

Rapport de PS6

Reverse-coding

Julien N'Diaye

Thème : Mémoire

Année 2019-2020

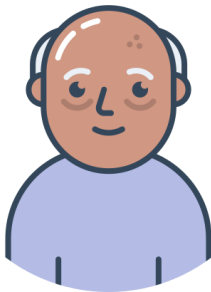
Sommaire

| | |
|--|-----------|
| I - Personas et scénarios | 2 |
| II - Architecture Client Serveur | 6 |
| 1) Composants principaux et services | 6 |
| 2) Architecture de notre api REST | 10 |
| a) Routes | 11 |
| b) Ressources | 12 |
| III - Evaluation croisée et analyse des résultats | 13 |
| Conclusion | 19 |
| Annexes | 20 |

I - Personas et scénarios

Nous allons dans un premier temps présenter les deux personas de notre site "Memory Quiz" puis nous aborderons par la suite les scénarios qui leurs sont liés.

Persona 1



François, 78 ans, marié, est atteint de la maladie d'alzheimer. Il est régulièrement angoissé par sa perte de mémoire immédiate et s'en apercevoir au quotidien le frustre énormément. Sa femme le conduit donc régulièrement au centre d'accueil de jour d'alzheimer afin de l'aider à faire travailler sa mémoire mais aussi lui apporter un soutien psychologique. Cependant, François aimerait travailler avec des outils qui ne le frustrent pas et qui lui permettent d'entraîner sa mémoire sur des thèmes qu'il apprécie.

Persona 2

Margaux, 35 ans, travaille au centre d'accueil de jour d'alzheimer en tant qu'aide médico-psychologique. Elle aimerait redonner confiance en soi à François en valorisant sa mémoire. Un outil lui permettant de créer des quizz personnalisés pour François dans lesquels il n'est jamais mis en échec serait idéal. De plus, elle voudrait avoir la possibilité de consulter les résultats de François sur les différents quizz afin de suivre son évolution au fil du temps.



Scénario 1 : Création d'un compte et d'un quiz

Margot se rend sur la page de connexion du site. Elle s'inscrit sur le site, puis se connecte avec ses identifiants. Arrivée sur la page d'accueil, elle clique sur le bouton *Gestion* puis *Création Quiz*. Elle veut créer un quiz personnalisé pour François sur le football car il est fan de ce sport depuis tout jeune. Elle entre donc "Football" comme nom de quiz et sélectionne le thème "SPORT" avant de cliquer sur *Créer*. Elle est ensuite redirigée vers la page d'édition de son quiz et peut lui ajouter des questions. Elle ajoute donc une première question en y ajoutant un indice et un temps limité d'une minute afin d'éviter que François ne reste bloqué trop longtemps sur la question. Une fois la question ajoutée, elle peut alors lui ajouter un maximum de quatre réponses dont une seule correcte et peut modifier à tout moment l'indice de la question. Elle peut également supprimer les questions et les réponses lorsqu'elle le souhaite. Ainsi, elle réitère l'opération pour chaque question qu'elle veut ajouter. Enfin, une fois son quiz terminé, elle clique sur *Valider mon quiz* en bas de la page et est redirigée sur la page "Mes Quizzes" où elle peut visualiser tous ses quizz, les modifier et même les supprimer si elle le veut.

Scénario 2 : Ajout d'un joueur et jouer un quiz

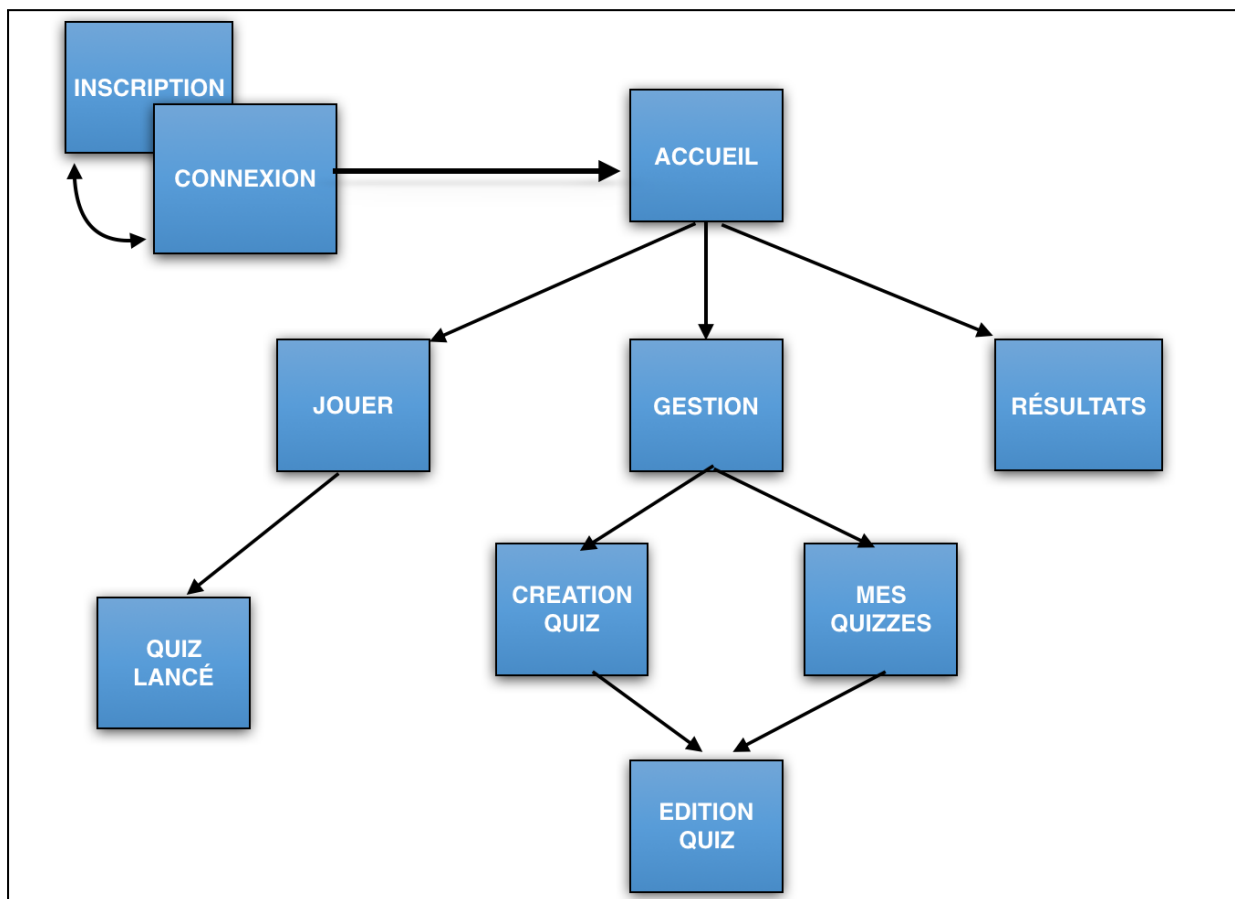
Margot est sur la page d'accueil du site. Elle clique sur le bouton *Jouer* et est redirigée vers la page de sélection de joueur et de quiz. François n'est pas encore inscrit sur le site en tant que joueur. Ainsi, Margot clique sur *Nouveau joueur* et entre le nom, prénom et la date de naissance de François avant de cliquer sur *Valider*. Une fois cette action effectuée, elle peut sélectionner François en tant que joueur puis le quiz qu'elle a créé précédemment pour lui. Une fois les deux informations sélectionnées, elle lance le quiz et François peut commencer à jouer. Lorsqu'il sélectionne une réponse, si elle est fausse la réponse disparaît pour éviter de le mettre en échec et il peut alors en sélectionner une autre. Il a également la possibilité de cliquer sur *Indice* s'il désire avoir de l'aide. Si la réponse est correcte ou que le temps pour répondre est écoulé, alors une pop up s'affiche en rappelant la réponse correcte et François peut passer à la question suivante. Une fois que François a répondu à toutes les questions, une pop up de fin de quiz apparaît et Margot peut alors décider de faire rejouer François, de retourner à l'accueil ou bien encore d'afficher le résultat du quiz qu'il vient de jouer.

Scénario 3 : Afficher les résultats d'un joueur

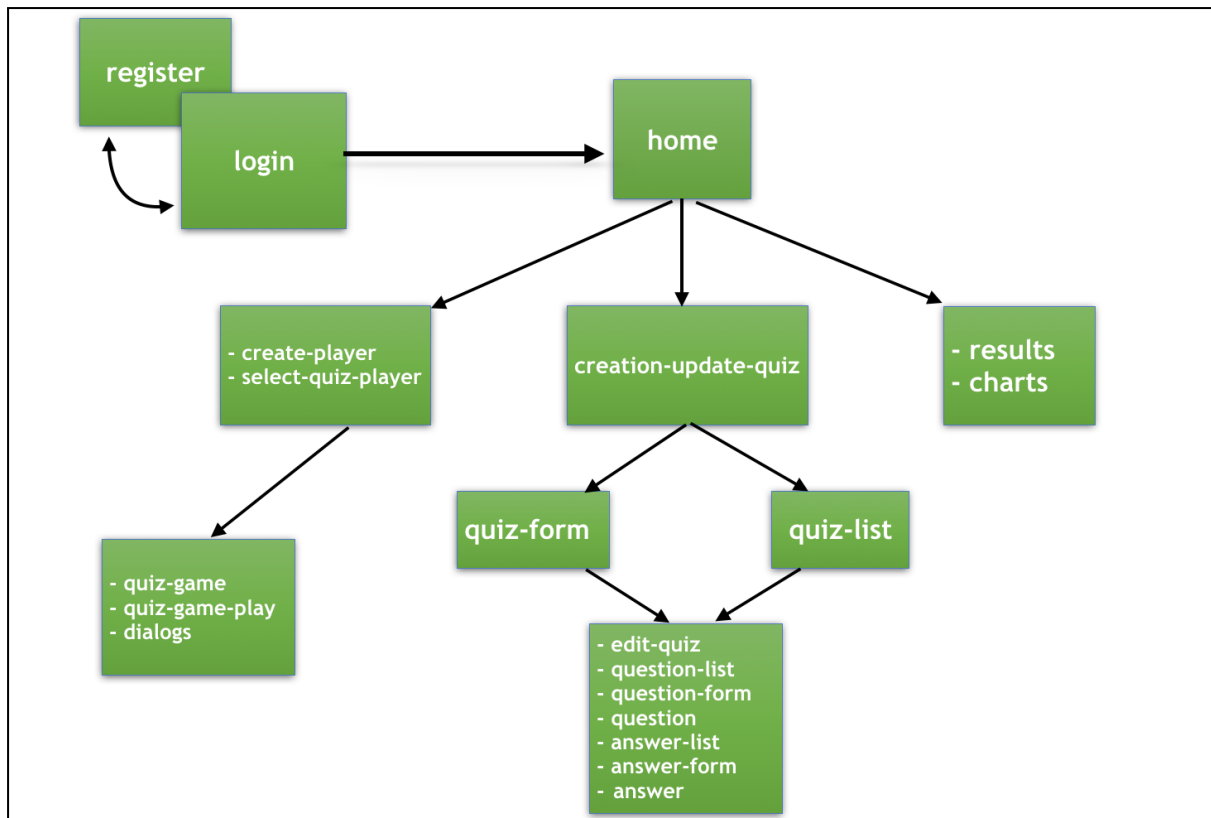
Après avoir fait jouer François sur le quiz "Football" pendant plusieurs jours, Margot désire observer ses résultats et voir son évolution. Elle se rend donc sur la page d'accueil et clique sur le bouton *Résultats*. Elle sélectionne alors dans un premier temps le joueur qu'elle veut analyser (François) puis le quiz dont elle veut les résultats (Football). Une série d'information apparaît alors. Elle peut voir dans le premier tableau la date et le nombre de questions réussies pour chaque partie effectuée par François. Elle a également la possibilité de cliquer sur *Détail* pour chaque partie, ce qui lui permet de savoir précisément pour chaque question, les réponses que François a coché avant de trouver celle correcte. Enfin, le deuxième tableau lui permet de savoir quelles sont les questions du quiz qui posent le plus de problème à François puisqu'il affiche le nombre total d'échec pour chaque question. Une fois toutes les actions terminées, Margot se déconnecte du site en cliquant sur *Se déconnecter* en haut à droite.

II - Architecture Client Serveur

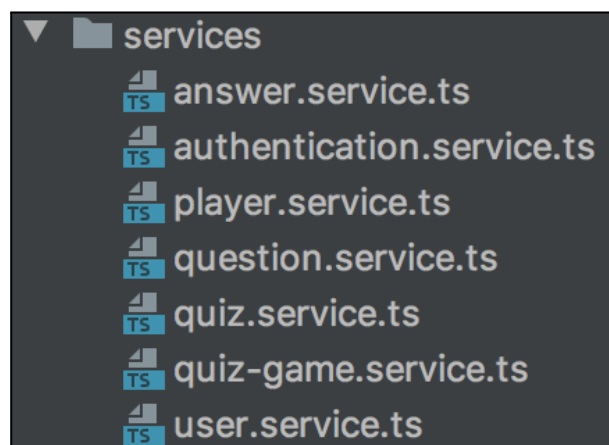
1) Composants principaux et services



Ce schéma représente l'architecture de notre site côté client, autrement dit vu par l'utilisateur. En effet, chaque carré bleu représente une page de notre site et chaque flèche représente les liens entre ces pages. Bien entendu il s'agit des liens majeurs et il en existe bien d'autres, notamment la page d'accueil qui reste accessible sur toutes les pages une fois connecté, mais également la possibilité de revenir à la page précédente. Ces pages sont composées de différents composants encapsulés que nous allons vous présenter ci dessous.



Ce schéma présente la majeure partie des composants de notre site tout en précisant sur quelles pages ils sont utilisés. Nous allons donc à présent expliquer leur rôle au sein du site mais également leurs interactions avec les différents services. En effet, nous avons créé un service spécifique pour chaque ressource de notre site (utilisateur, joueur, quiz, question,...) afin de gérer plus facilement les interactions client serveur qui leurs sont liées. Un service pour l'authentification a également été créé afin de gérer la connexion des utilisateurs. Ainsi, les services nous ont permis de faire différentes requêtes à notre API mais également de récupérer les informations qu'elle renvoie en les stockant dans des Observables.



Détails de nos composants

Register & Login : Ce sont les premiers composants auxquels l'utilisateur est confronté lorsqu'il arrive sur notre site. Comme leur nom l'indique, c'est grâce à eux que l'utilisateur (l'encadrante) va pouvoir respectivement renseigner ses informations pour se créer un nouveau compte utilisateur puis s'y connecter afin d'accéder aux différentes fonctionnalités offertes par notre site. Quant aux services, c'est respectivement le service *User* qui fait le lien avec le backend pour créer un nouvel utilisateur et le service *Authentication* qui gère la connexion à un compte utilisateur.

Header : Il n'apparaît pas sur le schéma puisqu'il est utilisé sur toutes les pages et change en fonction de la page affichée. Il est utile pour naviguer plus facilement à travers le site, permet également à l'utilisateur de se déconnecter du site (en faisant appel au service *Authentication*) ou encore à un joueur de quitter un quiz lorsqu'il le désire.

Home : Il fait office de page d'accueil et permet de rediriger efficacement les utilisateurs sur les différentes pages du site en fonction des actions qu'ils souhaitent réaliser.

Creation-update-quiz : Il représente la page de gestion et permet de rediriger l'utilisateur en fonction de s'il désire créer un quiz ou accéder à la liste de ses quizz.

Quiz-form : Il permet à l'utilisateur de créer des quizz et interagit avec le service *Quiz* afin d'envoyer une requête de création de quiz au backend. Il redirige ensuite directement l'utilisateur vers la page d'édition du quiz si le quiz créé ne contient pas d'erreur.

Quiz-list : Ce composant permet à l'utilisateur de visualiser ses quizz en détail, d'accéder à leur page de modification et d'avoir la possibilité de les supprimer par l'intermédiaire encore une fois du service *Quiz*.

Edit-quiz : Comme son nom l'indique, il permet d'éditer ou de modifier un quiz. Pour cela, il encapsule de nombreux composants qui permettent respectivement d'effectuer une tâche. La première étant la création de question, puis la visualisation des questions du quiz, et l'ajout et visualisation des réponses de chaque question. Ces composants font appel respectivement aux services des ressources qu'ils traitent. Ainsi, les services *Quiz*, *Question* et *Answer* sont utilisés afin de récupérer, créer, supprimer ou encore modifier les ressources d'un quiz.

Select-quiz-player : Il est utile, comme son nom l'indique encore une fois, pour sélectionner le joueur qui va jouer et le quiz qui va être joué. C'est un composant essentiel puisqu'il transmet les informations que l'utilisateur a sélectionné aux composants qui gèrent le déroulement d'une partie. Il fait appel aux services *Player* et *Quiz* afin de récupérer dans le backend les données associées à l'utilisateur. Le service *QuizGame* est également appelé au lancement du quiz afin d'enregistrer la partie dans le backend.

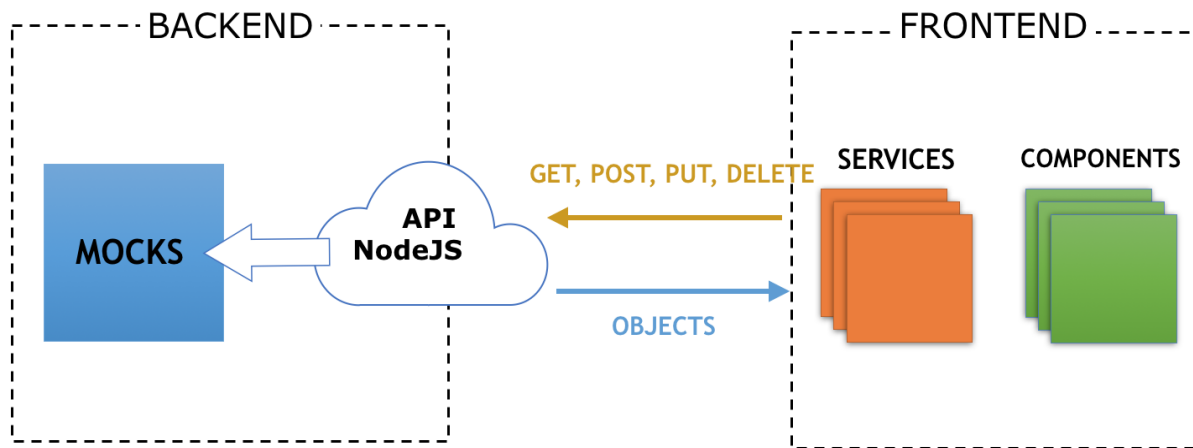
Create-player : Assez explicite, il permet à l'utilisateur de créer un nouveau joueur et fait appel au service *Player* afin de créer le joueur dans le backend.

Quiz-game & Quiz-game-play : Ce sont les composants qui gèrent le déroulement d'une partie. Ils sont également essentiels puisqu'ils permettent d'enregistrer les réponses cochées par le joueur en interagissant avec le service *QuizGame*. Leurs données sont précieuses puisqu'elles sont indispensables aux composants qui gèrent l'affichage des résultats.

Results & Charts : Ils gèrent tous les deux la partie résultat des joueurs. *Results* permet plus particulièrement de choisir le joueur et le quiz pour lequel on souhaite afficher les résultats. Il transmet donc les informations sélectionnées au composant *Charts* qui s'occupe de les afficher. *Results* fait appel aux services *Quiz* et *Player* afin de récupérer les quizz et les joueurs de l'utilisateur connecté.

Dialogs : Enfin, notre site est également composé de nombreux composants dialog (non présents sur le schéma), qui font office de pop up et qui permettent d'informer ainsi que d'effectuer différentes actions. Ils se trouvent principalement dans la partie de jeu de quiz et servent à passer aux questions suivantes, afficher des informations ou encore avertir de la fin du quiz.

2) Architecture de notre api REST



Le schéma ci-dessus représente l'architecture globale de notre site ainsi que son fonctionnement. On peut apercevoir dans un premier temps la partie frontend qui représente le côté client du site. Comme nous l'avons présenté en détail précédemment, elle est composée de différents composants et services. Ces composants font appel aux services afin qu'ils effectuent différentes requêtes (récupération, création, modification, suppression) sur les ressources du site à notre API.

Dans un second temps, on distingue la partie backend qui représente le côté serveur du site. Elle est composée de notre API REST mise en place grâce à Node JS, et de fichiers mocks permettant de stocker les différentes ressources de notre site. Lors de la réception d'une requête, l'API se charge d'effectuer l'action demandée en fonction de la route et du nom de la méthode (GET, POST, PUT, DELETE), puis de faire le lien avec les fichiers mocks. Enfin, elle retourne les ressources associées à la requête au service correspondant du côté client.

a) Routes

Maintenant que nous avons expliqué en détail le fonctionnement du site et plus particulièrement de l'API, nous allons présenter ses différentes routes.

Ainsi, notre API est basée sur **deux grandes routes**, une qui gère les utilisateurs et les ressources qui lui sont associées, c'est-à-dire ses quizz et ses joueurs. Et une qui gère les parties effectuées sur le site.

- **/users**

Comme nous l'avons indiqué, elle permet de traiter toutes les requêtes sur la ressource **User** mais également sur la totalité des ressources d'un utilisateur. Ainsi, elle se décompose ensuite en deux nouvelles routes :

- ❏ **/user/:userId/quizzes**

Celle-ci permet de traiter les requêtes sur les quizz d'un utilisateur identifié par son **userId**. Elle peut se prolonger afin de traiter des requêtes sur les questions d'un quiz identifié par son **quizId** grâce à la route **/user/:userId/quizzes/:quizId/questions**.

Enfin, de la même façon que user et quiz, des requêtes peuvent être traitées sur les réponses d'une question identifiée par son **questionId** grâce à la route **/user/:userId/quizzes/:quizId/questions/:questionId/answers**.

- ❏ **/user/:userId/players**

Et celle-là permet de traiter les requêtes des joueurs (patients) d'un utilisateur (encadrant).

- **/quizgames**

Cette route permet de traiter toutes les requêtes sur la ressource QuizGame. Nous avons fait le choix de créer la route **quizgames** en dehors de **users** afin de simplifier les routes et donc d'accéder plus facilement aux QuizGames. En effet, au début nous voulions la mettre à l'intérieur de la route users, de la même façon que quiz et player, mais cela ajoutait certaines contraintes et difficultés notamment lors de l'affichage des résultats.

b) Ressources

Comme nous l'avons évoqué à de nombreuses reprises précédemment, notre site manipule, gère, stocke, transmet différentes ressources. Nous allons donc ici les présenter en détail.

User : Ce modèle définit toutes les variables associées aux utilisateurs de notre application, à savoir les encadrants. Ainsi chaque user est identifié par son *prénom* et *nom*, *email* et *mot de passe* tous requis. Un utilisateur possède aussi un ensemble de *joueurs* (les personnes âgées) et de *quiz*.

Quiz : Comme son nom l'indique, cette ressource contient toutes les informations relatives à un quiz, son *thème*, son *nom*, qui sont tous deux requis, mais aussi sa *date* de création, les *questions* qui lui sont associées et *l'identifiant du user* qui l'a créé.

Question : Chacune a nécessairement un énoncé et un id lors de sa création. Elle possède aussi un ensemble de *réponses*, un éventuel *indice* et *chronomètre* ainsi qu'un *identifiant* déterminant le quiz à laquelle elle appartient.

Answer : De la même manière, chaque réponse a obligatoirement un énoncé et un booléen déterminant s'il s'agit d'une bonne réponse ou non. Elle a en plus un *identifiant*, celui de la question à laquelle elle est rattachée.

Player : En ce qui concerne un joueur, on doit lui associer un *prénom* et *nom*, un *âge* ainsi que l'ensemble des *parties* qu'il a réalisé. Il possède en plus *l'identifiant du user* (encadrant) qui a créé son profil.

QuizGame : Enfin, on attribue nécessairement à chaque partie les *identifiants du quiz* qui a été joué ainsi que celui du *joueur*, le *nombre* de mauvaises réponses et l'ensemble des *questions ratées*. De plus, une partie possède l'ensemble des *mauvaises réponses sélectionnées* et la *date* à laquelle elle a été jouée.

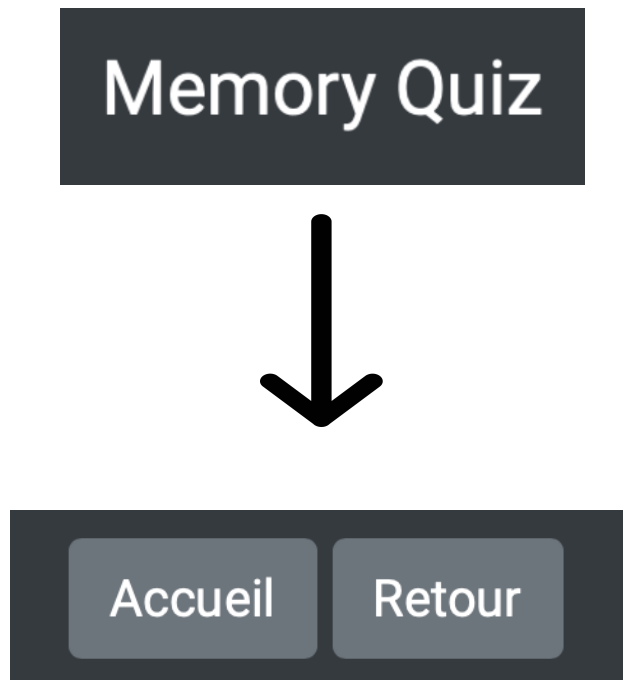
III - Evaluation croisée et analyse des résultats

Nous allons maintenant aborder l'évaluation croisée que nous avons réalisé avec l'équipe ze-commit. Nous avons choisi la méthode d'évaluation V1 car nous avons jugé qu'elle était la meilleure pour évaluer l'utilisabilité de notre site.

Ainsi, elle s'est révélée être d'une grande aide puisque nous avons pu identifier un nombre conséquent de problèmes plus ou moins importants et rapides à résoudre sur notre site. Néanmoins, nous avons pu également nous rendre compte à travers cette évaluation, des points forts du site ce qui nous a confirmé que nous étions sur la bonne voie.

L'objectif de cette partie est donc de présenter les différents problèmes constatés lors de l'évaluation et les réponses apportées pour les corriger. Bien entendu, nous n'avons pas encore pu régler tous les problèmes mais nous tâcherons de le faire pour la remise de la vidéo.

Général






Commençons donc par présenter le problème majeur qui est revenu tout au long de l'évaluation croisée et qui a rendu compliqué l'exécution du scénario par l'utilisateur. Ce problème est l'accessibilité des différentes rubriques de notre site. En effet, à de nombreuses reprises, l'utilisateur s'est retrouvé bloqué sur une page sans savoir où cliquer pour revenir à l'accueil ou bien à la page précédente. Ce problème vient du fait que nous avons jugé intuitif, à force d'utiliser le site, le fait de cliquer sur le nom du site dans le header pour revenir à l'accueil (voir l'image ci dessus). C'est un problème important mais facile à résoudre puisqu'il nous a fallu simplement ajouter un bouton **Accueil** et **Retour** dans le header du site.

Partie Gestion

Au niveau maintenant de la création et de la gestion de quiz, l'utilisateur a remarqué différents problèmes. Le premier est qu'il n'est pas possible pour l'utilisateur de créer de nouveaux thèmes. Ainsi, si aucun des thèmes ne représentent le quiz créé, l'utilisateur est coincé. Pour résoudre cela nous avons simplement ajouté un thème "Autre" car nous avons jugé que l'ajout de thème n'est clairement pas une priorité pour l'utilisateur.

Question n° 1 : Dans quel film Louis de Funès joue-t-il un chef d'orchestre pendant la 2eme guerre mondiale ?
Le nombre maximum de réponse pour cette question est atteint.

| | |
|----------------------|---|
| Le Corniaud |  |
| La Grande Vadrouille |  |
| La Folie Des Grands |  |
| Le Grand Restaurant |  |


[Supprimer cette question](#)



Question n°1
Dans quel film Louis de Funès joue-t-il un chef d'orchestre pendant la 2eme guerre mondiale ?

Indice : Ein gross filou ... [Retirer l'indice](#)

Le nombre maximum de réponse pour cette question est atteint.

| | |
|----------------------|---|
| Le Corniaud |  |
| La Grande Vadrouille |  |
| La Folie Des Grands |  |
| Le Grand Restaurant |  |

[Supprimer cette question](#)

Par la suite, l'ajout des questions s'est déroulé sans problème mais l'impossibilité de modifier l'indice une fois la question créée a posé problème. Nous avons donc ajouté cette fonctionnalité (voir image ci dessus). Des modifications ont également été faites au niveau des couleurs des boutons afin qu'ils soient plus intuitifs et une pop up de confirmation a été ajoutée pour tous les boutons de suppression afin d'éviter que l'utilisateur supprime par erreur.

De plus, nous avons constaté que l'utilisateur a pu renseigné des questions contenant aucune bonne réponse ou bien plusieurs. Nous n'avions pas pensé à cette éventualité et sommes en train de corriger le problème qui est assez complexe.



Enfin, lorsque l'utilisateur avait terminé son quiz, la validation ne paraissait pas intuitive. En effet, l'utilisateur pensait supprimer le quiz lorsqu'il cliquait sur le seul bouton possible pouvant les mener à la page suivante à savoir le bouton "voir mes quizz". Ainsi, nous avons renommé le bouton en "Valider mon quiz" afin que l'utilisateur ne doute plus lors de la validation du quiz.

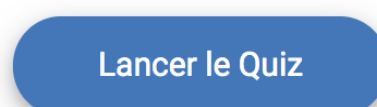
Partie Jouer

En ce qui concerne la partie de jeu, sur la page de sélection, l'utilisateur a identifié un bouton qui n'était pas intuitif. En effet, le bouton "+" servait de raccourci pour accéder à la page de création de quiz directement depuis la page de sélection. Nous avons donc pris la décision de le supprimer pour éviter toute ambiguïté et séparer distinctement les parties de jeu et gestion.

De même la sélection du quiz n'est pas intuitive avant de commencer une partie. L'utilisateur n'a pas trouvé rapidement, il a mis du temps avant de comprendre qu'il fallait cliquer sur un quiz pour lancer la partie. Ainsi, nous avons ajouté des phrases indicatives afin de guider au mieux l'utilisateur (voir ci-dessous).

Veillez sélectionner un quiz pour commencer à jouer.

Veillez sélectionner un joueur pour lancer le quiz.



Ce bouton apparaît une fois qu'un joueur et un quiz sont sélectionnés.

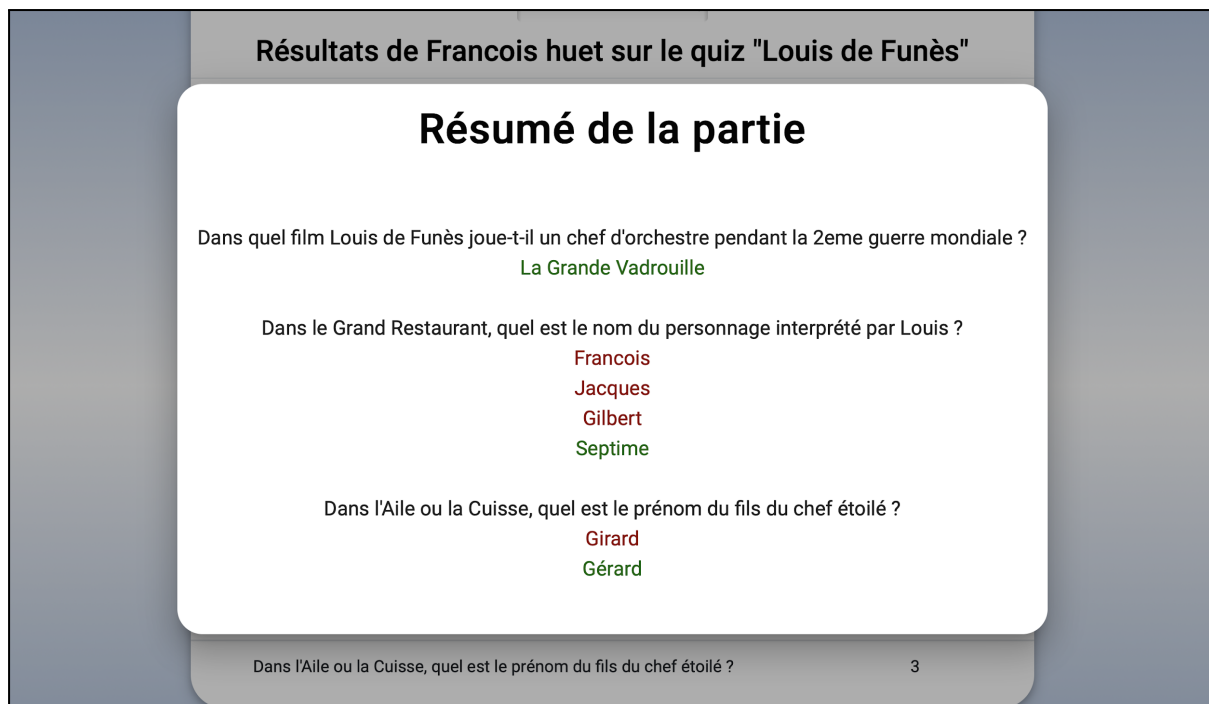
Par ailleurs, au cours de la partie, lorsque l'utilisateur demandait à voir l'indice de la question, une fenêtre pop-up s'affichait. Le problème que nous avons constaté est qu'il manque un bouton pour permettre au joueur de revenir à la question. Cela nous semblait évident, que pour quitter la pop up, il nous suffisait de cliquer à côté mais visiblement pas pour l'utilisateur. Nous avons donc décidé de rajouter un bouton qui permet de fermer la fenêtre pop-up.

Partie Résultats

Pour terminer, en ce qui concerne la partie résultat, nous avons ajouté le résumé d'une partie. Désormais, en plus d'un suivi global au travers des scores de chaque partie et des statistiques des questions ratées au fil du temps, les encadrants peuvent analyser en détail ce qui s'est passé lors de chaque partie et ce grâce au bouton "Détail". Ainsi, comme on peut le voir sur la capture d'écran ci-dessous, les mauvaises réponses qui ont été sélectionnées apparaissent en rouge. Cette fonctionnalité rendra le suivi d'autant plus efficace, puisqu'il permettra de mettre à jour des réponses, si celles ci sont trop souvent sélectionnées à tort.

| Résultats de Francois huet sur le quiz "Louis de Funès" | | |
|---|--------------------------------|-------------------------|
| Date | Nombre de questions réussies | |
| May 4, 2020 | 0/3 | Détails |
| May 15, 2020 | 1/3 | Détails |
| May 15, 2020 | 1/3 | Détails |
| May 20, 2020 | 1/3 | Détails |
| Questions | Nombre d'échec (sur 4 parties) | |
| Dans quel film Louis de Funès joue-t-il un chef d'orchestre pendant la 2eme guerre mondiale ? | 2 | |
| Dans le Grand Restaurant, quel est le nom du personnage interprété par Louis ? | 4 | |
| Dans l'Aile ou la Cuisse, quel est le prénom du fils du chef étoilé ? | 3 | |

Page des résultats principaux



Pop up s'affichant en cliquant sur le bouton Détail

Une remarque supplémentaire qui nous a été faite lors de l'évaluation croisée était le fait qu'en fin de partie, le bouton "Résultats" ne redirigeait pas directement sur les résultats de la partie courante mais qu'il fallait renseigner le joueur et la partie à nouveau. Ce léger manque de cohérence sera corrigé pour la vidéo.

Conclusion

En conclusion, ce projet nous a permis d'obtenir une meilleure vision de ce qu'est un projet informatique, ainsi que ses phases de développement et de peaufiner notre travail d'équipe. D'une part, nous avons appris à nous mettre à la place de l'utilisateur afin de créer une application pour répondre à ses besoins de la meilleure des façons possibles. Pour cela, nous avons pris connaissance de plusieurs nouveaux outils comme l'utilisation des personas et scénarios ainsi que la réalisation d'une maquette, avant de nous lancer dans le développement de notre solution web.

D'autre part, nous avons pu découvrir la technologie Angular et nous familiariser grandement avec celle-ci. Cependant, comme nous l'avons évoqué dans la dernière partie sur l'évaluation croisée, il reste des points de notre site à améliorer. Concrètement, si nous avons encore un mois de développement devant nous, il faudrait en priorité s'occuper de la navigation au sein d'une partie. En effet, par exemple, il serait intéressant de pouvoir naviguer entre les questions lors d'une partie, pour plus de flexibilité dans son déroulement. La possibilité d'ajouter des questions/réponses avec des images serait également intéressante. Nous ajouterions aussi des fonctionnalités concernant le bon remplissage d'un quiz notamment pour éviter les erreurs manifestes comme l'ajout de plus d'une bonne réponse. Enfin, il serait également intéressant de changer la charte graphique du site pour qu'elle soit plus visuelle et plus intuitive pour les encadrants et les patients.

Annexes

Webographie

<https://angular.io/>

<https://openclassrooms.com/>

<https://getbootstrap.com/docs/4.0/getting-started/introduction/>

<https://guide-angular.wishtack.io/>

La vidéo de présentation sera transmise le 29 mai.