Projet de Programmation Web : ClassClash

Alexis Cajal(4100340), Adrien Rigal(4200478), Pol-Elouan Brient(4100295), Favard Eloi(4200146)

I. Présentation du projet

Toute l'équipe s'est accordée dès le premier jour pour développer une application web de quiz de culture générale. Notre but initial était de construire une plateforme sur laquelle chacun pourrait tester ses connaissances culturelles sur des thèmes variés, mais aussi pouvoir se mesurer à un ami dans un duel de questions.
Notre application web remplit donc ces deux fonctions. La page d'accueil présente deux modes de jeu : l'entraînement et le versus. L'entraînement incite le joueur à choisir un thème et une difficulté parmi les 50 combinaisons possibles et lui propose alors une série de questions correspondant à ses critères jusqu'à ce que celui-ci décide d'arrêter son entraînement ou bien que l'algorithme ne dispose plus de questions pertinentes. Le mode versus quant à lui, voit 2 adversaires s'opposer dans un duel de questions. Chaque joueur inflige un certain nombre de dégâts à l'adversaire en cas de bonne réponse à la question dépendamment de la difficulté choisie : une bonne réponse à une question Extrême inflige bien entendu plus de dégâts qu'une question Facile. Le duel se déroule sous forme de tours successifs où les joueurs peuvent user intelligemment de leurs bonus pour vaincre l'adversaire. Le jeu s'arrête lorsque la vie d'un joueur descend à 0.

Notre application web est accessible à l'adresse suivante : https://classclash.onrender.com/

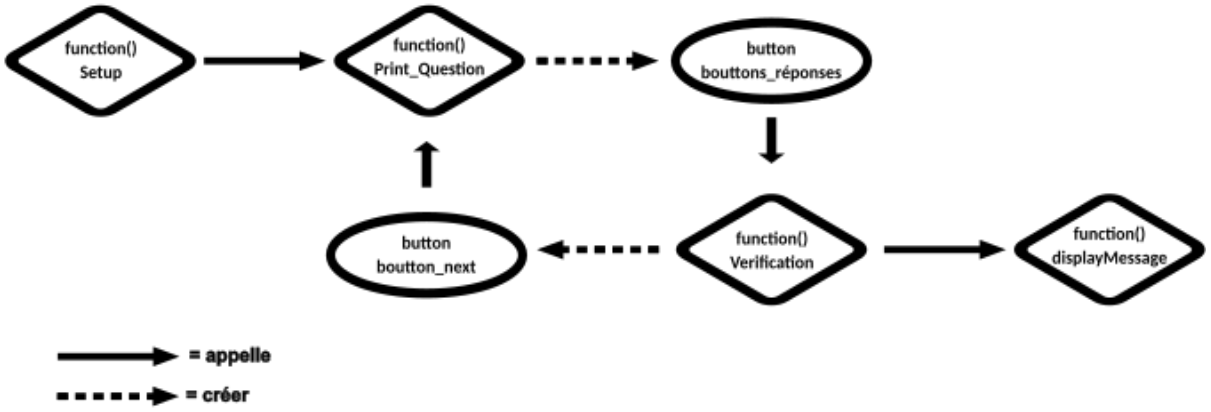
II. Explication de notre implémentation

Notre projet étant divisé en deux parties distinctes, nous avons pris la décision de nous séparer en deux groupes affectés chacun à un mode de jeu de notre application. Ceci, dès lors que nous nous étions mis d'accord sur des éléments communs, comme le nom des pages ou encore la direction artistique à suivre, implémentée dans un fichier CSS commun. Adrien et Eloi s'occupaient du mode Entraînement, tandis que Pol-Elouan et Alexis ont réalisé le mode Versus.

1) Implémentation du mode Entraînement

La partie Entraînement tient en une page HTML. Son contenu est actualisé dynamiquement tout au long du processus en utilisant des méthodes comme `createElement` ou `replaceChildren`. Celle-ci permet notamment de ne remplacer que certaines parties de la page sans avoir à recréer tous les éléments fixes à chaque actualisation. Les nombreux `addEventListener` permettent de faire le lien entre tous ces éléments de manière cohérente afin de garder en mémoire les différents éléments nécessaires au jeu. Pour ce qui tient de la partie technique de l'Entraînement, nous avons décidé de représenter nos questions dans un tableau stocké dans un fichier à part afin de ne pas alourdir le code de nos fichiers `.js`. Nous pouvons extraire de ce tableau les questions qui correspondent aux critères du joueur avec la méthode `filter` et une `boucle for`, qui nous a semblé plus adapté pour un parcours complet d'un tableau. Nous voulions initialement construire une seule fonction globale s'exécutant une fois le thème et la difficulté choisis dispensant les questions en continu à

l'aide d'une boucle while globale. Cependant, nous nous sommes heurtés au problème de l'attente entre les questions. Au vu des possibles solutions à notre problème trouvées sur Internet, il nous a semblé laborieux de "bloquer" notre boucle while dans l'attente d'une réponse de l'utilisateur. A la place, nous avons décidé de construire une sorte de boucle entre les fonctions, que l'on peut illustrer de la manière suivante :



Enfin, les sons spécialement créés par Pol pour ce projet sont joués à l'aide des méthodes classiques javascript play et pause. Le volume de la musique et des effets sonores est quant à lui transmis de page en page au sein de toute l'application au travers du localstorage, solution cross-plateforme qui correspond parfaitement à nos besoins.

2) Implémentation du mode Versus

Le mode Versus a été développé pour permettre à deux joueurs de s'affronter dans un duel de questions-réponses avec des mécanismes intégrés pour la gestion des points de vie et la sélection aléatoire de thèmes et de difficultés.
L'implémentation commence dans le game_logic.js avec les lignes 5-30 où les noms des joueurs sont récupérés et affichés grâce au localStorage, ce qui va garantir une continuité entre les pages et permettre d'afficher le nom des joueurs tout le long de la partie et dans les animations, sans besoin parasite de recharger la page. Les mécanismes principaux du Versus sont implémentés dans les lignes 75-220, où les fonctions comme applyDamage et switchPlayer gèrent respectivement les points de vie et le changement de joueur, ce qui amène la fluidité dans le gameplay en plus de la praticité du code. Une boucle forEach (ligne 180) est utilisée pour gérer dynamiquement les boutons des réponses, car cette boucle s'adapte vraiment bien à la liste d'options, ce qui évite les répétitions dans le code. Les thèmes et les difficultés sont sélectionnés aléatoirement avec des méthodes comme Math.random() (ligne 135), qui est une façon simple, rapide et performante pour la génération aléatoire. Enfin l'aspect visuel de la logique du jeu est principalement gérée par la fonction displayMessage (ligne 50), qui affiche les résultats dynamiquement par rapport aux résultats des fonctions citées juste avant, en ajoutant plusieurs animations pour maximiser l'immersion et l'interaction dans le jeu, principalement développées dans le CSS (versus.css). De plus, la fonction ControlSon() gère l'ensemble des interactions audio, y compris le réglage du volume des musiques et effets sonores, ainsi que la gestion des icônes associées. Cette fonction améliore l'expérience utilisateur en rendant le son interactif et personnalisable. Enfin, displaySummary() génère dynamiquement un tableau récapitulatif des actions effectuées à chaque tour. Elle permet aux joueurs de visualiser l'historique de la partie et d'analyser leurs performances.