# Objectif

**L’objectif** de ce travail est de **valider l’attente des sous-objectifs** dans le cadre du cours :

1. Effectuer le développement d’applications Web transactionnelles. (Compétence 00SU)
   * 5 - Programmer la logique applicative du côté serveur.
   * 6 - Programmer la logique applicative du côté client.
   * 7 - Contrôler la qualité de l’application.
   * 9 - Rédiger la documentation.

# Mise en situation

À partir de **récits utilisateurs**, et en suivant cette **méthodologie**, créer un **jeu en VueJS** dont l'objectif est de **survivre à cinq missions** tout en obtenant le plus de **crédits galactiques**.

L'application communique avec un service web.

## Détails supplémentaires

1. Respecter les [bonnes pratiques de code suivantes](http://appweb.progwmj.ca/documentations/bonnes-pratiques/code)
2. Vous devez respecter les **récits utilisateurs fournis**
3. Vous devez respecter les **consignes énoncées** plus bas.
4. N’oubliez pas de remplir la **grille d’auto-évaluation Excel**!

# Récits utilisateurs

🟢 = Récit disponible à tout moment.   
🟡 = Récit disponible dans la page d'accueil.  
🔴 = Récit disponible en cours de partie.  
🔷 = Bonus.   
CG = crédits galactiques

|  |  |  |
| --- | --- | --- |
|  | En tant qu'utilisateur | Précisions |
| 🟢 1 | Je veux pouvoir accéder à la page d'accueil. | Un utilisateur peut à tout moment revenir à la page d'accueil. À noter que s'il a commencé une partie, il perdra la partie en cours après avoir confirmé qu'il veut quitter. |
| 🟢 2 | Je veux pouvoir voir le tableau de pointage. | Le tableau contient tous les noms des gagnants avec le pointage (CG gagnés). Ce tableau est trié en fonction du nombre de CG gagnés (en tête de liste = le joueur avec le plus de CG). Un utilisateur peut à tout moment voir la page de pointages. À noter que s'il a commencé une partie, il perdra la partie en cours après avoir confirmé qu'il veut quitter. . |
| 🟡 3 | Je veux pouvoir choisir le nom de mon personnage. | Le nom est au choix du joueur. |
| 🟡 4 | Je veux pouvoir choisir le nom de mon vaisseau. | Le choix du nom du vaisseau se fait à partir d'une liste prédéfinie. |
| 🟡 5 | Je veux commencer une partie. |  |
| 🔴 6 | Je veux pouvoir voir les caractéristiques associées à mon personnage. | Les caractéristiques sont le nom du personnage, son expérience (tous les joueurs sont "Maitre"), le nombre de CG gagné (0 en début de partie), le nom du vaisseau et la vie en pourcentage (100% en début de partie). |
| 🔴 7 | Je veux pouvoir voir les caractéristiques de mon ennemi. | Les caractéristiques sont le nom du personnage, son expérience (débutant, confirmé, expert ou maitre), le nombre de CG qu'il possède, le nom du vaisseau et la vie en pourcentage. Dans l'API REST, 1=débutant et 4=Maitre. |
| 🔴 8 | Je veux voir quelle est ma mission en cours. | Au démarrage, la mission en cours est 1/5. |
| 🔴 9 | Je veux combattre un ennemi. | Dans un combat, le joueur et l'ennemi attaque en même temps. Un personnage a plus de probabilité de toucher la cible s'il possède plus d'expérience (débutant= 20%, confirmé=35%, expert=50%, Maitre = 70%). Un vaisseau touché fait perdre aléatoirement de 3% à 6% de vie. Il est autorisé de modifier tous les % ci-dessus, afin d'avoir une meilleure expérience de jeu. Si un ennemi est détruit, le joueur gagne les CG (récit #10) et la mission est terminée (récit #11). Si le joueur est détruit, c'est la fin de la partie (récit #13). On ennemi détruit ne peut plus combattre à nouveau. |
| 🔴 10 | Je veux pouvoir récupérer les CG de l'ennemi. | Le joueur gagne tous les CG de l'ennemi s'il détruit, lors d'un combat (récit #9), le vaisseau de l'ennemi. Un message doit informer l'utilisateur des CG gagnés. |
| 🔴 11 | Je veux pouvoir terminer la mission en cours. | Le fait de terminer une mission affiche un nouvel ennemi et incrémente "mission en cours". |
| 🔴 12 | Je peux gagner une partie. | Si l'utilisateur a survécu à 5 missions, il est informé de son pointage (nombre de CG gagné) et il est ensuite redirigé vers la page des pointages. Le nom du joueur et son pointage doivent se retrouver dans le tableau de pointage (récit #14). |
| 🔴 13 | Je peux perdre une partie. | Le joueur perd la partie dès que sa vie est de 0 ou moins. Un message doit informer l'utilisateur qu'il a perdu (il n'a pas complété l'objectif de 5 missions). Il est ensuite redirigé vers la page d'accueil. |
| 🔴 14 | Je veux ajouter mon nom au tableau des pointages | Un joueur ayant réussi 5 missions doit se retrouver dans le tableau des pointages (récit #2). |
| 🔴🔷 15 | Je veux pouvoir réparer mon vaisseau. | Un joueur peut décider, en une même action, de terminer la mission (récit #11) et de réparer son vaisseau. Le coût de réparation est de 5 CG par % de vie. |

# Consignes

* Travail pratique **en équipe de deux (20% de la note)**
* Réalisé en-**dehors** **du** **cours et temps en classe**
* Durée approximative de **40 heures par équipe**.
* Le fichier de remise est un fichier compressé nommé : **appweb\_trpr02\_votrenom.zip.**
  + Les **fichiers complets** de votre site
  + Modifier le fichiers **CERTIFICATION.md** avec vos infos
  + **IMPORTANT** supprimer le dossier **node\_modules**
  + La **grille de correction** au format **Excel dans le dossier doc du site**
* Remis sur **Léa**:**Mardi 19 avril avant 15:00**

# Critères d’évaluation (100 %)

Le travail est évalué selon les critères suivants et est ensuite pondéré par le pourcentage des récits utilisateurs réalisés et fonctionnels.

### Quantité et utilisabilité des récits utilisateurs réalisés

QTU(14 récits \* 3) + QC(14 récits \* 3) = 42% + 42% => 84%

*Récits utilisateurs fonctionnels. Un récit utilisateur est considéré réalisé que s'il est accompagné par des tests pertinents et suffisants pour en certifier le bon comportement.*

## Qualité des tests unitaires (42 %)

* Bonnes pratiques de programmation et code propre (exemples: noms significatifs, évitez la répétition, clarté et simplicité, respect des normes, etc.)
* Choix et pertinence des tests
* Bonne utilisation des "mocks"
* Bonne couverture des tests

## Qualité du code (42 %)

* Bonnes pratiques de programmation et code propre (exemples: noms significatifs, évitez la répétition, clarté et simplicité, respect des normes, etc.)
* Bonne gestion des erreurs
* Architecture facilitant la maintenabilité et l'évolution des applications
* Bonne structure de projet

## Interface utilisateur (10 %)

* Aucun bogue, aucune erreur d'exécution et aucun problème de performance.
* Qualité de l'interface : convivialité, notifications à l'utilisateur, alertes, messages de confirmation, etc.

## Bonus (+5 %)

Le récit #15 est un bonus d'une valeur de 5%. Pour **être considéré comme bonus**, le récit doit **être fonctionnel** et accompagnés **de tests unitaires**.

## Respect des consignes (6 %)

* Les consignes du travail sont respectées

***La grille de correction est fournie sur Léa.***