « Terre » 🡪 « Mars » 🡪 « Pendora » 🡪 « Aurora ». Ce qui signifie qu’il faudrait, en partant de la Terre, se diriger vers la planète Mars, puis Pendora, pour arriver ensuite à Aurora.

#### 2.2 L’analyse des galaxies

2.2.1 La galaxie la plus prometteuse

Étant donné le nombre de planètes et de galaxies qui seront explorées, il est nécessaire que le logiciel puisse savoir quelle galaxie possède le plus grand nombre de planètes habitables.

2.2.2 La galaxie ayant le plus grand nombre de planètes explorées

Afin d’éviter de visiter trop de fois les mêmes endroits, le logiciel doit sortir par ordre descendant le nombre de planètes visitées par galaxie. Par exemple :

1. Triangle (11 planètes visitées)
2. Andromède (7 planètes visitées)
3. Cigare (5 planètes visitées)
4. Voie lactée (2 planètes visitées)

#### 2.4 Les statistiques générales

Une section du logiciel doit donner un aperçu des différentes données du système :

* Le nombre d’entrées dans le journal de bord
* La moyenne du nombre d’exploration sur le nombre total d’entrées dans le journal de bord
* Le nombre de planètes habitables (sur le nombre total de planètes explorées)
* Le nombre de photos sauvegardées (doit être égal au nombre de planètes explorées)
* Le nom des dernières planètes visitées

# Informations supplémentaires

* L’application doit être codée en Java (squelette fourni).
* L’application doit utiliser avec BerkeleyDB pour les photos, et soit MongoDB ou Neo4j pour le reste.
* Le logiciel doit fonctionner sur le poste de travail du professeur sans installation de librairies supplémentaires
* Le projet doit être fait en équipe de 2 au maximum
* Chaque jour de retard entraîne 10% de pénalité sur la note de ce travail
* Si vous utilisez du code pris sur Internet, veuillez citer la source en commentaire, autrement cela pourrait être considéré comme du plagiat.
* Il faut utiliser git pour faire la gestion du code source (dépôt **privé**!!). Le fichier .gitignore a déjà été créé.
* Même s’il n’y a pas beaucoup de données dans vos bases de données, créez les index aux bons endroits comme si le logiciel avait un très grand nombre de données.

### README.txt

Un fichier README.txt doit être remis avec votre projet. Celui-ci doit contenir :

1- Le nom des coéquipiers

2- Quelle base de données sera utilisée pour quel type d’informations (environ 50-150 mots). La justification de votre choix est très importante.

3- Où sont les index, leur type (unique ou pas) et leur ordre (si applicable)?

### Squelette du programme disponible

Une base du programme a déjà été conçue. Celle-ci intègre déjà une première interface graphique et possède les librairies pour communiquer avec BerkeleyDB (*embedded*), Neo4j et MongoDB.

### Les étapes de développement

Contrairement au laboratoire et projet, il n’y a pas d’étapes à suivre. Je vous conseille fortement de débuter par LogDAO.addLog(), puis les autres méthodes vides dans LogDAO.

## Échéancier et livrables

Le projet (le code de l’application complète ainsi que le dossier .git) doit être remis sur Léa.

## Grille de correction générale

|  |  |
| --- | --- |
|  |  |
| README.md et git   * Noms des coéquipiers * Justification du choix de la BD (environ entre 50-150 mots)   + Quel type de base de données est utilisé et pourquoi ? * Emplacement des index + type * Git : Historique des commits (avec messages pertinents) | % 15 |
| Fonctionnalité – insertion dans le journal de bord   * Création personnalisée en fonction des 3 types d’entrées (statut) | % 15 |
| Fonctionnalité – supprimer une entrée | % 5 |
| Fonctionnalité – suppression toutes les données | % 5 |
| Fonctionnalité – visionnement du journal de bord   * Boutons « suivant » et « précédent » fonctionnels | % 10 |
| Fonctionnalité – trajectoire | % 10 |
| Fonctionnalité – l’analyse des galaxies   * La plus prometteuse * Le plus grand nombre de planètes explorées | % 10 |
| Fonctionnalité – les statistiques | % 20 |
| Divers   * Qualité de la programmation, indentation, etc. | % 10 |
|  |  |
| Total | **% 100** |