

ADRAR Formation – Pôle Numérique

Formation Développeur Web & Web Mobile

LOTR Quiz

Site de quiz sur l'univers du Seigneur des Anneaux



SENAC Jason

Table des matières

| | |
|--|---------------|
| I. Introduction | 4 |
| A. Abstract (Introduction en anglais)..... | 4 |
| B. Introduction en Français | 5 |
| C. Compétences du titre couvertes par le projet (d'après le REAC)..... | 6 |
| II. Analyse du Besoin | 7 |
| A. Contexte – Besoin..... | 7 |
| B. Cibles | 8 |
| C. Matrice SWOT | 8 |
| D. Concurrence | 9 |
| E. Contraintes techniques | 9 |
| F. Contraintes légales et réglementaires | 9 |
| G. Outils Techniques Utilisés | 10 |
| III. Spécifications Fonctionnelles..... | 11 |
| A. Use Case – Cas d'utilisation | 11 |
| B. Diagramme d'activité | 12 |
| C. Diagramme de séquence..... | 13 |
| D. Arborescence..... | 14 |
| E. Maquettage | 15 |
| 1. Charte graphique..... | 15 |
| 2. Zoning..... | 17 |
| 3. Wireframe | 18 |
| 4. Mockup..... | 19 |
| IV. Conception..... | 20 |
| A. MCD – Modèle Conceptuel de Données | 20 |
| B. MLD – Modèle Logique de Données | 21 |
| C. Fonctionnalités | 22 |
| D. Code..... | 23 |
| 1. Front-End..... | 23 |
| 2. Back-End | 26 |

| | |
|---------------------------|-----------|
| V. Conclusion..... | 33 |
|---------------------------|-----------|

| | |
|--------------------------|-----------|
| VI. Annexes | 34 |
|--------------------------|-----------|

| | |
|----------------------------------|----|
| A. Maquettage complet..... | 34 |
| 1. Zoning..... | 34 |
| 2. Wireframe | 36 |
| B. Fonction saveDiffilty() | 39 |
| C. Documentation traduite..... | 39 |

I. Introduction

A. Abstract (Introduction en anglais)

My name is Jason SENAC, I'm 22 years old and I'm going to introduce you to my LOTR quiz project, aimed at Tolkien fans and novices alike, to help them discover the subtleties of Middle-earth.

The aim of my project is to create a quiz site on the Lord of the Rings universe, which is vast and a pioneer in the fantasy theme, and to enable all those who are passionate about this universe to increase their knowledge, but also to allow novices to discover the universe little by little.

The quiz offers five distinct levels of difficulty, ranging from three accessible stages for novices (one stage per Peter Jackson film) to a difficult level for die-hard fans (questions on the extended universe with a stopwatch), via a medium difficulty level also featuring questions on the extended universe, but without a stopwatch. Players will also earn points at the end of each quiz, depending on its difficulty and the number of correct answers.

There's also a function that allows players to turn the points they earn into a pixel art creation, which fosters a sense of creativity and achievement in the game environment, as well as making quizzing a 'useful' way to learn and earn points.

As a big fan of the universe, I've seen the films many times, the series and some of the books, and I can understand that some people are put off by all that, and I felt that there was a lack of a place to learn while having fun, other than the video games in the universe.

In the long term, all these features will be added and the list of available questions will be completed as we go along, and then I'll see how things evolve.

B. Introduction en Français

Le but de mon projet est donc de créer un site de quiz sur l'univers du Seigneur des Anneaux, univers très vaste et pionnier dans le thème fantasy, et de permettre ainsi à tous ceux qui sont passionnés par cet univers d'accroître leurs connaissances, mais aussi de permettre aux novices de découvrir petit à petit l'univers.

Le quiz propose cinq niveaux de difficulté distincts, allant de trois étapes accessibles pour les novices (une étape par film de Peter Jackson) à un niveau difficile pour les fans inconditionnels (questions sur l'univers étendu avec chronomètre) en passant par un niveau de difficulté moyen possédant également des questions sur l'univers étendu, mais sans chronomètre. Les joueurs gagneront également des points à la fin de chaque quiz selon la difficulté de celui-ci et le nombre de bonnes réponses.

Ensuite, une fonction permet aux joueurs de transformer leurs points gagnés en une création artistique en pixels, ce qui favorise un sentiment de créativité et d'accomplissement dans l'environnement du jeu, tout en rendant "utile" le fait de faire des quiz, pour avoir comme but de s'instruire et de pouvoir gagner des points.

Etant grand fan de l'univers, j'ai vu les films de nombreuses fois, la série, certains ouvrages et je peux comprendre que tout ça rebute certains, et je trouvais qu'il manquait un endroit pour pouvoir s'instruire en s'amusant, autre que les jeux vidéo de l'univers.

A long terme toutes ces fonctionnalités seront ajoutées et la liste des questions disponibles complétée au fur et à mesure, et ensuite je verrai comment les choses évoluent.

C. Compétences du titre couvertes par le projet (d'après le REAC)

II. Analyse du Besoin

Je suis seul à travailler sur ce projet, je n'ai pas d'équipe, je n'ai que des amis et famille qui ont testé mon site pour voir si les quiz fonctionnaient bien.

A. Contexte – Besoin

Il n'y a que peu de moyens de s'instruire facilement sur l'univers du Seigneur des Anneaux actuellement, il y a les films, une série, des ouvrages, des jeux, mais surtout des forums et dictionnaires en ligne. Ces forums apportent beaucoup de connaissances, mais peuvent être indigestes pour certains car c'est vraiment que des informations, il n'y a pas ce côté ludique qui permettrait de faciliter l'apprentissage.

Je développe donc un site internet de quiz sur l'univers du Seigneur des Anneaux afin de pouvoir s'instruire tout en ayant un côté ludique. Mon site aura plusieurs fonctionnalités :

- **Différents niveaux de difficulté**

5 difficultés différentes :

- 3 difficultés faciles (une pour chaque film de Peter Jackson, vus par beaucoup de monde)
- 1 difficulté intermédiaire (questions sur l'univers étendu, incluant notamment de nombreux livres)
- 1 difficulté difficile (questions sur l'univers étendu également, mais avec l'ajout d'un chronomètre de 10s par question)

- **Gain de points**

En jouant, un utilisateur accumule ainsi des points grâce aux quiz, selon la difficulté choisie et le nombre de bonnes réponses obtenues.

- **Dessin commun**

Ces points gagnés pourront être dépensés dans un dessin commun, similaire au r/place, avec donc une grille de pixels où l'on peut ajouter un ou plusieurs pixels d'une couleur choisie et ainsi faire de belles créations que tous pourront voir.

- **Consultation d'historique**

Les utilisateurs pourront également consulter leur profil, où ils retrouveront leur historique de quiz, la difficulté dudit quiz et le résultat obtenu.

B. Cibles



Mélanie

45 ans

Professeure

Grande lectrice, a déjà plusieurs ouvrages de Tolkien



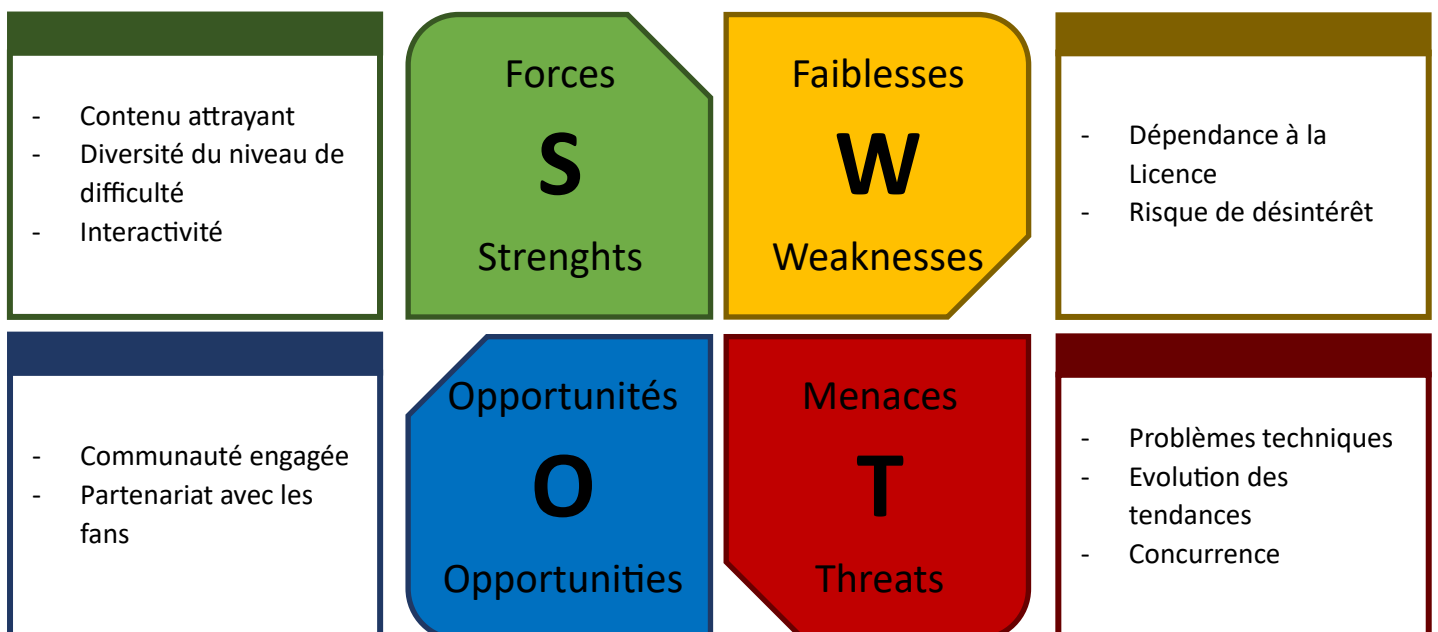
Paul

25 ans

Etudiant

Gros joueur de jeux vidéos mais également fan des films du Seigneur des Anneaux

C. Matrice SWOT



D. Concurrence

Il n'y a pas énormément de jeux de quiz sur l'univers du Seigneur des Anneaux, mais il existe cependant un site qui aide à apprendre des connaissances tout en ayant ce côté jeu qui divertit, il s'agit de [jeux-geographiques.com](https://www.jeux-geographiques.com/jeux-en-ligne-La-Terre-du-Milieu-pageid332.html), qui possède une option Terre du Milieu (<https://www.jeux-geographiques.com/jeux-en-ligne-La-Terre-du-Milieu-pageid332.html>), ce site est bon pour apprendre la géographie de la Terre du milieu sous forme de jeu, mais il ne se limite qu'à la géographie de la Terre du Milieu, il n'y a pas d'autres continents.

E. Contraintes techniques

Le site utilisera un système d'utilisateurs, donc nécessite la création de compte pour les utilisateurs et l'hébergement des données.

Le site nécessitera une modération fréquente pour surveiller le dessin commun et supprimer tout dessin offensant, raciste ou toute forme de discrimination.

Les questions seront à compléter aussi souvent que possible pour pouvoir diversifier au mieux chaque quiz que les utilisateurs feront.

F. Contraintes légales et réglementaires

- Propriétaire du site : SENAC Jason
- Propriétaire du code source : Le code source appartient au développeur dudit code.
- Maquettes et intégrations : Les droits appartiennent au propriétaire du site.
- Logo : Le logo appartient au créateur de celui-ci et au propriétaire du site d'un accord mutuel.
- La licence du Seigneur des Anneaux et l'œuvre de J.R.R. Tolkien appartient à Middle-earth Enterprises depuis 1976, je l'utilise pour ce site mais je ne possède aucuns droits sur celle-ci.

G. Outils Techniques Utilisés

| | |
|---|---|
|  | <u>Balsamiq Wireframe</u> Maquettage |
|  | <u>Github / Github Desktop</u> Sauvegarde du projet et gestion des versions du projet |
|  | <u>Looping</u> MCD/MLD |
|  | <u>StarUML</u> Diagrammes de cas d'utilisations, diagrammes d'activités, diagrammes de séquence |
|  | <u>Visual Studio Code</u> Programmation / Développement du projet |
|  | <u>MySQL / MySQL Workbench</u> Base de données |
|  | <u>Node JS</u> Serveur |

III. Spécifications Fonctionnelles

A. Use Case – Cas d'utilisation



Le diagramme de cas d'utilisation permet de modéliser toutes les interactions entre l'utilisateur et le système. Pour mon diagramme de cas d'utilisation, je possède 3 acteurs : un visiteur, un utilisateur et un administrateur.

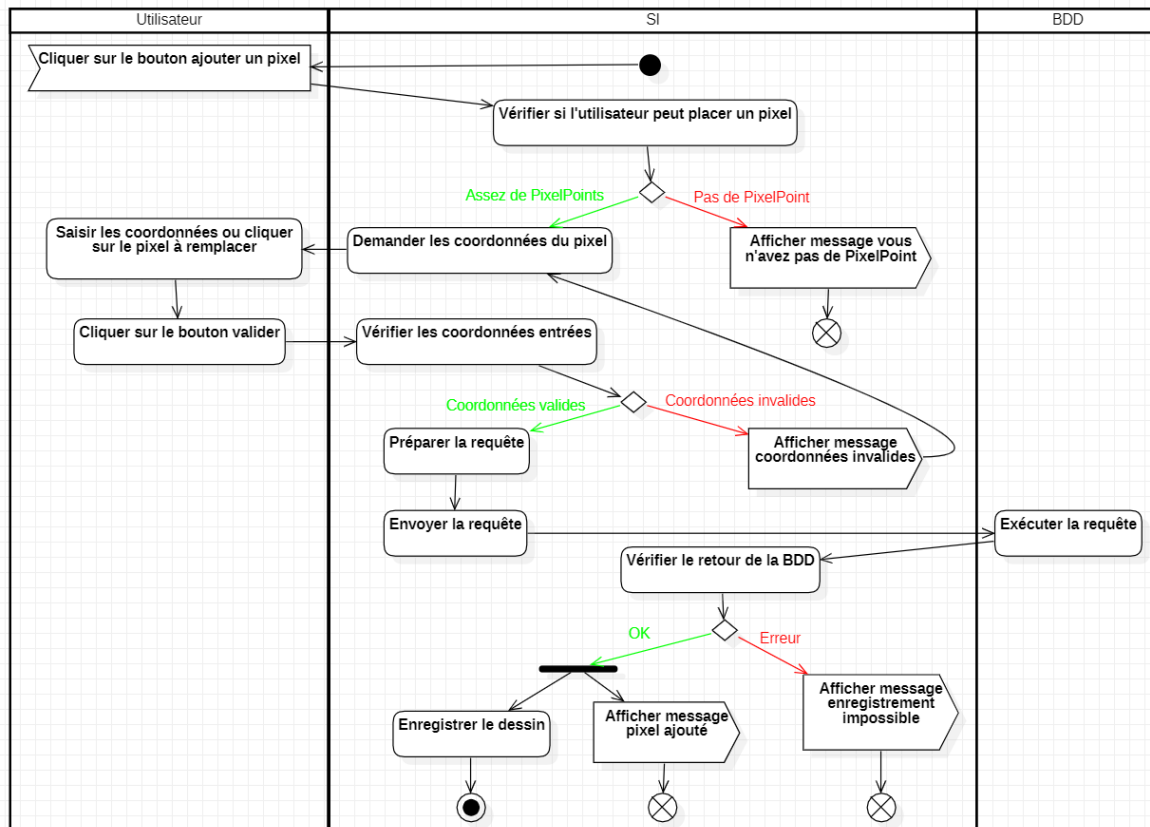
Le visiteur peut créer un compte ou se connecter.

L'utilisateur peut utiliser le site simplement, faire des quizz, utiliser ses points sur le dessin commun, modifier son profil, ...

L'administrateur s'occupera de créer les questions et les quizz, de modifier les comptes et de modérer le dessin commun.

L'utilisateur et l'administrateur partagent certains droits, surtout d'affichage ainsi que la déconnexion.

B. Diagramme d'activité



Le diagramme d'activité permet de visualiser les interactions entre le système **SI**, l'**utilisateur** et la **base de données**.

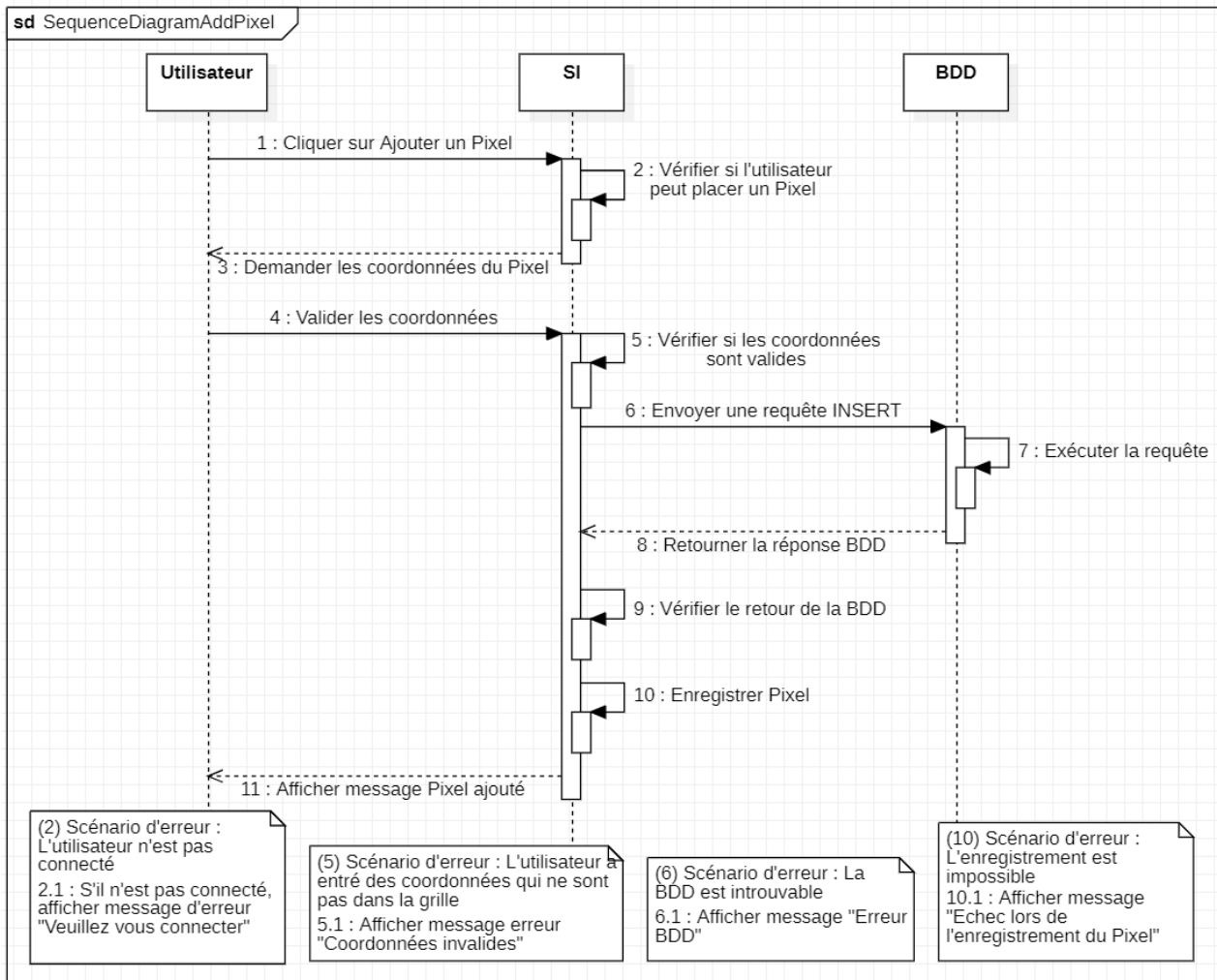
Voici le diagramme de l'activité « Ajouter un pixel » sur le dessin commun. On commence à l'état initial dans le système SI, représenté par une boule noire.

Tout d'abord le système va réagir à une action de l'utilisateur, ici si l'utilisateur « clique sur le bouton ajouter un pixel ». Démarre ensuite l'activité et s'en suit divers échanges entre le système, l'utilisateur et la BDD.

Les losanges indiquent un nœud, c'est à cet endroit que l'on peut avoir plusieurs scénarios possibles, ici les scénarios positifs sont indiqués en vert, et les scénarios négatifs en rouge. Une barre noire horizontale indique un « fork », une « fourchette », ici un événement provoque au moins 2 branches simultanément.

L'activité se termine soit avec des erreurs, au niveau d'un « flow final » (les ronds avec une croix), soit en se terminant normalement en arrivant à l'état final indiqué par un rond noir entouré d'un cercle.

C. Diagramme de séquence



Le diagramme de séquence permet de schématiser les scénarios de l'activité associée. Sur un diagramme de séquence, on lit à la verticale en suivant les lignes de vie « life line » et les rectangles verticaux représentent des actions en cours et s'appellent des « barres d'activation ». Le scénario se déroulant sans erreurs s'appelle le scénario nominal.

Les flèches indiquent des interactions entre les acteurs, on retrouve trois types de flèches :

- Les flèches vers la droite indiquent des messages synchrones, ce sont des demandes qui vont créer une barre d'activation et attendre une réponse avant de continuer l'activité.
- Les flèches vers la gauche sont les réponses reçues par un acteur et permettant de mettre fin à une barre d'activation.
- Les flèches qui reviennent sur elles-mêmes sont des « self messages », il s'agit d'une méthode gérée par l'acteur actuel.

Ici le diagramme de séquence correspond à l'activité « Ajouter un pixel » présenté juste au-dessus, donc on retrouve les trois mêmes acteurs : Utilisateur, Système SI et base de données BDD.

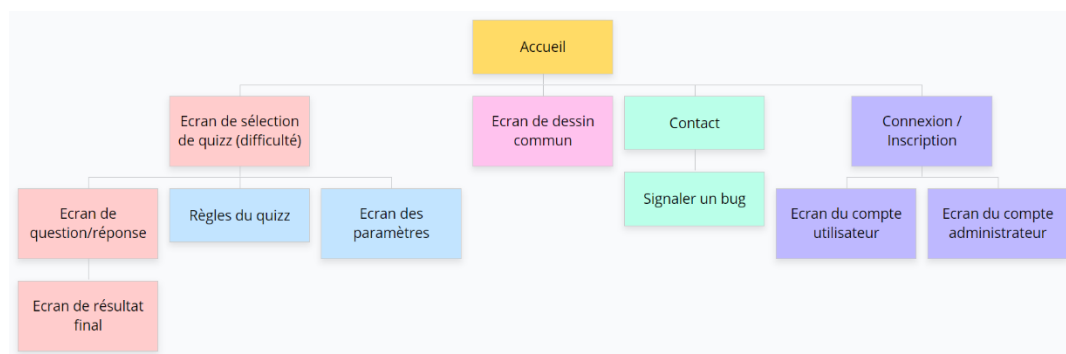
Les rectangles en bas indiquent des cas alternatifs ou d'erreur, les cas alternatifs permettent de voir des variantes du scénario nominal et les cas d'erreur permettent d'imaginer les scénarios stoppant l'activité.

Ici je possède 3 scénarios d'erreur :

- Le premier scénario d'erreur est que l'utilisateur n'est pas connecté.
- Le second scénario d'erreur se produit si l'utilisateur est connecté et essaye de placer un pixel sans avoir de points.
- Le troisième scénario d'erreur se produit si la base de données ne répond pas.

D. Arborescence

L'arborescence d'un site internet représente la structure hiérarchique et organisée des pages, sections et contenus du site. Elle définit la manière dont les informations sont organisées et accessibles pour les utilisateurs. L'arborescence peut être représentée comme une sorte de plan ou de carte qui montre les relations entre les différentes parties du site.



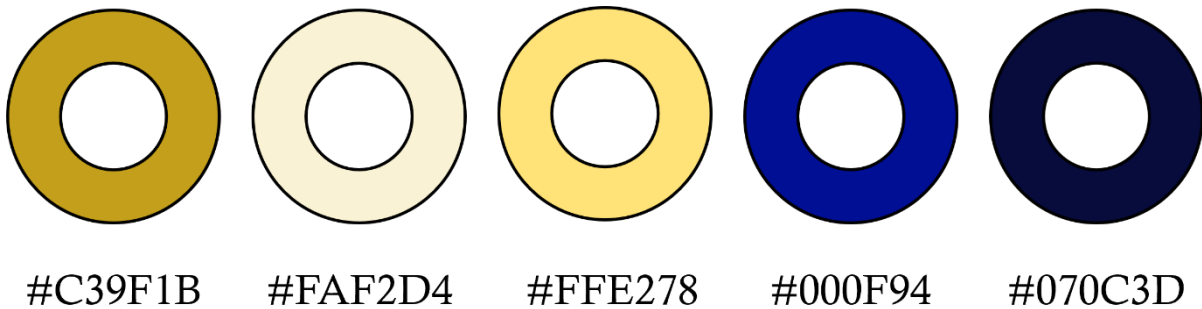
Arborescence du site

La racine est la page de départ, celle sur laquelle on arrive en premier lorsqu'on arrive sur le site internet. Ici il s'agit de la page « Accueil ».

E. Maquettage

1. Charte graphique

Les couleurs utilisées pour le site sont des teintes de jaune (#C39F1B, #FAF2D4 et #FFE278) et de bleu (#000F94 et #070C3D) principalement. Le bleu est une couleur symbolisant le savoir, la sérénité tandis que le jaune un peu terne comme ici rappelle un peu la couleur des parchemins, symbolisant aussi le savoir. Ici le choix notamment de la première couleur (#C39F1B) a été fait également pour ressembler et faire penser à la couleur de l'anneau de pouvoir, avec ce côté doré.



Les typographies utilisées seront Ringbearer (police des films du Seigneur des Anneaux) ainsi que Calisto MT qui possède un empattement faisant penser à des gardes d'épées, notamment sur le T.

RINGBEARER

UNE PETITE PHRASE EXEMPLE

Calisto MT

Une petite phrase exemple

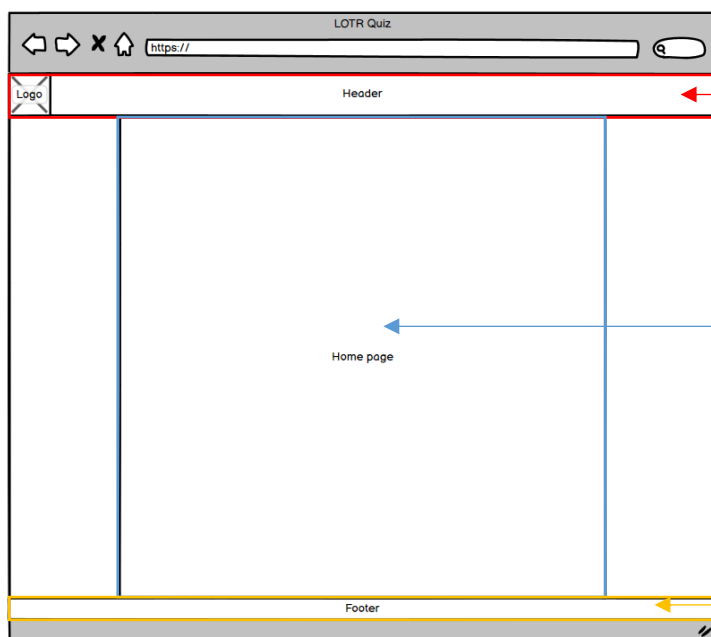
Le logo a été réalisé par Jaysön Dutheil, un dessinateur de mon entourage qui a gentiment accepté de me le faire, je l'en remercie donc ici. Ce logo représente Gandalf étudiant un vieux grimoire, Gandalf étant l'un des personnages les plus sages du Seigneur des anneaux et ayant beaucoup de savoir. Et ici il s'instruit davantage en étudiant ce grimoire.



2. Zoning

Le zoning représente le visuel global du site, mais de la manière la plus simplifiée possible, avec juste les emplacements pour les éléments.

Voici le zoning de la page d'accueil, les autres sont en [annexe A.1](#).



Tout en haut nous retrouvons le Logo et le header, qui contiendra les différents menus.

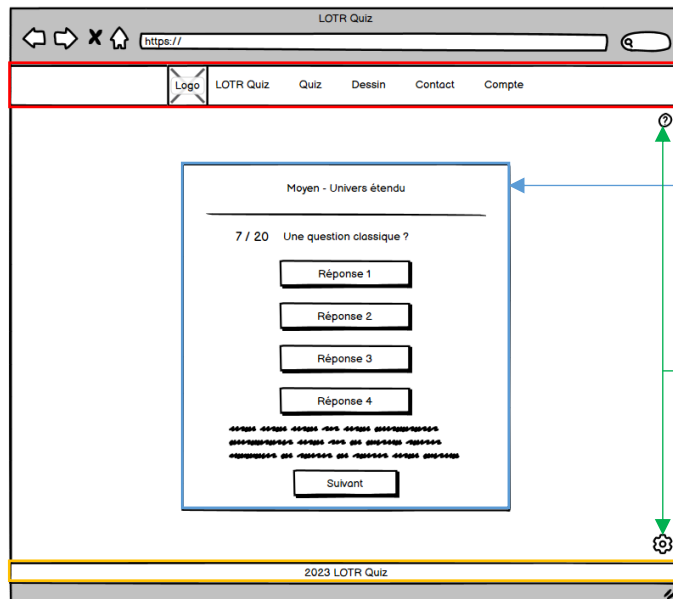
Ensuite nous retrouvons ici le corps de la page, qui contiendra un paragraphe explicatif du concept, une image, explication des règles,...

Enfin nous retrouvons le footer, contenant généralement les mentions légales, les conditions d'utilisation,...

3. Wireframe

Le wireframe montre également les éléments du site, mais en plus détaillé que le Zoning, les menus sont visibles, les boutons aussi, mais il manque encore un peu de précision, ce sera le rôle du Mockup que nous verrons juste après.

Ici je montre le wireframe de l'écran de question, les autres seront disponibles en [annexe A.2.](#)



En haut nous retrouvons le header de tout à l'heure, mais cette fois les menus sont visibles.

Ensuite nous retrouvons le cadre du quiz avec le niveau de difficulté ainsi que les questions/réponses.

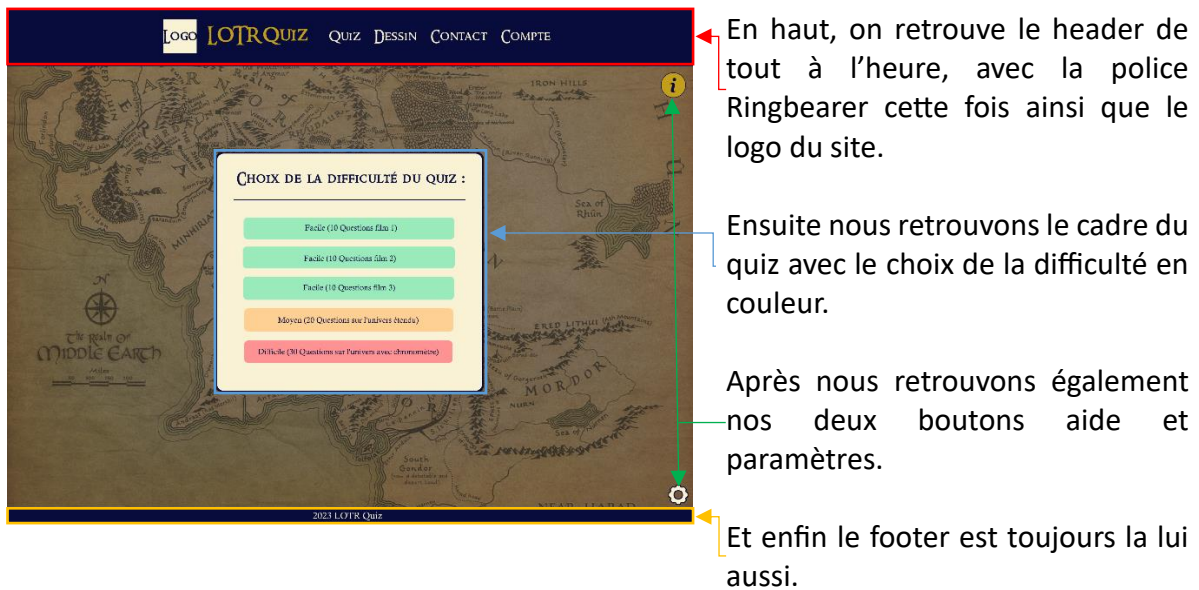
Après nous avons également deux boutons, l'un pour les règles/l'aide et l'autre pour les paramètres.

Et enfin nous avons le footer en bas.

4. Mockup

Le mockup, quant à lui, comme expliqué brièvement plus haut, correspond au visuel le plus détaillé que l'on puisse avoir, il doit correspondre quasiment à l'identique au visuel final du site ou de l'application. Les éléments présents dans le wireframe sont complétés de sorte qu'ils aient leur aspect final et les couleurs font leurs apparitions.

Voici le mockup de l'écran de choix de difficulté du quiz :



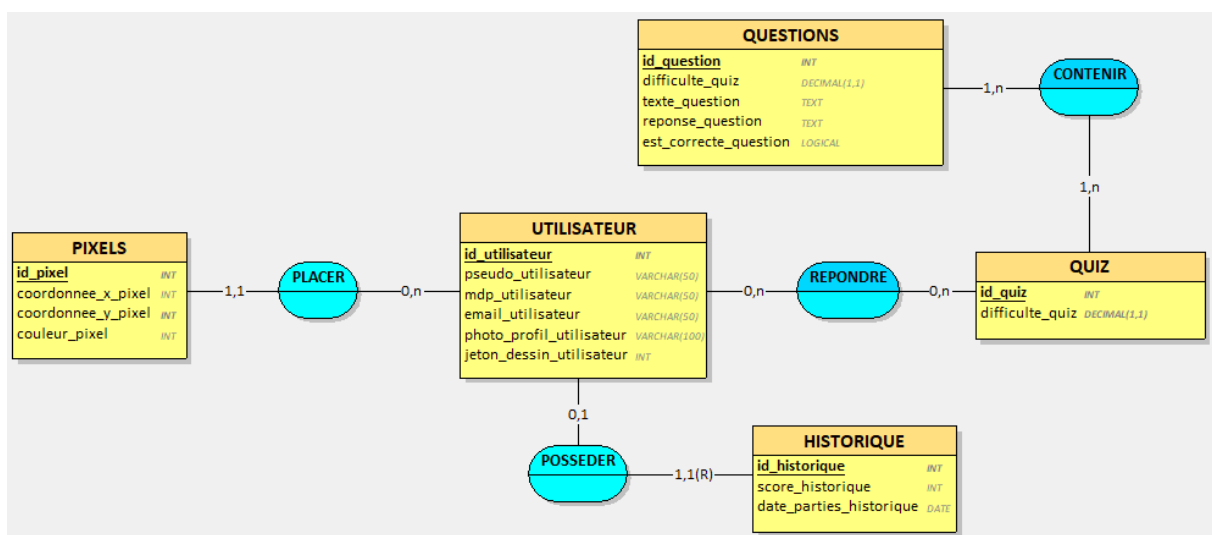
A noter aussi qu'un fond est apparu, il s'agit d'une carte de la Terre du Milieu assombrie pour faire un côté « parchemin » et pour que le fond n'attire pas trop l'attention.

IV. Conception

A. MCD – Modèle Conceptuel de Données

Le MCD est une représentation abstraite et visuelle des concepts et des relations qui existent dans un système d'information. Il vise à décrire la structure logique des données sans se préoccuper des détails techniques de leur implémentation. Les éléments clés du MCD comprennent les entités (les rectangles), les associations entre ces entités (ellipses bleues), les héritages (triangles verts), et les contraintes (ronds verts) qui régissent ces relations.

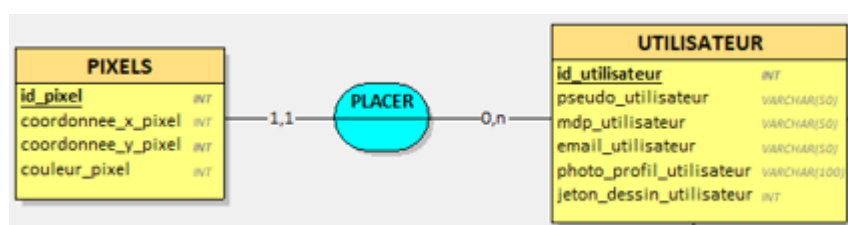
Le MCD est souvent utilisé dans le processus de conception de bases de données pour fournir une vision claire et compréhensible des données et de leurs relations avant de passer à la phase de conception physique de la base de données.



Dans mon projet, je possède 5 entités : Pixels, Utilisateur, Historique, Quiz et Questions.

Chaque entité possède une **clé primaire**, il s'agit de son id, toujours placé en première position au sein de l'entité, exemple avec l'entité QUIZ qui possède id_quiz.

Les cardinalités (0,1 ; 1,1 ; 0,n ; 1,n) représentent le nombre de relations possibles entre une entité et une association, exemple prenons cette situation :



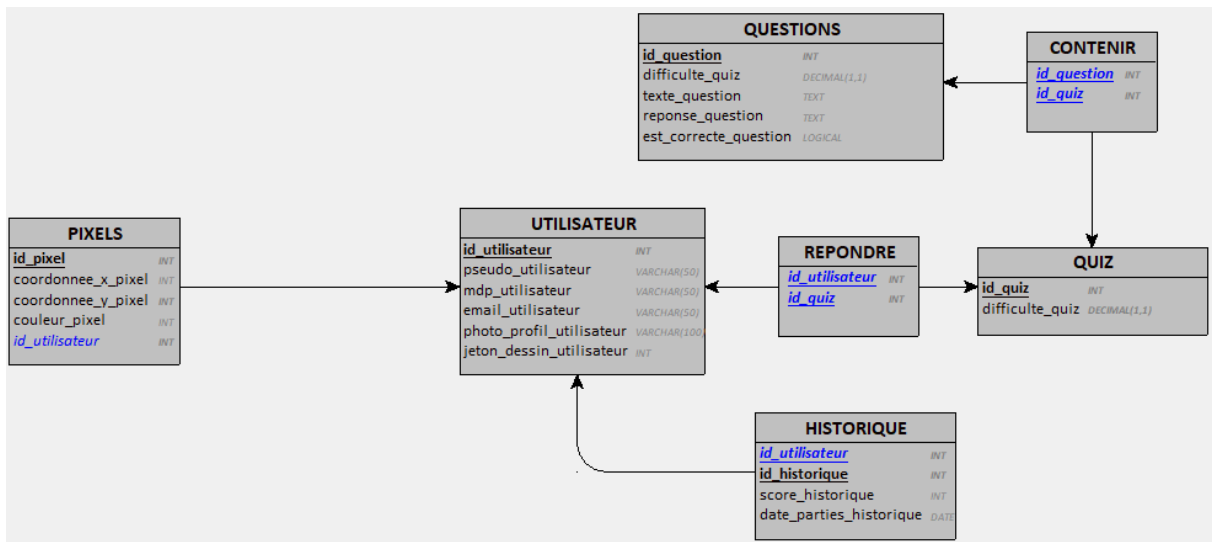
Ici on peut traduire en français par :

- « Un utilisateur peut placer 0 ou plusieurs pixels ».
- « Un pixel peut être placé par un seul et unique utilisateur ».

B. MLD – Modèle Logique de Données

Le Modèle Logique de Données (MLD) est une représentation plus détaillée et concrète des données d'un système d'information par rapport au Modèle Conceptuel de Données (MCD). Contrairement au MCD, le MLD prend en compte les contraintes de la technologie de gestion de bases de données utilisée (comme SQL) et se concentre sur la manière dont les données seront stockées physiquement.

Dans un MLD, les entités du MCD sont traduites en tables, les relations deviennent des clés étrangères, et les attributs sont détaillés avec des types de données spécifiques. Le MLD sert de base pour la création effective de la base de données. Il inclut également des éléments tels que les index, les contraintes d'intégrité, et d'autres aspects liés à la mise en œuvre technique. Ainsi, le MLD fait le pont entre le modèle conceptuel (MCD) et la réalité physique de la base de données.



C. Fonctionnalités

Je vais maintenant vous présenter les fonctionnalités présentes sur mon site une fois qu'il sera achevé, c'est-à-dire toutes les fonctionnalités que je voudrais ajouter avant de le rendre fonctionnels aux tests.

- **Gestion de compte**

Permet à l'utilisateur de gérer ses informations de compte (identifiant, mot de passe, image de profil).

- **Responsive**

Permet d'avoir un affichage cohérent du site peu importe l'appareil utilisé pour le voir.

- **Sélection de la difficulté des quiz**

Avec 5 choix de difficulté, les quiz sont adaptés à tous, expert ou non.

- **Historique des scores**

Les utilisateurs pourront voir sur leurs comptes leur historique des scores avec les derniers quiz effectués, la date et le score obtenu.

D. Code

Je vais à présent parler de tout le code que j'ai pu effectuer pour ce projet, que ce soit le front-end ou le back-end en présentant quelques parties de codes de mon projet, le code entier restant visible sur le projet GitHub suivant :

<https://github.com/Theagorn/ProjetFilRouge/tree/main/quiz>

1. Front-End

Afin de créer mon site de quiz, j'ai évidemment eu besoin de réaliser des pages HTML (HyperText Markup Language) pour la structure ainsi que du CSS (Cascading Style Sheets) pour l'aspect esthétique de mes pages internet. Je vais présenter un peu tout ça et tout le code sera disponible sur le lien GitHub juste au-dessus.

Utilisation du HTML :

Concernant le HTML, je vais présenter l'écran de sélection de la difficulté, puis je détaillerai le CSS utilisé pour cette même page.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Choix de la difficulté</title>
7      <link rel="stylesheet" href="style.css">
8  </head>
```

Tout d'abord et comme toute page HTML nous avons besoin de spécifier le `<!DOCTYPE html>` (qui précise la version d'HTML que la page utilise) ainsi que le contenu de la balise `<head>` qui permet de stocker les méta-informations, les liens, les scripts,...

`<meta charset="UTF-8">` spécifie l'encodage de la page, UTF-8 prenant en charge la plupart des caractères du monde.

`<meta name="viewport" content="width=device-width, initial-scale=1.0">` définit la manière dont la page doit être affichée sur différents appareils afin de gérer le responsive. En spécifiant `width=device-width`, la largeur de la page est définie égale à la largeur de l'appareil et `initial-scale=1.0` indique que le niveau de zoom initial doit être de 1.0 (pas de zoom).

`<title>Choix de la difficulté</title>` définit le titre de la page, qui sera affiché dans l'onglet du navigateur.

Enfin, `<link rel="stylesheet" href="style.css">` établit un lien vers la feuille de style `style.css`, permettant de séparer la structure HTML de la présentation CSS.

```

9  ✓ <body>
10 ✓   <div class = "app">
11     <h1>Choisissez la difficulté du quiz :</h1>
12 ✓   <div class = "quiz">
13     <h2 id = "question"></h2>
14 ✓   <div id = "answer-buttons">
15     <button onclick="saveDifficulty(1.1)" class = "difficulty1"><b>Facile
16       (10 Questions sur les films de Peter Jackson : La Communauté de l'Anneau)</b></button>
17     <button onclick="saveDifficulty(1.2)" class = "difficulty1"><b>Facile
18       (10 Questions sur les films de Peter Jackson : Les Deux Tours)</b></button>
19     <button onclick="saveDifficulty(1.3)" class = "difficulty1"><b>Facile
20       (10 Questions sur les films de Peter Jackson : Le Retour du Roi)</b></button>
21     <button onclick="saveDifficulty(2)" class = "difficulty2"><b>Moyen
22       (20 Questions sur l'univers entier)</b></button>
23     <button onclick="saveDifficulty(3)" class = "difficulty3"><b>Difficile
24       (30 Questions sur l'univers entier avec chronomètre de 10s par question)</b></button>
25   </div>
26 </div>
27 </div>

```

Une fois sorti de la balise `<head>` qui sert à indiquer les méta-informations, on arrive dans la balise `<body>` qui contient tout le contenu visible de la page web, ici se trouve ce que les utilisateurs voient.

Les balises `<div>` sont des conteneurs dans lesquels on peut avoir du contenu. A ces conteneurs on peut appliquer des attributs css (`class` ou `id`) que nous verrons plus loin et servant à améliorer le rendu visuel.

Les balises `<h1>` et `<h2>` sont des titres de niveaux 1 et 2, plus le niveau du titre est proche de 1, plus il sera gros, on peut aller jusqu'à `<h6>` soit 6 niveaux.

Enfin la balise `<button>` désigne des boutons cliquables par l'utilisateur, l'attribut `onclick=""` permettant d'effectuer une action si l'utilisateur clique sur l'un des boutons. Ici cela exécute la fonction `saveDifficulty()` en envoyant un nombre en paramètre de cette fonction en fonction du bouton cliqué par l'utilisateur. On remarque également la balise `` qui sert à mettre le texte contenu dans la balise en gras.

A propos de la fonction `saveDifficulty()`, vous pouvez la voir en [annexe VI B](#) car il s'agit de code Javascript, mais utile pour la compréhension de cette partie.

Utilisation du CSS :

Maintenant que nous avons présenté la structure de l'écran de sélection de la difficulté, passons maintenant au CSS de celui-ci et donc de son rendu visuel pour l'utilisateur.

Tout d'abord le CSS est un langage de style utilisé pour définir la présentation visuelle des documents HTML ou XML. Son rôle principal est de permettre la séparation de la structure d'une page web de sa présentation, facilitant ainsi la maintenance et la personnalisation du design. En bref, le CSS est utilisé pour appliquer des styles tels que les couleurs, les polices, les marges, les espacements, les positions et d'autres propriétés visuelles aux éléments d'une page web. Il permet aux développeurs de créer des mises en page attrayantes et cohérentes tout en maintenant une structure de code claire. Les règles CSS peuvent être incluses

directement dans le fichier HTML, liées en externe via un fichier séparé ou encore générées dynamiquement par des langages côté serveur.

Concernant le CSS de mon site, je vais expliquer rapidement une partie du CSS, dont le responsive, plutôt basique pour mon site car mon affichage de base est assez adapté pour mobile, mais j'ai quand-même un bout de code en responsive que je vais expliquer ici, ainsi que le CSS de mes boutons de choix de difficulté.

```
57 .difficulty1, .difficulty2, .difficulty3 {
58     color: #070c3d;
59     font-weight: 500;
60     width: 100%;
61     border: 1px solid #070c3d;
62     padding: 10px;
63     margin: 10px 0;
64     text-align: center;
65     border-radius: 4px;
66     cursor: pointer;
67     transition: all 0.3s;
68     font-size: 16px;
69 }
70
71 .difficulty1:hover{
72     background: #1e8048;
73     color: #faf2d4;
74 }
75
76 @media only screen and (max-width: 400px) {
77     .app {
78         width: 350px;
79     }
80     .btn, #question {
81         text-align: center;
82     }
83 }
```

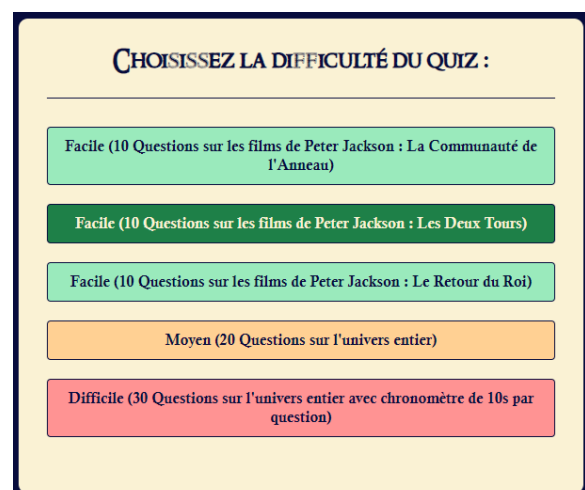
Tout d'abord, je vais expliquer rapidement les paramètres changés ici :

- color change la couleur du texte
- font-weight définit l'épaisseur de la police du texte
- width change la largeur du container (ici la totalité horizontalement)
- border ajoute une bordure autour de mon container
- padding ajoute des espacements internes dans le container
- margin ajoute un espace en haut et en bas
- text-align aligne le texte
- border-radius arrondit les coins
- cursor définit l'icone du curseur
- transition ajoute une transition
- font-size définit la taille de police

Ensuite, ligne 71 nous avons l'exemple d'une pseudo-classes :hover.

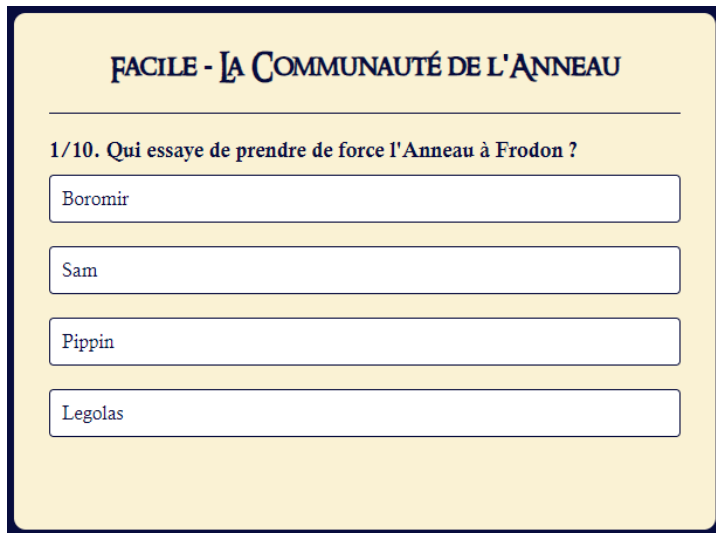
Elle permet d'effectuer une modification visuelle lors du survol de l'élément sélectionné (ici le survol de tout élément HTML ayant la classe .difficulty1.

L'action réalisée est le changement de couleur du fond en vert ainsi que le changement de couleur du texte en blanc teinté.



Rendu du survol d'une difficulté 1

Enfin, le dernier bloc de code est là pour gérer le responsive de notre page, c'est-à-dire l'adaptabilité sur d'autres appareils. Ici le code s'active pour les appareils avec une largeur d'écran maximale égale à 400px, et si c'est le cas, la taille de la fenêtre de quiz devient fixe (à 350px) et la question ainsi que les propositions de réponses se centrent dans leurs cases.



FACILE - LA COMMUNAUTÉ DE L'ANNEAU

1/10. Qui essaye de prendre de force l'Anneau à Frodon ?

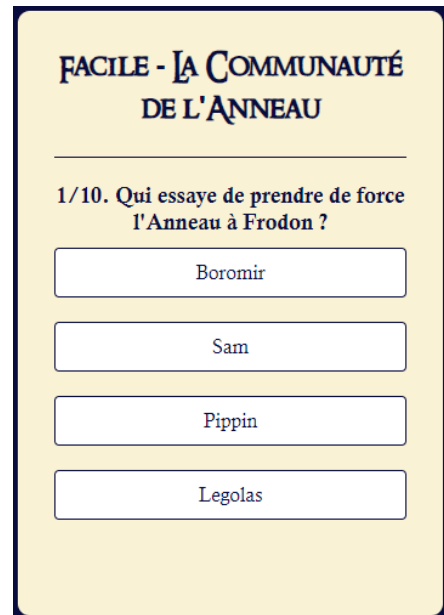
Boromir

Sam

Pippin

Legolas

Exemple d'une question sur PC



FACILE - LA COMMUNAUTÉ DE L'ANNEAU

1/10. Qui essaye de prendre de force l'Anneau à Frodon ?

Boromir

Sam

Pippin

Legolas

Exemple d'une question sur mobile

2. Back-End

Utilisation de MySQL et Node JS :

Dans le cadre de mon projet, j'ai donc eu besoin de gérer une base de données (BDD), pour ce faire j'ai installé MySQL sur mon ordinateur depuis le site officiel (mysql.com) dans un premier temps, puis il a fallu que j'installe un serveur pour faire la jonction entre ma base de données et mon site. Pour assurer ce rôle, j'ai également installé Node JS sur mon ordinateur depuis le site officiel (nodejs.org).

Pour la première utilisation je dois initialiser un serveur Express en passant pas Node JS, j'utilise donc la commande ``npm init -y`` qui me permet d'initialiser un projet Node JS par défaut. Ensuite j'utilise la commande ``npm install express mysql2`` qui me permet d'installer les modules express et mysql2.

- Le module express me permet d'avoir une infrastructure robuste pour la gestion des routes, des requêtes et des réponses, ...
- Le module mysql2 est nécessaire pour interagir avec une base de données MySQL comme celle que je possède et présentée juste en dessous, ce module me permettant donc d'exécuter des requêtes SQL, gérer des connexions à la base de données et manipuler les résultats.

```

PS D:\Codes\ProjetFilRouge\Projet\quiz\BDD 2> npm init -y
Wrote to D:\Codes\ProjetFilRouge\Projet\quiz\BDD 2\package.json:

{
  "name": "bdd-2",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

PS D:\Codes\ProjetFilRouge\Projet\quiz\BDD 2> npm install express mysql2

added 73 packages, and audited 74 packages in 7s

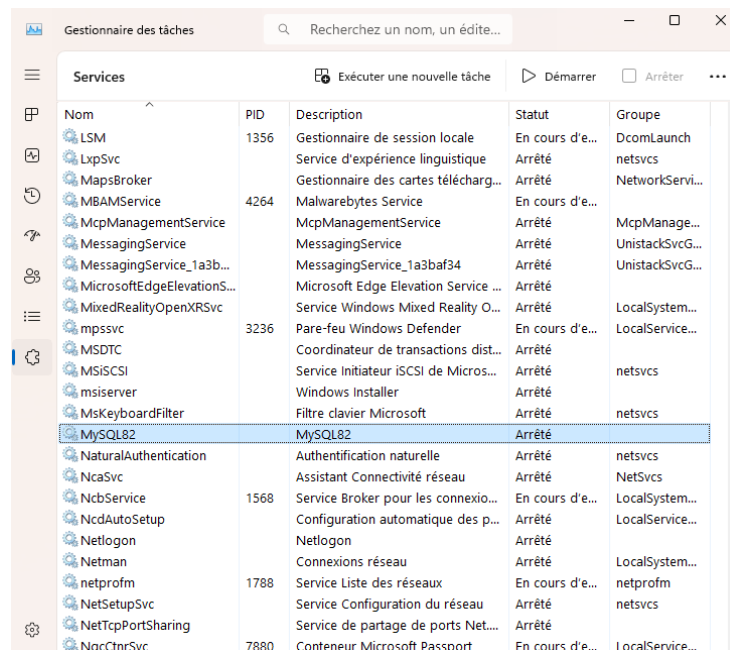
11 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS D:\Codes\ProjetFilRouge\Projet\quiz\BDD 2> |

```

Initialisation Node JS

Ensuite, à chaque utilisation, je dois lancer le service MySQL et Node JS. Pour lancer le service de MySQL, je dois passer par mon gestionnaire des tâches, aller dans l'onglet « Services », chercher le service qui se nomme MySQL82, faire clic droit sur le service puis sélectionner « Démarrer » (ou cliquer sur le bouton « Démarrer » en haut à droite de la fenêtre du gestionnaire des tâches, après quelques secondes il passera en « En cours d'exécution » et la BDD sera lancée.



Activation du service MySQL

Une fois ma base de données lancée, je dois donc lancer également mon serveur Node JS. Pour ce faire plusieurs choix s'offrent à moi, la seule condition est de passer par un terminal, ici je passe par le terminal de Visual Studio Code.

La première chose est de se rendre dans le serveur courant de mon projet Node JS, la ou j'ai installé les packages lors de la première utilisation grâce à la commande **cd** suivi de l'adresse du dossier.

Ensuite, on lance notre serveur Node JS utilisant Express, pour ce faire on utilise la commande ``node app.js`` ou `app.js` contient le code de notre serveur Express ainsi que nos routes utilisant les méthodes GET (pour demander des données au serveur) et POST (pour soumettre des données au serveur pour traitement).

Une fois ceci fait, je peux utiliser MySQL Workbench ou directement le terminal système afin de gérer mes requêtes SQL, voici par exemple la création de la base de données et de la table « utilisateur » :

```
1 • CREATE DATABASE IF NOT EXISTS lotrquiz;
2 • USE lotrquiz;
3
4 • CREATE TABLE IF NOT EXISTS utilisateur (
5     id_utilisateur INT AUTO_INCREMENT PRIMARY KEY,
6     pseudo_utilisateur VARCHAR(255) NOT NULL,
7     mdp_utilisateur VARCHAR(255) NOT NULL,
8     email_utilisateur VARCHAR(255) NOT NULL UNIQUE,
9     photo_profil_utilisateur VARCHAR(100),
10    jeton_dessin_utilisateur INT
11 );
```

Ici dans un premier temps on crée la base de données si elle n'existe pas déjà avec la ligne 1, avant de sélectionner la base de données à la ligne 2.

Ensuite à la ligne 4 on crée la table « utilisateur » si elle n'existe pas, ainsi que les colonnes de cette table suivant différentes options que nous allons voir ici :

- Tout d'abord la nature de la colonne, on retrouve **INT** (qui signifie que la colonne est un entier) ou **VARCHAR** (qui signifie que la colonne est une chaîne de caractère, suivi du nombre de caractères maximums de la chaîne de caractères).
- L'option **AUTO_INCREMENT** indique à la base de données d'incrémenter automatiquement la valeur de cette colonne à chaque insertion.
- La clause **PRIMARY KEY** définit cette colonne comme insertion, assurant ainsi un identifiant unique pour chaque utilisateur.
- La clause **NOT NULL** signifie que la colonne ne peut pas contenir de valeurs nulles.
- Et enfin la clause **UNIQUE** garantit que chaque adresse e-mail dans cette colonne est unique dans la table, empêchant ainsi l'inscription de plusieurs utilisateurs avec la même adresse email.

Voici notamment l'exemple d'une inscription sur le formulaire en ligne, ainsi que les données enregistrées :

Inscription

Nom :

Email :

Mot de passe :

Il s'agit d'un formulaire très simple, avec nom, email et mot de passe, puis en cliquant sur « S'inscrire » une requête POST est envoyée dans ma base de données, l'inscription est validée et les informations sauvegardées dans la base de données.

Ensuite, en regardant dans la base de données et en utilisant la commande ``SELECT * FROM utilisateurs;`` on peut récupérer les données des inscriptions réalisées, que voici :

| | id_utilisateur | pseudo_utilisateur | mdp_utilisateur | email_utilisateur | photo_profil_utilisateur | jeton_dessin_utilisateur |
|---|----------------|--------------------|-------------------|-----------------------|--------------------------|--------------------------|
| ► | 1 | Dark Vador | PAmidala | DVador@sw.com | NULL | NULL |
| | 2 | Davy Jones | Calips0 | DJones@pdc.com | NULL | NULL |
| | 3 | Gandalf | VousNePasserezPas | GandalfLeGris@tdm.com | NULL | NULL |
| * | NULL | NULL | NULL | NULL | NULL | NULL |

Ici on remarque qu'il y a eu 3 inscriptions, on voit donc l'ID qui s'incrémente à chaque nouvelle insertion, rendant chaque utilisateur unique, on peut voir le pseudo choisi, le mot de passe et l'adresse email. Comme vu plus haut, la photo de profil et les jetons de dessins peuvent être nuls lors de l'inscription car non essentiel, l'utilisateur aura sûrement une photo de profil par défaut et démarrera à 0 jetons.

Enfin, lorsque nous regardons un peu du côté du terminal, lorsque l'on réalise nos 3 inscriptions tout se passe bien, puis lorsqu'on essaye d'entrer un utilisateur autre que ceux que nous avons saisi nous rencontrons un échec de connexion et enfin lorsque l'on saisit l'un de nos trois utilisateurs inscrits, la connexion est effectuée.

```
Node.js v20.9.0
PS D:\Codes\ProjetFilRouge\Projet\quiz\BDD> node app.js
Serveur démarré sur le port 3000
Inscription réussie
Inscription réussie
Inscription réussie
Échec de la connexion
Connexion réussie
```

Utilisation de Javascript

Dans le cadre de mon projet, j'ai également eu besoin de gérer mon système de quiz, et pour ce, j'ai donc dû créer le quiz de A à Z en javascript. Je possède donc 2 fichiers javascripts, un pour tout l'aspect fonctionnel du quiz, et un autre où se trouve la classe contenant toutes les questions, leur difficulté, les différentes réponses avec la bonne qui est

répertoriée et enfin un texte de réponse pour appuyer la bonne réponse. J'aurais pu également stocker toutes mes questions/réponses dans la base de données et les appeler lors de requêtes, seulement au moment de développer cette partie, je n'étais pas encore des plus à l'aise avec le SQL, j'avais donc plutôt choisi la méthode en Javascript. Je vais parler un peu de ces deux fichiers en présentant une partie du script de mon quiz mais également la structure de mes questions dans le fichier les stockant.

Commençons par un bout du code de fonctionnement de mon quiz, je vais présenter la partie qui permet d'afficher une question.

```
// Fonction pour afficher une question
function showQuestion(){
  // Réinitialisation de l'état
  resetState();

  // Récupération de la question actuelle
  let currentQuestion = questionsActives[currentQuestionIndex];
  let questionNo = currentQuestionIndex + 1;
  questionElement.textContent = `${questionNo}/${totalQuestions}.
  ${currentQuestion.question}`;

  // Affichage des réponses possibles sous forme de boutons
  currentQuestion.answers.forEach(answer => {
    const button = document.createElement("button");
    button.textContent = answer.text;
    button.classList.add("btn");
    answerButtons.appendChild(button);
    if(answer.correct){
      button.dataset.correct = answer.correct;
    }
    button.addEventListener("click", selectAnswer);
  });
}
```

Nous avons donc ici la fonction `showQuestion()` qui nous permet, au sein de notre script, d'afficher la question actuelle et les boutons de réponse, et d'afficher un texte de justification de la réponse lorsque l'utilisateur clique sur une des réponses possibles, qu'elle soit vraie ou fausse.

Dans cette fonction, on commence par appeler une autre fonction, `resetState()` qui permet de vider tous les champs sur l'affichage pour la question, les boutons de réponse et le texte de justification, afin de pouvoir les remplacer par les informations de la nouvelle question. On se retrouve donc avec notre affichage vierge d'informations, comme ceci (pour l'exemple, la difficulté est Normal) :

Affichage vierge sans questions

Ensuite nous retrouvons un bloc qui récupère la question actuelle (`currentQuestion`) dans la liste des questions sélectionnées (`questionsActives`), passe à la question suivante (`questionNo`) et enfin affiche le numéro de la question actuelle sur le total de questions ainsi que la question actuelle.

Il faut noter qu'une question récupérée sera de cette forme :

| | |
|---|--|
| <code>{</code> | |
| <code> question: "",</code> | L'intitulé de la question |
| <code> difficulty: 2,</code> | La difficulté de la question |
| <code> trueAnswer: "",</code> | Le texte explicatif de la réponse |
| <code> answers:[</code> | Les réponses : |
| <code>{ text: "", correct: false},</code> | Les 4 propositions |
| <code>{ text: "", correct: true},</code> | Celle avec correct : <code>true</code> |
| <code>{ text: "", correct: false},</code> | est la bonne réponse |
| <code>{ text: "", correct: false},</code> | |
| <code>]</code> | |
| <code>}</code> | |

Le gros bloc suivant permet dans un premier temps d'afficher autant de boutons de réponse qu'il y a d'éléments dans `answers` (donc 4), en ajoutant les intitulés des réponses dans les boutons créés au fur et à mesure, ainsi que le statut de bonne réponse à celle l'étant. Et la dernière ligne de ce bloc permet de rendre les boutons cliquables et mémoriser ce choix dans `selectAnswer`.

NORMAL - UNIVERS ÉTENDU

1/20. Qui est le père de Legolas ?

Elwe

Haldir

Féanor

Thranduil

Exemple d'une question

V. Conclusion

Ce projet a été une bonne expérience afin de mettre en pratique au fur et à mesure de la formation les connaissances que j'ai acquies, que ça soit le front-end, le back-end, du maquettage, de l'UML,... Sur toutes les matières étudiées, j'ai forcément rencontré des difficultés parfois sur ces neuf mois, même en ayant eu une formation pré-qualifiante avant celle-ci (ce qui m'a quand-même permis d'avoir plus de facilité dans certaines matières, notamment l'algorithmique ou le réseau).

Parmi les matières qui m'ont posé problème, je relève surtout l'UML et le SQL avec lesquels il m'a fallu plusieurs relectures, explications et aide pour enfin m'en sortir. Maintenant elles me bloquent moins, surtout le SQL et à vrai dire je trouve ça même plutôt utile et intéressant !

Inversement également pour certaines matières, que j'ai beaucoup aimé de a à z, notamment l'algorithmique, car j'en fais depuis longtemps maintenant et j'ai toujours apprécié cela, que j'ai des difficultés ou non. J'ai également beaucoup apprécié le Java, bien que la matière ait été étudiée très tardivement et très rapidement, le contenu reste intéressant selon moi.

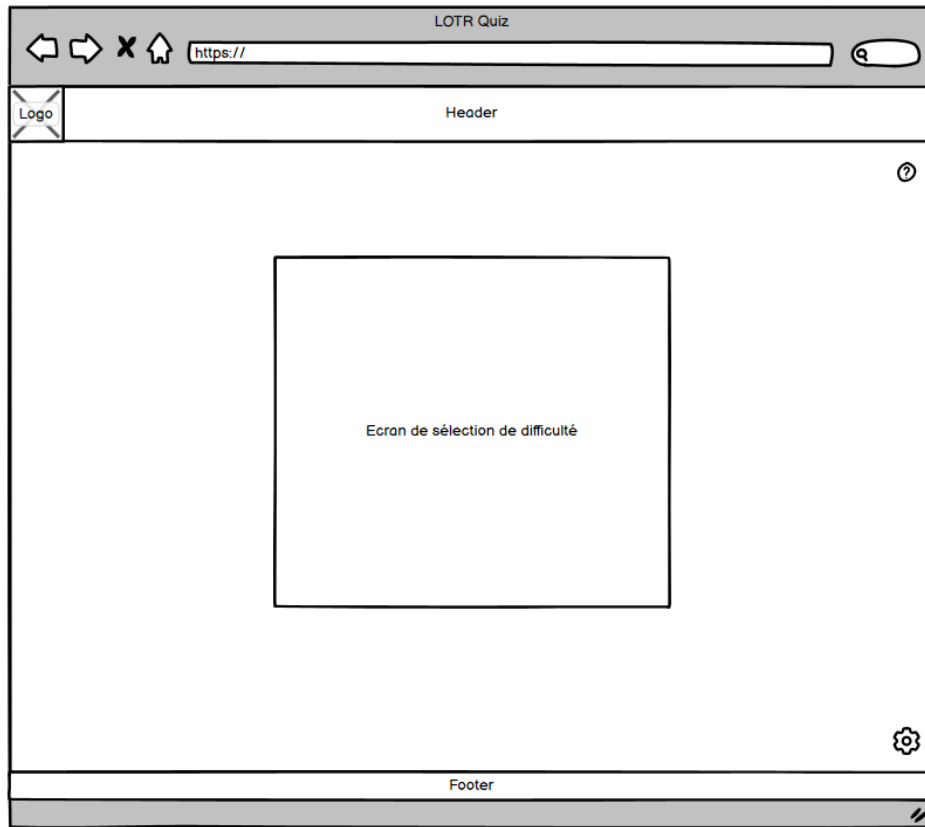
Pour la suite du projet, je compte finir le site tel qu'il était pensé, donc faire la liaison entre le système de connexion et mon quiz, rajouter la fonctionnalité du dessin commun, la minuterie pour les questions difficile, l'historique des parties, rajouter de nouvelles questions,... et ensuite je verrai une fois tout cela fait si je rajouterai de nouvelles fonctionnalités ou non.

Enfin, je souhaite remercier certaines personnes, sans qui j'aurais eu du mal à avancer ce projet, donc mes collègues de formation, ma famille pour une aide précieuse sur les questions, mon ami Jaysön Dutheil pour le logo et ChatGPT, très utile pour desceller les petites erreurs de code ou pour aider à comprendre ses crashes.

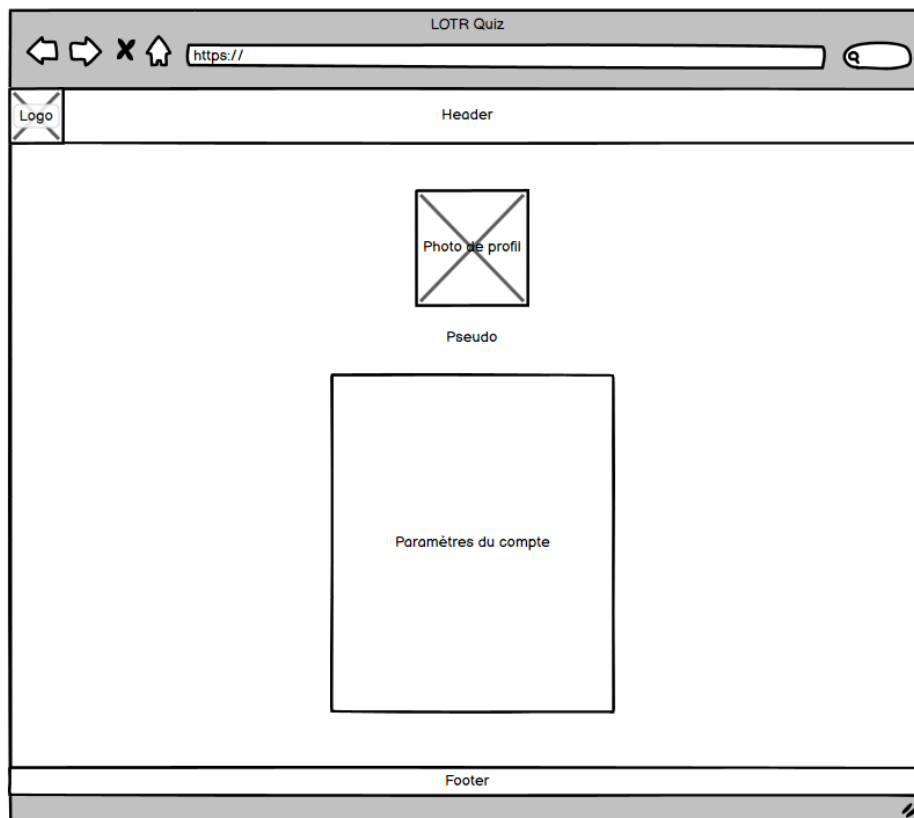
VI. Annexes

A. Maquettage complet

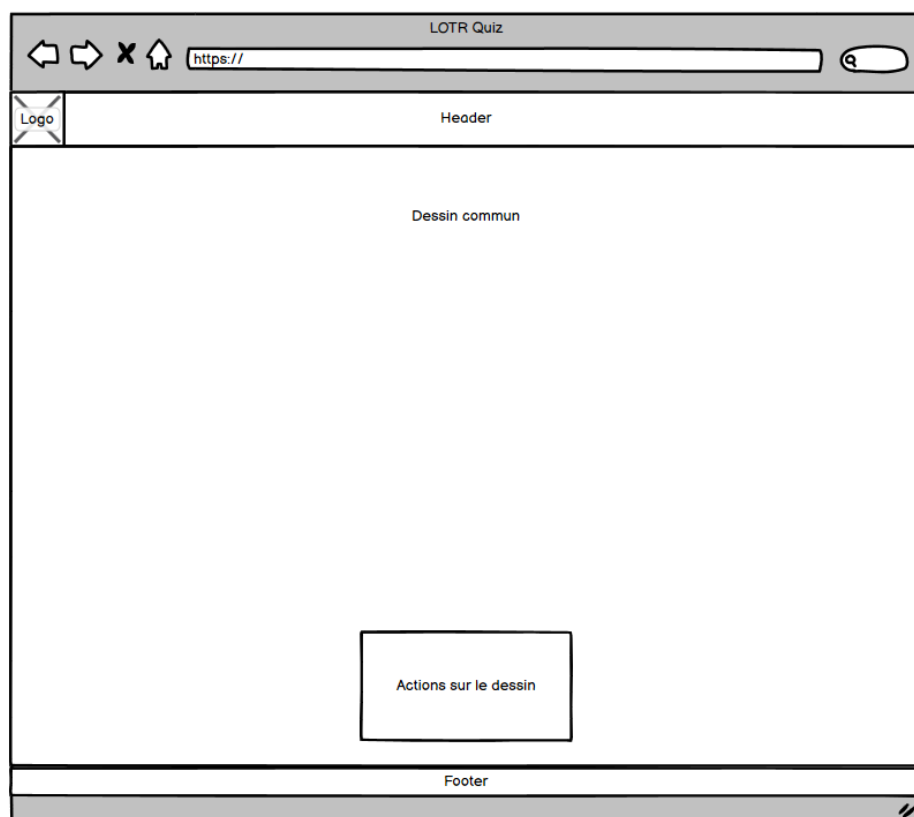
1. Zoning



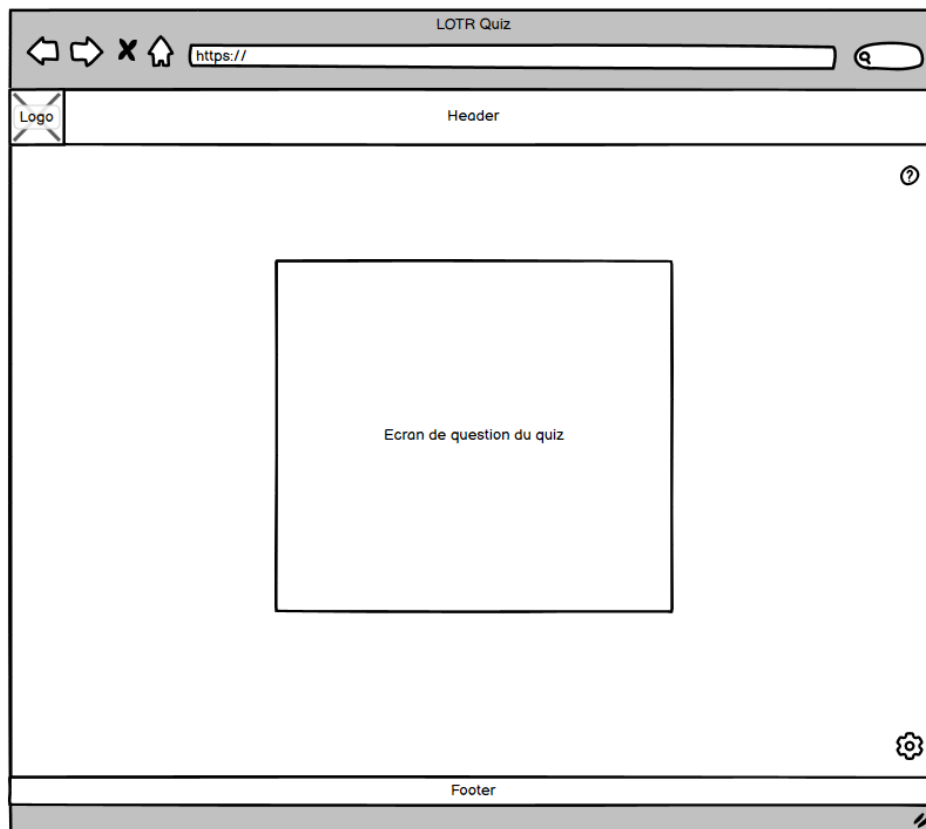
Zoning choix de la difficulté



Zoning compte

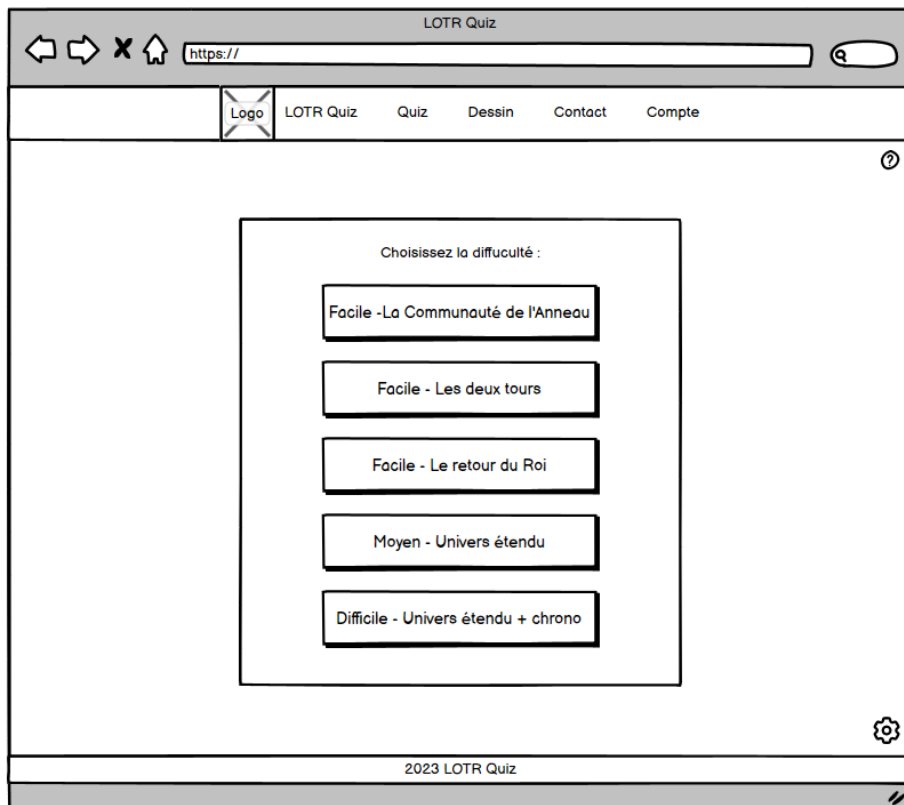


Zoning dessin

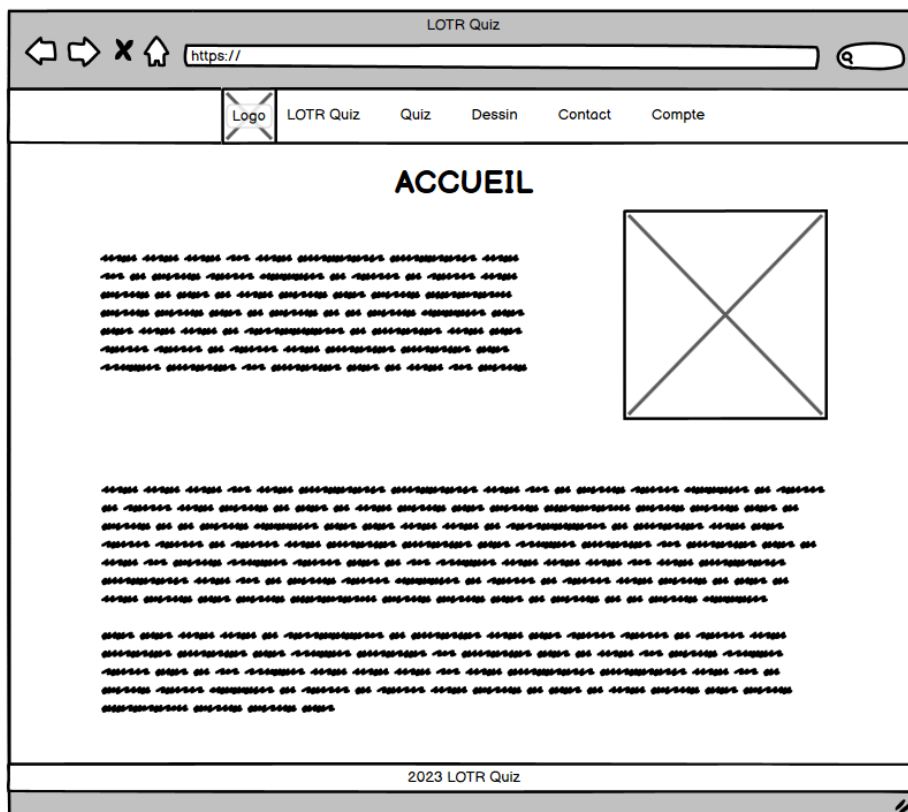


Zoning écran des questions

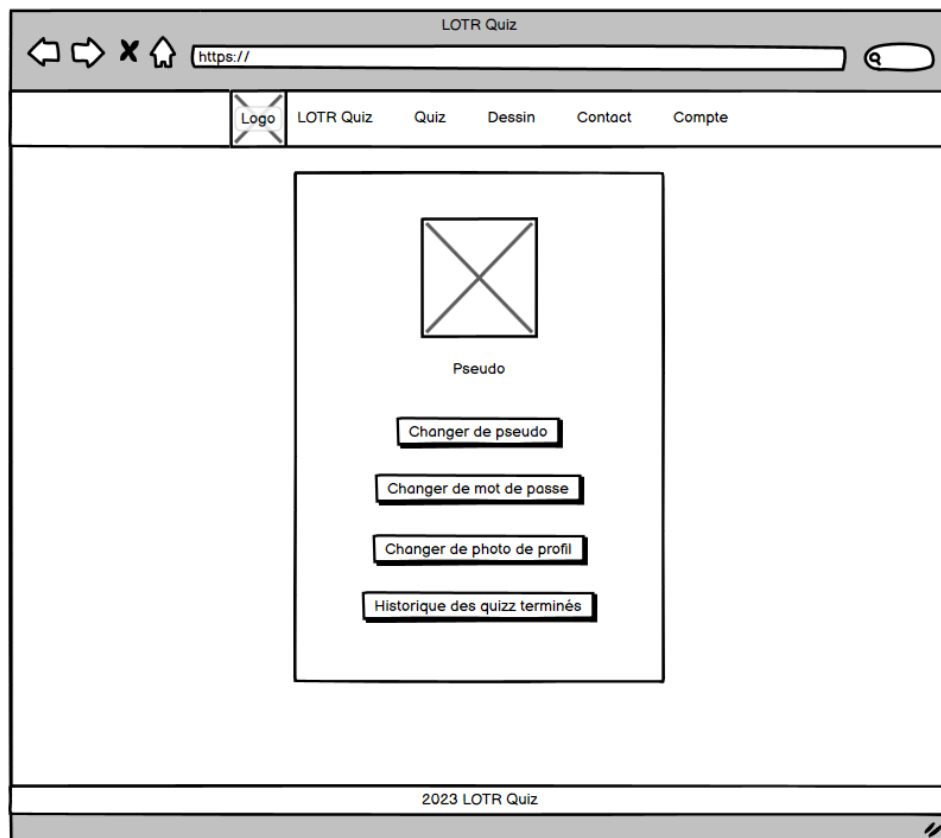
2. Wireframe



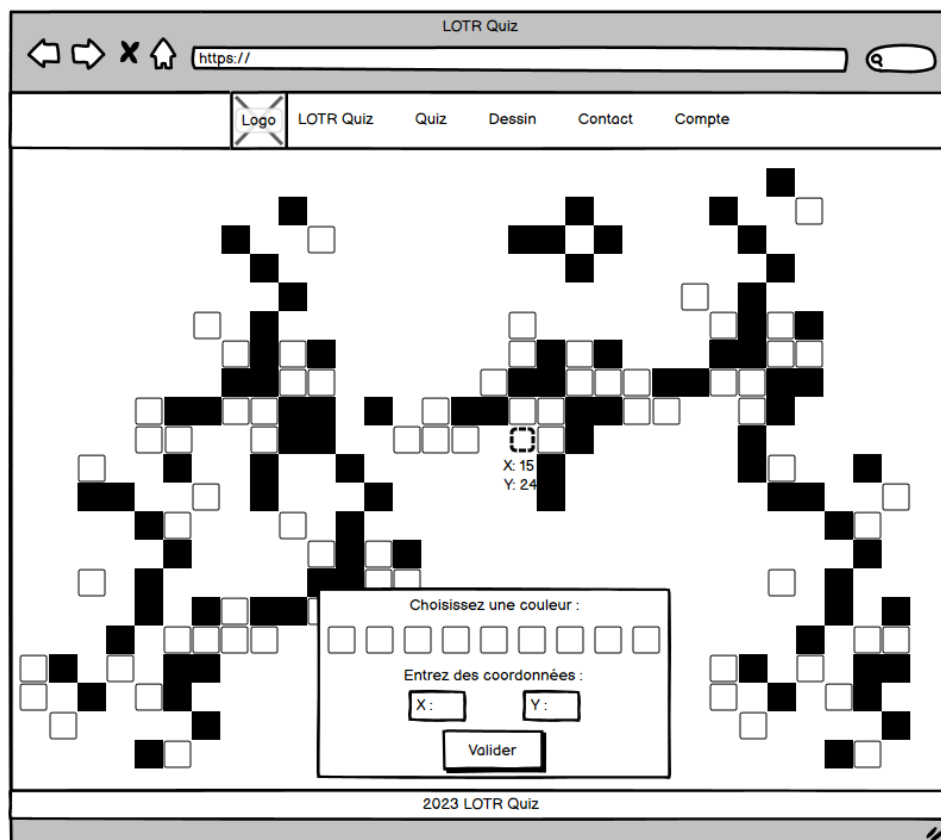
Wireframe écran de choix de difficulté



Wireframe accueil



Wireframe compte



Wireframe dessin

B. Fonction saveDifficulty()

```
24     <script defer>
25         function saveDifficulty(difficulty) {
26             localStorage.setItem("selectedDifficulty", difficulty);
27             window.location.href = "quiz.html"; // Redirection vers la page du quiz
28         }
29     </script>
30 </body>
31 </html>
```

La balise `<script>` est utilisée pour inclure du code JavaScript dans une page HTML. L'attribut `defer` indique au navigateur de retarder l'exécution du script jusqu'à ce que l'analyse du HTML soit terminée, ce qui est souvent utilisé pour garantir que le script est exécuté dans l'ordre correct.

La fonction `saveDifficulty()` prend en entrée une difficulté (`difficulty`) et la stocke dans le stockage local du navigateur grâce à `localStorage`. Cela permet de conserver la difficulté sélectionnée même si l'utilisateur actualise la page ou revient plus tard.

Enfin, `window.location.href = "quiz.html";` redirige vers la page « `quiz.html` » après avoir enregistré la difficulté sélectionnée.

C. Documentation traduite

Dans ma recherche de documentations sur le SQL suite à mes difficultés, je suis tombé sur cette page expliquant le fonctionnement de MySQL Workbench, et plus précisément le Tableau de bord des performances, qui me semble être un outil très pratique pour veiller à la maintenance de la base de données, j'ai donc décidé de traduire la moitié de cette page.

Lien de la page : <https://dev.mysql.com/doc/workbench/en/wb-performance-dashboard.html>

Affichez les statistiques de performance du serveur dans un tableau de bord graphique. Pour afficher le tableau de bord, ouvrez un onglet de requête, puis cliquez sur **Tableau de bord** dans la zone Performances de la barre latérale du navigateur lorsque l'onglet **Gestion** est sélectionné. La figure suivante montre la disposition des informations dans l'onglet **Administration - Tableau de bord**.

Etat du réseau

Cette section présente les statistiques du trafic réseau envoyé et reçu par le serveur MySQL par l'intermédiaire des connexions client. Les points de données comprennent **le trafic réseau entrant**, **le trafic réseau sortant** et **les connexions client**.

Etat de MySQL

Cette option met en évidence les principales statistiques d'activité et de performance du serveur MySQL. Les points de données incluent l'efficacité du **cache ouvert de la table**, les **instructions SQL exécutées** et le nombre (par seconde) d'instructions SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER et DROP.

Version originale :

View server performance statistics in a graphical dashboard. To display the dashboard, open a query tab and then click **Dashboard** from the Performance area of the Navigator sidebar with the **Management** tab selected. The following figure shows the layout of the information within the **Administration - Dashboard** tab.

Network Status

This highlights statistics for network traffic sent and received by the MySQL server over client connections. Data points include the **Incoming Network Traffic**, **Outgoing Network Traffic**, and **Client Connections**.

MySQL Status

This highlights the primary MySQL server activity and performance statistics. Data points include the **Table Open Cache** efficiency, **SQL Statements Executed**, and counts (per second) for SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, and DROP statements.