École Nationale Supérieure d'Informatique pour l'Industie et l'Entreprise

22 mai 2018

Rapport de projet en développement web

Tales of Tiny Adventurers

Groupe T9

Laurie Pandraud, Maria Minko, Martin Bourdon, Jordan Aurey

Présentation du projet

L'objectif de ce projet a été de réaliser un jeu incrémental, ou idle game, sur navigateur. Ici, le but est de progresser en niveau à l'aide de quêtes, réparties par niveaux de difficulté. Le personnage pourra également intégrer une guilde et augmenter le prestige de celle-ci, se faire aider par un familier et acheter différents objets dans la boutique. Chaque quête dure un certain nombre de minutes en temps réel et n'est ouverte qu'aux personnages dont le niveau est suffisament élevé. Chaque utilisateur possède son propre compte protégé par un mot de passe et peut, à sa convenue, jouer un mage ou un guerrier, accompagné du familier de son choix parmi un chat, un mimic, une licorne, un dragon et un paresseux. A mesure qu'un personnage monte de niveau, ses caractéristiques augmentent et les quêtes deviennent plus simples. Chaque membre du groupe de projet, quant à lui, possède un compte administrateur permettant notamment de bannir des utilisateurs.

Les technologies utilisées pour ce projet sont PHP 7.0 pour l'implémentation des différentes fonctionalités, PostgreSQL pour la base de données et JavaScript pour la vérification des formulaires. Nous avons également utilisé PDO pour les accès à la base de données pour empêcher les injections SQL.

Exécution

1.1 Base de données

La base de données contient différentes tables. Pour plus de lisibilité, il a été choisi de séparer certaines tables en deux tables distinctes, ce qui explique le fait que certaines d'entre elles aient la même clé primaire.

Liste des tables :
`users(#username, character_name, password, status)` **: cette table sert à enregistrer les utilisateurs du site. Le champ** `status` **permet de distinguer les administrateurs du site (qui ont le droit d'exclure des utilisateurs) des utilisateurs réguliers.**
`perso(#username, character_name, gear, stamina, mana, strength, mind, sword_skill, staff_skill, pet, pet_xp, pet_lvl, guild, money, charac_lvl, charac_xp, lastquest, endtimequest)` **: cette table enregistre les données chiffrées associées à un perso, notamment ses caractéristiques qui détermineront les chances de succès d'une quête réalisée par le personnage.**
`guilde(#guild_name, guild_symbole, guild_points, guild_leader)` **: cette table sert à enregistrer les différentes guildes que le joueur peut intégrer.**
`adventure(#q_name, q_level, reward_money, reward_sword, reward_staff, reward_xp, q_time)` **: cette table permet d'enregistrer les informations relatives aux quêtes.**
`pet(#species)` **: c'est ici que sont enregistrés les différents familiers disponibles dans le jeu.**
`items(#item_name, price)` **: le jeu dispose d'un magasin, et c'est dans cette table que sont consignés les différents consommables.**
`inventory(#id, username, item_name, nb)` **: cette table permet de savoir quels consommables ont été achetés par le personnage.**

1.2 Page d'accueil du site

La page d'accueil contient les champs pour l'authentification qui vérifient si le nom d'utilisateur et le mot de passe sont corrects. Pour vérifier qu'elle fonctionnait correctement, il lui était d'abord demandé d'afficher simplement "vous êtes connecté" si le nom d'utilisateur et le mot de passe correspondaient à ceux de la base de données. Ceci a permis de se rendre compte de problèmes de correspondance d'identifiants. Une fois la connexion fonctionnelle, plusieurs fonctions ont été implémentées dans la page d'accueil afin de ne pas avoir besoin de les répéter lorsque l'utilisateur navigue d'une page à l'autre du site une fois connecté.

1.3 Inscription sur le site

Un bouton d'inscription a ensuite été ajouté à la page d'accueil. Il mène à une page contenant un champ pour le nom d'utilisateur, un champ pour le nom du personnage, un champ pour le mot de passe et un bouton de confirmation. Celui-ci appelle une fonction qui ajoute l'utilisateur à la table `users`, après avoir validé le formulaire avec JS.

1.4 Page du personnage

Une fois identifié, un utilisateur a accès à sa page personnelle contenant les informations relatives à son personnage. Le username est conservé en tant que variable de session afin d'accéder aux différentes fonctionnalités. Comme il y a potentiellement beaucoup de requêtes à effectuer, toutes les requêtes sont effectuées au début et leurs résultats sont également conservés dans des variables de session afin d'avoir les informations nécessaires en temps réel.
La page du personnage est divisée en plusieurs onglets. L'onglet "profil" permet de changer de mot de passe, l'arme équipée par le personnage, la photo de profil et de supprimer son compte.
La boutique permet d'acheter différents consommables pour faciliter la progression.
L'onglet du familier permet d'accéder aux points d'expérience de son familier si l'utilisateur en a un. En effet, jusqu'au niveau 5 n'apparait qu'un message disant que le niveau du personnage n'est pas suffisant. A partir du niveau 5, la première fois que l'onglet sera ouvert, l'utilisateur pourra choisir un familier parmi ceux de la table `pet`. Il sera alors renvoyé sur la page du personnage afin de laisser le temps à l'onglet de se mettre à jour.
L'onglet de quêtes affiche toutes les quêtes disponibles et permet d'en lancer une si le niveau du personnage est suffisant. Cependant, la description associée à chaque quête n'apparait que dans le fichier code et

non dans la base de données, il faut donc vérifier les quêtes une par une pour leur associer le bon descriptif ce qui peut être fastidieux.
L'onglet des guildes affiche les informations sur ces dernières, incluant notamment le prestige de la guilde ainsi que son chef.
L'utilisateur a également la possibilité d'accéder entre autres aux meilleurs scores, à son classement personnel, ainsi qu'à une recherche de profil des autres utilisateurs via l'onglet des scores.

1.5 Le magasin

La principale difficulté pour l'implémentation de l'onglet magasin provient de l'utilisation de variables de session. Lorsqu'un utilisateur faisait un achat, le montant n'était pas décompté dans la base de données. Pour y pallier, la base de donnée est modifiée directement après l'achat et l'utilisateur est redirigé sur la page du personnage afin de laisser le temps de faire la mise à jour, faute de quoi les achats successifs entraînaient des retraits et des ajouts d'argent aléatoires.
Un autre problème qui s'est posé pour l'affichage des articles du magasin (ainsi que pour l'affichage du nom des quêtes) a été la présence d'espaces dans les noms de ces derniers. Les espaces ont alors été remplacés par des underscore, et ceux-ci sont à nouveau remplacés par des espaces à l'affichage.

1.6 Le statut administrateur

Le statut administrateur permet de bannir une personne (cette personne ne pourra alors plus se connecter, ni s'inscrire sous le même nom d'utilisateur car celui ci sera gardé en mémoire), enlever le bannissement, attribuer le statut administrateur à un utilisateur qui ne l'a pas déjà et changer une personne de guilde pour la mettre dans la guilde à laquelle l'administrateur appartient.

1.7 Intégration d'une guilde

Chaque personnage se voit affecté à l'une des 4 guildes. Le choix de celle-ci se fait par le biais d'un formulaire de type "test de personnalité", où chaque question laisse le choix entre deux réponses. Chacune des réponses est en faveur de deux des guildes et en défaveur des deux autres, ce qui se traduit à chaque réponse par un +1 pour les guildes favorisées par la réponse et un -1 pour celles qui sont défavorisées. Le joueur serait alors intégré dans la guilde où il a le plus grand nombre de points.
Le problème qui s'est posé est que cela forçait à modifier 4 variables JS en direct, variable qui sont difficilement récupérables en PHP. Une

première solution a été d'utiliser des variables cachées, mais il aurait fallu avoir un formulaire pour chaque question. Finalement, les boutons de type `radio` ont été remplacés par des boutons de type `button` qui permettent de transmettre la valeur associée à chaque guilde dans un seul champ caché.
Cependant cela a posé un nouveau problème : si l'utilisateur cliquait plusieurs fois sur un même bouton, il pouvait incrémenter d'autant le score associé à la guilde. Pour exclure cette possibilité, un attribut `onclick` a été ajouté, faisant appel à la fonction modifiant le score puis désactivant le bouton.

1.8 Gestion du temps de mission

Afin de lancer une mission, un bouton `submit` est associé à chaque quête de l'onglet correspondant et permet de récupérer la quête sélectionnée. La table `adventure` permet d'accéder à sa durée, et les champs `lastquest` et `endtimequest` de la table `perso` sont mis à jour avec respectivement le nom de la quête et l'heure à laquelle elle se termine. Tant que la quête est en cours, la page de quête affiche le nom de la quête en cours et un minuteur montrant le temps restant. Ce minuteur continue même si l'on change de page ou que l'on se déconnecte : il ne s'arrête pas là où on l'a laissé pour ensuite reprendre lorsque l'on revient sur la page.
Une fois la quête finie, un bouton caché jusque là apparait pour ramener le joueur vers la page de récompense qui va directement le rediriger vers la page des quêtes où il pourra en sélectionner une nouvelle.

Pour aller plus loin

Pendant ce projet, nous avons mené une réflexion critique sur notre réalisation et avons envisagés plusieurs manières de l'améliorer dans un contexte plus libre, avec plus de temps.
- Ajouter des armes et armures avec leurs caractéristiques propres, permettant d'aller au-delà de la dualité mage-guerrier.
- Ajouter des objets à la boutique permettant notamment d'augmenter les chances de réussite des mission ou de réduire le temps nécessaire.
- Ajouter la possibilité d'envoyer des messages, au moins aux administrateurs pour signaler des utilisateurs.
- Organiser des compétitions occasionnelles entre les guildes.
- Ajouter la possibilité de payer un compte premium qui donnerait différents avantages tels que l'argent ou l'expérience rapporté multiplié par deux, et ce dans le but de fidéliser les utilisateurs et de rentabiliser le jeu.
- Rendre le jeu un peu plus ergonomique, par exemple en affichant un popup en fin de quête pour informer le joueur de ce qu'il a obtenu.
- Ajouter de l'interactivité avec le familier, par le biais d'un clicker par exemple afin de lui faire gagner de l'expérience plus rapidement.
- Ajouter une jauge de fatigue, qui empêcherait le joueur d'enchainer trop de quêtes à la suite, et incitant à prendre un compte premium.
- Ajout d'un système d'amis et de chat afin de donner au jeu une dimension sociale.

Répartition des rôles au sein de l'équipe

Création de la base de données : Martin Bourdon, Maria Minko
Formulaires JS : Jordan Aurey
Architecture du site : Laurie Pandraud
Implémentation des pages du site : Laurie Pandraud
Fonctionnement des quêtes et gestion du timer : Martin Bourdon
Création des assets graphiques : Maria Minko
Equilibrage des chances de succès : Maria Minko
Game Design : chacun a apporté de ses idées
Aspect général du site et CSS : Jordan Aurey
Partie Controlleur et PHP : Laurie Pandraud
Rédaction du rapport : Maria Minko