



Knapsack ... le sac à dos de survie. ¹

Objectifs

Ce travail vise principalement à vous faire expérimenter les aspects suivants:

- Appliquer la méthode agile de gestion de projets, « scrum » pour mener à bien le développement d'un projet.
- Produire le sprint zéro.
- Concevoir une base de données.
- Concevoir la logique applicative.
- Concevoir un prototype visuel.
- Effectuer les tests unitaires, d'intégration et d'acceptation.
- Produire un document de conception.
- Travailler en équipe de 3 ou 4. Pas plus, pas moins.

Préambule :

En algorithmique, le problème du sac à dos, parfois noté (KP) (de l'anglais **Knapsack Problem**) est un problème d'optimisation combinatoire. Il modélise une situation analogue au remplissage d'un sac à dos, ne pouvant supporter plus d'un certain poids.

(https://fr.wikipedia.org/wiki/Probl%C3%A8me_du_sac_%C3%A0_dos)

Mise en situation

Nous sommes dans un environnement du jeu vidéo de type **Fallout** et nous souhaitons développer l'application **Knapsack** qui permet l'achat d'un certain nombre d'items concernant le jeu, la gestion des items pour le jeu et l'évaluation de l'efficacité d'un item.

¹ Source de l'image :<https://www.flickr.com/photos/playstationblog/3715830608/>

Dans l'environnement du jeu, il y a plusieurs joueurs. Chaque joueur peut acheter plusieurs items et des joueurs différents peuvent acheter des items identiques. La quantité d'un Item acheté peut être supérieure à 1. **Pour les achats, nous utiliserons un panier d'achat.**

Dans le jeu, les items achetés par le joueur sont transportés dans son sac à dos. **Il y a une contrainte sur le poids à transporter.**

Items du jeu :

1. Un Item du jeu a un numéro unique, un nom, une quantité en stock, un type d'item (armes, armures, médicaments, nourritures, ressources et munitions), un prix unitaire, **un poids** (en livres), l'utilité et une photo.
2. Les armes ont une efficacité, un genre (Arme lourdes, arme de combat, laser, arme à radiation etc...) et une description.
3. Les armures ont matière qui compose l'armure et la taille : exemple des gants, casques, etc.)
4. Les médicaments : (stimpack, Rad-x, sérum mystérieux etc...) ont un effet attendu, la durée pour l'effet et l'effet indésirable. Si aucun effet indésirable alors la valeur est : aucun.
5. La nourriture : Fourmis rôties, faux-filet, cuisse de poulet, bourbon, eau, etc.). La nourriture a un apport calorique, le composant nutritif principal (sucre, protéine, gras etc..) et le minéral principal : fer, calcium etc...
6. Les munitions, doivent indiquer sur quel type d'arme elles vont. Elles ont au moins le calibre.
7. Lorsque l'utilité d'un item est 1, alors cet item doit se retrouver dans votre sac à dos (il est obligatoire à votre survie)
8. Les médicaments et la nourriture ont une utilité qui est 1

Les joueurs :

1. Un joueur a un alias unique, un nom, un prénom et **un montant initial en caps** (capsules), une dextérité (en points), le nombre de points de vie et un poids maximal à transporter. Les joueurs sont identifiés par un numéro séquentiel. D'autres informations pourraient être ajoutées
2. Tous les joueurs peuvent acheter n'importe quel item
3. Un joueur ne peut pas transporter plus qu'un certain poids sinon il perd de la dextérité.
4. La dextérité ne doit jamais être en dessous d'une certaine valeur.

5. Régulièrement le joueur doit se nourrir et prendre des médicaments pour que le nombre de points de vie ne baisse pas.

Le sac à dos :

Chaque joueur a un sac à dos dans lequel il peut transporter ses items. Lorsque le sac à dos a atteint le poids maximum que le joueur peut transporter alors le joueur doit :

- Soit abandonner l'achat, modifier la liste des Items à acheter ou alors modifier la quantité.
- Supprimer un ou plusieurs items de son sac à dos afin de finaliser l'achat.
- Accepter de diminuer sa dextérité (car le joueur est plus lourd). Pour chaque livre dépassée le joueur perd un point de dextérité.
- Manger ou se soigner pour monter le nombre de points de vie.

Caps, pour les capsules :

Initialement tous les joueurs ont un nombre fixe de capsules (caps) qu'ils peuvent utiliser pour acheter des Items. Cependant, ils peuvent obtenir des caps grâce au moyens suivants :

- Faire une quête. Une quête consiste à résoudre une énigme. Chaque quête a un nombre de caps pour sa résolution.
- Demander à l'administrateur. L'administrateur augmente le nombre de caps avec un montant fixe. L'administrateur ne peut pas dépasser un montant maximal à donner pour chaque joueur.
- Vendre un item.

Indications concernant l'application d'achat de *Knapsack*

Lorsque la personne n'est pas un joueur inscrit (pas de nom usager ni de mot de passe), alors on peut consulter les Items selon les critères suivants:

1. Tous les Items : l'affichage doit être ordonné par type.
2. Par type Item : l'affichage doit se faire par poids, puis par prix. Ici on souhaite consulter la liste des Items selon le type. Par exemple, je veux juste afficher les armes.
3. En combinant plusieurs types : l'affichage doit être ordonné par type, puis par poids et enfin par prix. Ici on souhaite consulter la liste des Items selon des types combinés. Par exemple, je veux juste afficher les armes et les médicaments
4. Si aucun type n'est sélectionné, alors on doit pouvoir afficher tous les Items.

5. S'inscrire au site.

Si la personne est un joueur inscrit alors le joueur peut :

1. S'identifier avant d'acheter.
2. Faire un achat d'un ou plusieurs Items et mettre à jour la BD après un achat.
 - Lors d'un achat, on doit pouvoir ajouter, supprimer ou modifier la quantité d'un Item dans le « **panier**».
 - Lors d'un achat, on ne doit pas pouvoir acheter un Item avec une quantité N, si cette quantité est plus grande que la quantité en stock → équivalent à dire : Je ne peux pas acheter 20 stimpack s'il en existe juste 18.
 - Lors d'un achat, si le montant total de l'achat est plus élevé que le capital du joueur, alors il ne peut pas acheter à moins de modifier son panier ou d'augmenter son capital.
 - Avant de confirmer l'achat, si le sac à dos du joueur dépasse le poids requis alors le joueur doit prendre une décision
 - Abandonner l'achat.
 - Diminuer sa dextérité en acceptant l'achat.
 - Abandonner des items du sac à dos pour compléter l'achat.
3. Modifier son profil.
4. Consulter son sac à dos.
5. Évaluer et **commenter** un Item qu'il a acheté. Chaque joueur poste un commentaire sur un Item qu'il a déjà acheté et évalue l'Item en donnant une note de 1 à 5 pour : Efficacité.
6. Consulter les évaluations des Items sur le site Web et sur son appareil mobile. Nous souhaitons :
 - Afficher le nombre de personnes ayant attribué un nombre d'étoiles (exemple 3 personnes ont donné 4 étoiles, une personne a donné 3 étoiles, 2 personnes ont donné 2 étoiles).
 - Afficher la moyenne des évaluations. (Tel Item a en moyenne 3 étoiles...) et effectuer une recherche selon le nombre d'étoiles (exemple tous les Items avec le nombre d'étoiles 3 ou plus).

Indications concernant la partie administration de l'application *Knapsack*

Minimalement, l'application doit permettre :

- L'ajout d'un Item (nouvelle arme, nouvelle armure, etc.)
- Augmenter le capital du joueur.
- Supprimer un commentaire jugé inadéquat.

Enigma, pour les quêtes

Les quêtes pour augmenter le capital du joueur sont réalisées à l'aide de l'application **Enigma**. L'application doit fournir au joueur des énigmes à résoudre. Lorsque le joueur a réussi à résoudre l'énigme le capital du joueur est augmenté. Ce ne sont pas toutes les énigmes qui donnent le même montant. Avant de résoudre l'énigme, le joueur doit se connecter à l'application.

Les énigmes difficiles donnent 200 caps les énigmes de difficulté moyenne donnent 100 caps et les énigmes faciles donnent 50 caps. L'énigme à résoudre est tirée de manière aléatoire, mais le joueur pourrait décider de choisir la difficulté de l'énigme.

Lorsque le joueur a répondu successivement à 3 énigmes difficiles il gagne 1000 caps.

Dans **Enigma**, nous pouvons également garder un certain nombre de statistiques comme : le nombre d'énigmes résolues, le nombre d'énigme échouées etc...

Recommandation pour la conception

Pour **Knapsack**, vous avez le choix de conception entre LAMP (MySQL, PHP) ou ASP.NET (SQL Server et C#, ASP.NET). Dans tous les cas l'application doit être hébergée et il est fortement recommandé d'utiliser des procédures stockées pour votre base de données.

Le développement

Le développement est un développement agile SCRUM, donc le backlog est **VIVANT** et **le client peut apporter des changements en tout temps**.

La description de ce qui est attendu dans l'application peut être complétée par les clients concernés en tout temps. Il est de votre responsabilité de clarifier les éléments du backlog en tout temps.