|  |  |  |
| --- | --- | --- |
| Travail pratique 2 |  | *15 points* |

# Développement d’un site de jeux en ligne

**Contexte :**

Vous allez développer un site de jeux en ligne permettant aux utilisateurs de jouer à des jeux classiques. Ce site comprendra plusieurs jeux que vous devrez implémenter avec React. De plus, l’accès à ces jeux devra être protégé par une authentification via Auth0.

Le travail s’effectuera en équipe de deux ou trois personnes. Une grande marge de manœuvre vous est accordée pour ce travail. Assurez-vous de bien définir votre stratégie avant de vous avancer trop loin. Pour la durée du travail, l’enseignant assumera les rôles de client et de chef d’équipe. Dans le doute, référez-vous à lui!

**Objectif :**

Créer une application web avec React permettant à l'utilisateur de se connecter via Auth0, puis de jouer à l'un des jeux proposés : Yahtzee, Démineur, Échecs ou Horloge. Vous choisirez l’un des jeux à développer. Le site devra offrir une interface intuitive et réactive.

**Exigences :**

1. Authentification avec Auth0 :

* L'accès au site doit être protégé par un système d'authentification via Auth0.
* Les utilisateurs doivent pouvoir se connecter avec un compte (par exemple, via un e-mail et mot de passe ou un autre moyen fourni par Auth0).
* Une fois authentifié, l'utilisateur pourra accéder à l'interface de jeux.

1. Jeux à implémenter :

* Yahtzee : Un jeu de dés où le but est de réaliser des combinaisons spécifiques pour marquer des points.
* Démineur : Un jeu de réflexion où l'objectif est de dévoiler toutes les cases sans toucher de mines.
* Échecs : Implémentation du jeu de société des échecs, avec toutes les règles classiques.
* Horloge : Un jeu de cartes où les cartes sont disposées en forme d'horloge, et le but est de les déplacer jusqu’à leur position correcte.

1. Fonctionnalités générales :

* Après authentification, l'utilisateur sera dirigé vers une page d'accueil où il pourra choisir un jeu.
* Une fois le jeu choisi, l'utilisateur peut commencer une nouvelle partie, voir ses scores et éventuellement recommencer une partie.
* Les jeux doivent être fonctionnels et respectueux des règles classiques de chaque jeu.
* L'interface doit être responsive et agréable à utiliser.

1. Aspects techniques :

* L'application devra être construite avec React.
* Vous utiliserez React Router pour gérer la navigation entre les différentes pages de l'application.
* L’authentification sera gérée via Auth0.
* Pour le développement du jeu, vous devrez respecter les bonnes pratiques de React et organiser correctement votre code (composants, hooks, gestion d'état).

1. Bonus (facultatif) :

* Ajouter un tableau de scores pour chaque jeu, stocké dans le localStorage et affiché dans votre application.

**Particularités par jeu :**

Selon le jeu que vous décidez d’implémanter, vous devez suivre les particularités suivantes :

1. Yahtzee :
   * Vous devez supporter l’ensemble des règles du jeu classique (<https://en.wikipedia.org/wiki/Yahtzee>)
   * Vous devez permettre de jouer de 1 à 4 personnes;
   * Le nom des joueurs doit être choisi à partir des choix retournés par <https://jsonplaceholder.typicode.com/users>;
2. Démineur :
   * Vous devez supporter l’ensemble des règles du jeu classique (<https://en.wikipedia.org/wiki/Minesweeper_(video_game)>)
   * Vous devez supporter les trois difficultés classiques (facile - 8x8, 10 mines, intermédiaire - 16x16, 40 mines, expert - 24x24, 99 mines)
   * Vous devez supporter le pointage et fournir une horloge pour calculer le temps de la partie. Lorsque le joueur réussi une partie, il doit choisir un nom parmi ceux retournés par <https://jsonplaceholder.typicode.com/users>;
3. Échec :
   * Vous devez supporter l’ensemble des règles du jeu classique (<https://en.wikipedia.org/wiki/Chess>);
   * La partie doit se jouer entre deux joueurs.
   * Le nom des joueurs doit être choisi à partir des choix retournés par <https://jsonplaceholder.typicode.com/users>;
4. Horloge :
   * Vous devez supporter l’ensemble des règles du jeu classique (<https://gamerules.com/rules/clock-patience/>) (exemple d’application : <https://www.solitaireparadise.com/games_list/clock-solitaire.html>);
   * La partie se joue à un seul joueur.
   * Utiliser l’API suivante pour gérer votre paquet de cartes : <https://deckofcardsapi.com/>
5. Autre :
   * Vous souhaitez développer un autre jeu? Parlez-en à votre enseignant! Gardez-en tête que votre jeu doit utiliser une API.

**Contraintes techniques :**

* Utilisez uniquement les modules React vus en classe;
* Utilisez uniquement les hooks React vus en classe;
* Ne remettez pas de répertoire build et node\_modules;
* Pas d'utilisation d'intelligence artificielle;
* L’utilisation de useEffect est interdite sauf pour appeler fetch lors d’un premier affichage ;

**Évaluation :**

|  |  |
| --- | --- |
| **Critères** | **Points** |
| Fonctionnalités : | |
| * Implémentation correcte des fonctionnalités individuelles et globales demandées | /60 |
| * Convivialité de l’application | /10 |
| Qualité globale du code (typage, lisibilité, modularité, indentation, commentaires) | /30 |
| Bonus : Hébergement de l’application en ligne | /5 |
| Bonus : Implémentation du tableau de scores | /5 |
| **Total** | **/100** |

La proportion des points attribuée à la qualité globale du code ne pourra pas dépasser la proportion des points atteinte pour les fonctionnalités. Par exemple, si une personne cumule 35 points pour les fonctionnalités (50% de 70 points possibles) elle ne pourra pas avoir plus de 15 points pour la qualité du code (50% de 30 points possibles).

Les propositions des étudiants peuvent être évaluées en entrevue après la remise du travail.

Qualité globale du code : Démontrez vos compétences en programmation en produisant du code de haute qualité. Une attention particulière sera portée sur la réutilisation de vos composants ainsi que sur le respect des principes DRY et SRP.

Convivialité de l’application : respectez les bonnes pratiques d’utilisabilité vues en classe.

Le non respect des contraintes techniques vous donnera immédiatement la note 0.

**Remise : 27 janvier 2025 avant minuit**

* Un MIO par équipe envoyé à l’enseignant contenant les membres de l’équipe, le jeu choisi par chaque membre et un lien vers le dépôt git de votre projet.

**Remise : 9 février 2025 à minuit**

* Aucun code sur votre dépôt git ne sera pris en compte après la date de remise.
* Un MIO par équipe contenant un lien vers votre application hébergée.