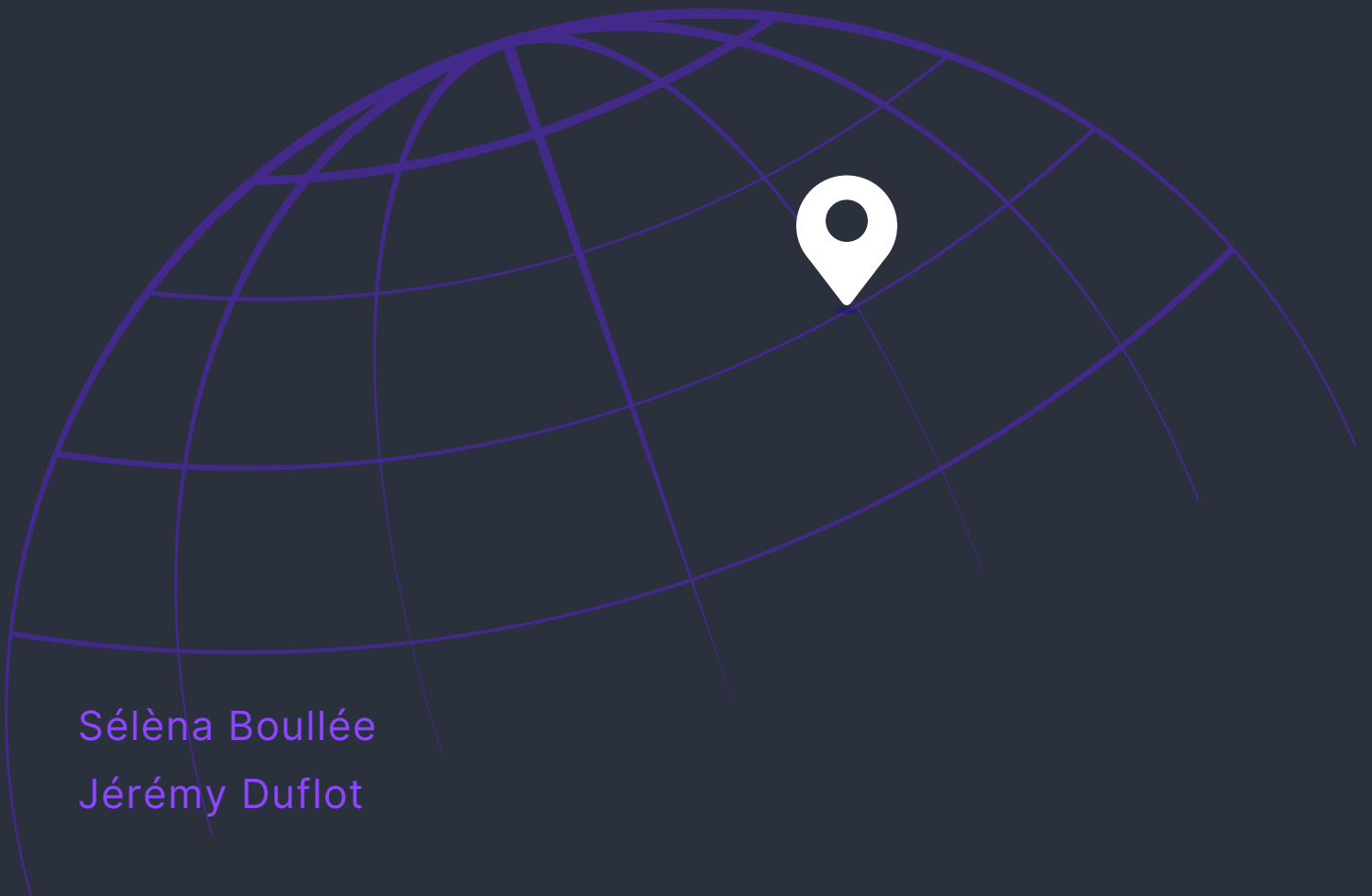


JUIN 2024

AtlasMystère

PROJET NEXT.JS

<https://album-travel.vercel.app/home>



Sélène Boullée
Jérémy Duflot



SOMMAIRE

- 01** Notre idée
- 02** Fonctionnalités et documentation utilisateur
- 03** Modèle MCD
- 04** Problèmes rencontrés et évolution du projet
- 05** Perspectives d'évolution





NOTRE IDÉE

01

Création d'un quiz avec Next.js basé sur le concept du jeu "Qui est-ce ?", où l'objectif est de deviner un **pays secret**. L'utilisateur sélectionne un pays sur une carte interactive, et l'interface lui fournit des indices sur des critères comme la superficie, la région, le nombre d'habitants et le continent.

Exemple:

"Le pays secret a deviné est la Serbie"

Tentative 1 : L'utilisateur clique d'abord sur la France

→ L'interface affiche que la superficie du pays recherché est plus petite, que le nombre d'habitants est plus faible, et que le continent est bien l'Europe mais que la région n'est pas l'Europe de l'ouest.

Tentative 2 : L'utilisateur clique ensuite sur l'Irlande

→ L'interface affiche que la superficie du pays recherché est plus grande, que le nombre d'habitants est plus important, et que le continent est bien l'Europe mais qu'il ne s'agit pas du Nord de l'Europe.

Tentative 3 : L'utilisateur rentre ensuite la Serbie

→ L'interface affiche qu'il a remporté le jeu en 3 tentatives, et lui annonce son temps qu'il pourra aller comparer avec les autres utilisateurs



FONCTIONNALITÉS ET DOCUMENTATION UTILISATEUR

PARTIE 1

Contenu du projet

Authentification et autorisations : Système d'inscription et de connexion pour les utilisateurs. Possibilité de se déconnecter.

Affichage de la Carte et Intégration du Jeu :

- Utilisation de la bibliothèque Leaflet pour afficher une carte interactive.
- Récupération des données GeoJSON à partir du référentiel GitHub de Johan, [world.geo.json/master/countries.geo.json](https://github.com/johan/world.geo.json/master/countries.geo.json), pour tracer les frontières des pays et les rendre cliquables.

Interaction sur la Carte

- Les pays sont rendus cliquables sur la carte.
- Si on appuie sur un pays c'est qu'on pense que le pays secret est peut être celui-ci
- Si la réponse est incorrecte, des indices sur le pays secret sont affichés, aidant ainsi l'utilisateur à deviner.
- Si la réponse est correcte, le jeu enregistre le nombre de tentatives et le temps passé pour deviner le pays.
- Le pays devient rouge sur la carte si ce n'est pas lui

Page des scores: les données de chaque parties des deux modes de jeu sont enregistrées en base de donnée et afficher sous la forme de classement.

Base de données: Utilisation de PostGress avec Vercel

Plateforme: Vercel pour le déploiement continu et l'hébergement.

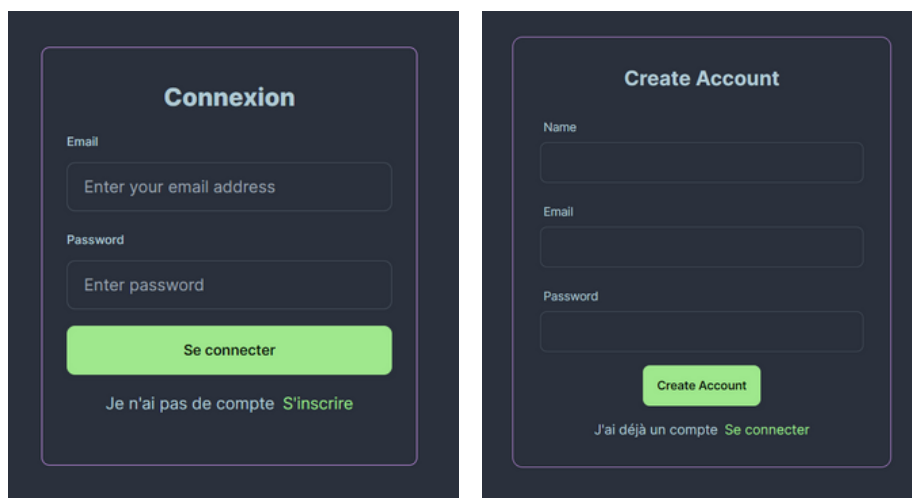


FONCTIONNALITÉS ET DOCUMENTATION UTILISATEUR

PARTIE 2

Etape 1 - La connexion/inscription

Avant de pouvoir jouer, l'utilisateur doit créer un compte et se connecter. Lors de la création du compte, il doit fournir un nom ou pseudo, une adresse e-mail, et un mot de passe incluant des majuscules, des minuscules, des chiffres et des caractères spéciaux. Une barre d'indication affiche la complexité du mot de passe au fur et à mesure de sa saisie. Ce compte permet à l'utilisateur d'être enregistré dans les classements. Sans compte, il serait impossible d'identifier véritablement le joueur et d'attribuer correctement les scores.



The image displays two side-by-side screenshots of a dark-themed user interface. The left screenshot, titled 'Connexion', features input fields for 'Email' (with placeholder 'Enter your email address') and 'Password' (with placeholder 'Enter password'), a green 'Se connecter' button, and a link 'Je n'ai pas de compte S'inscrire'. The right screenshot, titled 'Create Account', features input fields for 'Name', 'Email', and 'Password', a green 'Create Account' button, and a link 'J'ai déjà un compte Se connecter'.

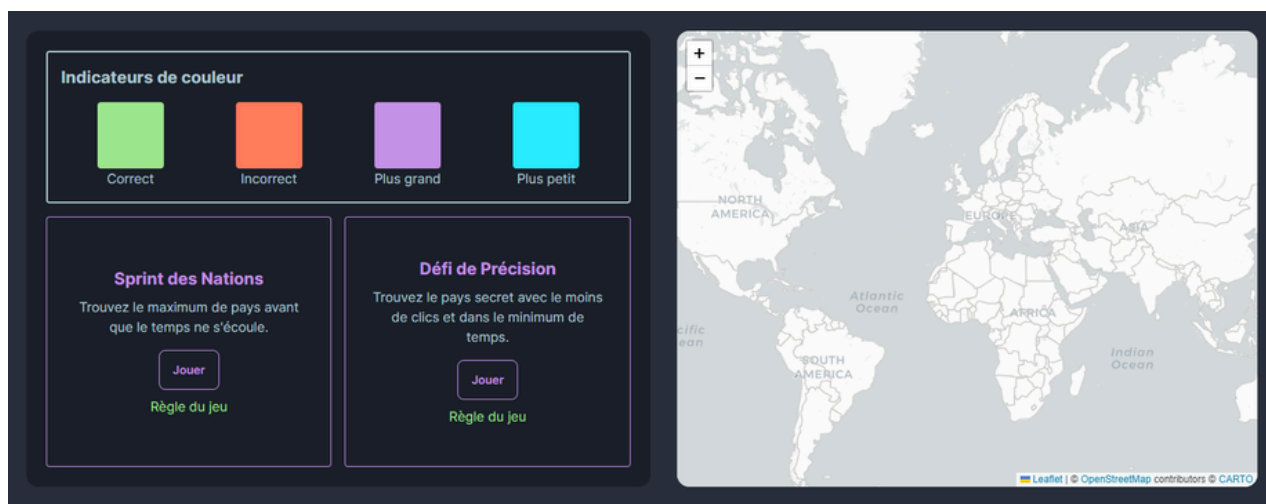
Etape 2 - Page home et modes de jeu

- La section de gauche est composée de la légende, elle vient expliquer les indices qui seront fournis à l'utilisateur pendant une partie. En dessous on trouve un menu permettant de choisir le mode de jeu, c'est cette section qui affichera les résultats de chaque click de l'utilisateur et lui donnera le score à la fin.
- La section de droite correspond à la carte interactive où l'utilisateur pendant une partie devra cliquer sur les pays. Lorsque le pays n'est pas le bon il s'affiche en rouge, si le pays est bon il s'affiche en vert avant de disparaître.

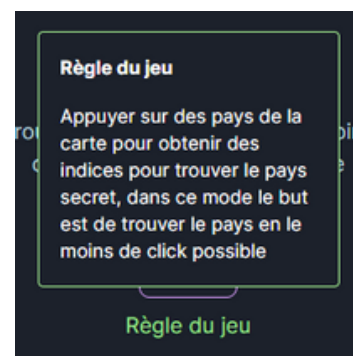
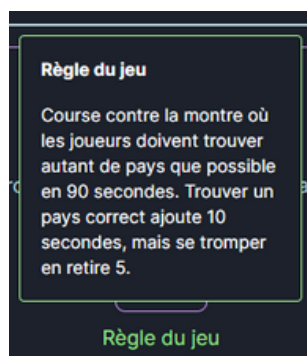


FONCTIONNALITÉS ET DOCUMENTATION UTILISATEUR

PARTIE 3



En survolant les textes des règles du jeu, les utilisateurs obtiennent des informations supplémentaires sur les différents modes de jeu. Ce petit détail améliore l'expérience utilisateur, un petit rappel des règles leur sera proposé après avoir cliqué.



Depuis toutes les pages après connexion l'utilisateur peut naviguer avec la barre de navigation ainsi que de se déconnecter

Etape 3 - Les deux modes de jeu

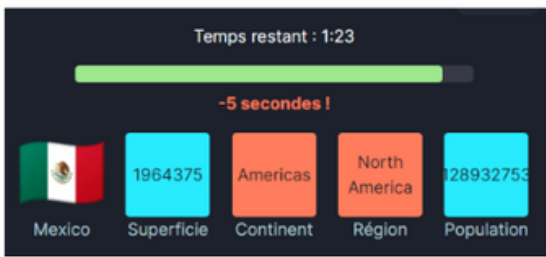

L'utilisateur peut choisir entre deux modes de jeu

- Le **Sprint des nations**, comme décrit il s'agit d'un jeu de rapidité, l'objectif est de trouver le plus de pays avant que le temps s'écoule. Quand l'utilisateur clique sur jouer, le timer se déclenche, initialement de 90 secondes, dès que l'utilisateur clique sur un mauvais pays, il perd -5s, en revanche s'il trouve le bon pays il gagne 30s.



FONCTIONNALITÉS ET DOCUMENTATION UTILISATEUR

PARTIE 4

Si l'utilisateur clique sur le mauvais pays	Si l'utilisateur clique sur le bon pays
	

Objectif : Deux éléments composent le score final pour déterminer un classement. Premièrement, le nombre de pays trouvés avant que le temps s'écoule. En cas d'égalité, le nombre de "pénalités", c'est-à-dire le nombre de pays cliqués qui n'étaient pas la bonne réponse, détermine la position du score dans le classement.

- Le **défi de précision**, il faut cliquer sur les pays de la carte pour obtenir des indices qui vous aideront à découvrir le pays secret. Dans ce mode de jeu, l'objectif est de trouver le pays secret en utilisant le moins de clics possible. Plus vous êtes précis et efficace, meilleur sera votre score.



Objectif : Deux éléments composent le score final pour déterminer un classement. Premièrement, le nombre de clics pour trouver le pays, en cas d'égalité c'est le temps qu'a pris le joueur à trouver le pays qui déterminera la position dans le classement.



FONCTIONNALITÉS ET DOCUMENTATION UTILISATEUR

PARTIE 5

Etape 4 - Consulter les scores

Classement Sprint des nations				
Rang	Utilisateur	Nb pays devinés	Nb pénalités	Date
1	Lenny	7	43	19/06/2024
2	user 1	6	38	19/06/2024
3	Lenny	5	34	19/06/2024
4	Lenny	5	34	19/06/2024
5	Lenny	5	34	19/06/2024
6	Lenny	5	36	19/06/2024
7	Lenny	4	28	19/06/2024
8	Lenny	4	29	19/06/2024
9	Lenny	4	31	19/06/2024
10	anonymeSel	3	22	23/06/2024
Précédent		Suivant		

Classement Défi de précision				
Rang	Utilisateur	Temps pour trouver	Nb de clicks	Date
1	user 1	0:03	3	19/06/2024
2	Lenny	0:08	3	19/06/2024
3	anonymeSel	0:33	4	25/06/2024
4	Lenny	0:06	5	20/06/2024
5	user 1	0:16	5	19/06/2024
6	valentin (le best)	0:20	5	19/06/2024
7	anonymeSel	0:34	5	23/06/2024
8	Totoleboss34%	0:15	6	19/06/2024
9	anonymeSel	0:34	6	25/06/2024
10	valentin (le best)	0:51	6	19/06/2024
Précédent		Suivant		

À tout moment, l'utilisateur peut choisir de consulter les classements. Ces tableaux s'appuient sur les scores enregistrés à la fin de chaque partie, affichant uniquement les 50 meilleurs scores. Les deux types de jeu ne sont pas mélangés dans les classements, un sujet qui sera abordé dans l'évolution de la base de données dans la partie 3, intitulée Modèle MCD.

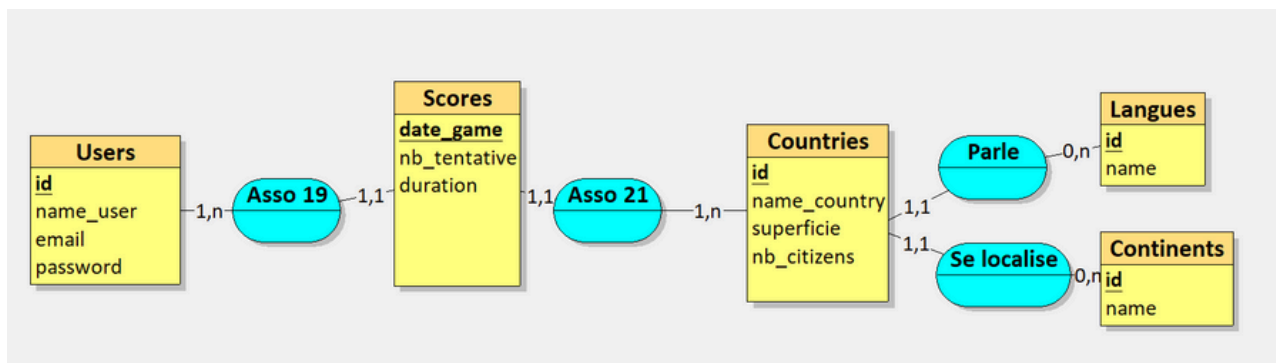
Remarque : la requête SQL associée limite le nombre de données renvoyées pour éviter une surcharge. Un tri est également appliqué en fonction des critères et objectifs de chaque type de partie, décrits plus haut.



MODÈLE MCD

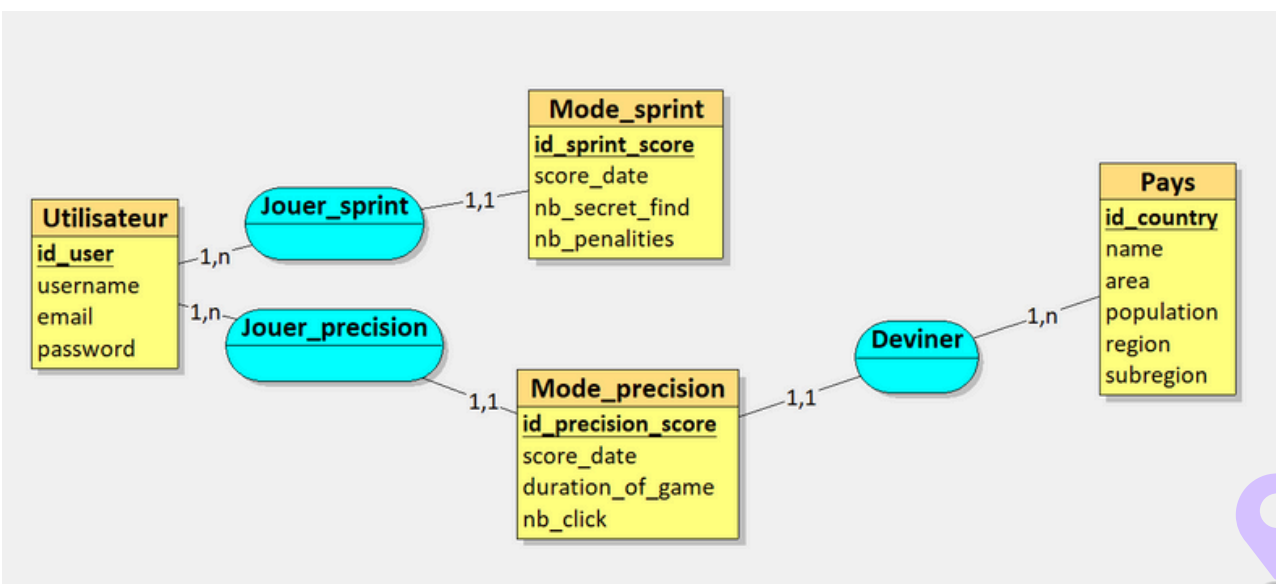
03

Au départ, notre base de données servait à enregistrer les scores du jeu, reliant chaque utilisateur à un pays avec un score. Nous pensions initialement que les pays partageraient des données communes, comme l'indice de la langue. Cependant, nous avons finalement jugé cet indice insuffisamment pertinent et avons fait évoluer notre base de données en conséquence.



Evolution de la bdd pour s'adapter aux deux modes de jeu

Avec l'ajout de nouveaux modes de jeu, les données de score varient désormais selon les parties. Les attributs des deux types de jeux sont différents, seul l'attribut score_date est commun. Pour simplifier la gestion des données, nous avons préféré créer deux tables distinctes plutôt que d'utiliser des tables héritées.





PROBLÈMES RENCONTRÉS

04

API

L'api la plus réputée et référencée est restCountries, qui permet à l'inverse des autres api, fournit des informations précises sur chaque pays.

<https://restcountries.com/v3.1/all?fields=name,population,region,area>

Nous avons découvert par la suite que cette api n'avait pas renouvelé le certificat SSL, le support de contact de l'api a annoncé que celle-ci n'était plus fonctionnelle puisqu'ils ont décidé de la passer payante sous un autre nom de domaine. Par inscription nous avons le droit à 250 requêtes à cette API (countryapi). Nous avons donc choisi d'utiliser celle-ci malgré tout.

Pour réduire le nombre de requêtes inutiles, on a ajouté une condition pour vérifier que le dernier pays cliqué n'est pas le même que le précédent, et ainsi ne pas rappeler l'API.

```
layer.on('click', async function () {  
  if (!activePlay) return; |
```

Pour optimiser notre efficacité malgré les restrictions sur le nombre de requêtes API, nous avons décidé de stocker l'API_KEY dans le fichier .env.local. Cette mesure vise principalement à renforcer la sécurité de notre système.

Après avoir fait fonctionner notre jeu, nous avons pris la décision d'améliorer notre organisation, en déplaçant les données de l'api dans une table. Cette décision nous permet à terme de limiter le nombre de requêtes à l'API, améliorant ainsi la performance et la disponibilité de notre jeu. Pour cela nous avons créé un script exécutable depuis le terminal, il vient récupérer toutes les données de l'API countryapi, et insère les données qui nous intéressent une seule et unique fois.

```
"seed": "node -r dotenv/config ./scripts/seed.js"
```



PROBLÈMES RENCONTRÉS

04

API

Un autre conflit que nous avons rencontré concerne les intitulés des pays. Nous utilisons en effet deux API : l'une pour récupérer les informations nécessaires pour placer les zones cliquables sur la carte au format GeoJson, et l'autre pour obtenir des informations sur les pays, comme la démographie.

Les noms renvoyés lors des clics n'étaient pas toujours les mêmes que ceux enregistrés dans notre base de données, ce qui posait de gros problèmes pour le gameplay. Par exemple, si l'utilisateur tombait sur Eswatini et cliquait correctement sur le pays, il n'y aurait jamais de victoire, car le nom du pays était différent lors de la comparaison.

Pour résoudre ce problème, nous avons créé un script qui modifie les noms de tous les pays identifiés comme problématiques.

```
const { db } = require('@vercel/postgres');

async function updateCountryNames(client) {
  try {
    const updateQueries = [
      client.sql`UPDATE countries SET name = 'Swaziland' WHERE name = 'Eswatini'`,
      client.sql`UPDATE countries SET name = 'United States of America' WHERE name = 'United States'`,
      client.sql`UPDATE countries SET name = 'Czech Republic' WHERE name = 'Czechia'`,
      client.sql`UPDATE countries SET name = 'Macedonia' WHERE name = 'North Macedonia'`,
      client.sql`UPDATE countries SET name = 'Democratic Republic of the Congo' WHERE name = 'DR Congo'`,
      client.sql`UPDATE countries SET name = 'The Bahamas' WHERE name = 'Bahamas'`,
      client.sql`UPDATE countries SET name = 'United Republic of Tanzania' WHERE name = 'Tanzania'`,
      client.sql`UPDATE countries SET name = 'Guinea Bissau' WHERE name = 'Guinea-Bissau'`,
      client.sql`UPDATE countries SET name = 'East Timor' WHERE name = 'Timor-Leste'`,
      client.sql`UPDATE countries SET name = 'Republic of Serbia' WHERE name = 'Serbia'`,
    ];

    await Promise.all(updateQueries);

    console.log('Updated country names');
  } catch (error) {
    console.error('Error updating country names:', error);
    throw error;
  }
}
```

Le nom des pays n'était pas le seul point de divergence entre les deux API. Les données du GeoJson étaient incomplètes et des pays comme l'Andorre ou le Liechtenstein n'étaient pas cliquables, bien qu'ils puissent être désignés comme pays secrets. Nous avons d'abord envisagé de mettre une condition basée sur la superficie pour le choix du pays secret, mais après quelques tests, nous avons opté pour un critère de population. Désormais, un pays doit avoir plus de 500 000 habitants pour être sélectionné comme pays secret.



PROBLÈMES RENCONTRÉS

04

Doublons useEffect

À partir de React 18.0, une nouvelle fonctionnalité appelée Strict Mode a été introduite. Cette fonctionnalité a été conçue pour aider les développeurs à détecter et à corriger les mauvaises pratiques potentielles.

L'un des effets secondaires de l'utilisation de Strict Mode est qu'elle peut parfois provoquer des comportements inattendus avec `useEffect`. Lorsque Strict Mode est activé, React peut volontairement exécuter deux fois certaines fonctions, y compris les fonctions de rendu et les effets tels que `useEffect`.

Pour régler ce comportement inattendu, nous avons rajouté une condition d'initialisation pour éviter que le `useEffect` boucle à chaque arrivée sur les pages.

Exemple de l'usage de cette condition dans la création de la carte Leaflet

```
useEffect(() => {  
  const initializeMap = () => {  
    lastCountryClickedRef.current = "";  
    if (mapRef.current === null) {  
      mapRef.current = L.map('map').setView([30.505, -0.09], 2);  
      L.tileLayer(  
        'https://{s}.basemaps.cartocdn.com/light_all/{z}/{x}/{y}{r}.png',  
        {  
          attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors &copy;';  
        }  
      ).addTo(mapRef.current);  
    }  
  }  
});
```



PROBLÈMES RENCONTRÉS

04

Connexion à la base de donnée

Nous avons rencontré des problèmes de configuration initiale avec le script `seed.js`, qui était basé sur un tutoriel de Next.js et incluait des utilisateurs de test insérés via un script `placeholder.js`. Pour résoudre cela, nous avons ajusté le script `seed.js` pour qu'il corresponde à notre base de données réelle en supprimant les configurations de test et en ajoutant les configurations spécifiques à notre projet. De plus, après avoir refait la base de données une première fois, nous avons dû la refaire une seconde fois puisque certaines informations ne correspondaient pas. Cependant, certaines tables avec des clés étrangères ne voulaient pas se drop, causant des erreurs. Pour résoudre cela, nous avons modifié le script `seed.js` pour ajouter une fonction `dropTables` contenant des instructions `await client.sql'DROP TABLE IF EXISTS ... CASCADE'` afin de supprimer correctement les tables même si elles avaient des clés étrangères.

Pour vérifier la connexion à la base de données, nous avons dû créer un script `testDbConnection.js` et l'exécuter avec la commande `node scripts/testDbConnection.js` pour s'assurer que la base de données était correctement accessible.

Fonction `dropTables()` dans le script `seed.js` :

```
async function dropTables(client) {
  try {
    await client.sql`DROP TABLE IF EXISTS precisions CASCADE`;
    await client.sql`DROP TABLE IF EXISTS sprints CASCADE`;
    await client.sql`DROP TABLE IF EXISTS countries CASCADE`;
    await client.sql`DROP TABLE IF EXISTS users CASCADE`;
    await client.sql`DROP TABLE IF EXISTS country CASCADE`;
    await client.sql`DROP TABLE IF EXISTS langues CASCADE`;
    await client.sql`DROP TABLE IF EXISTS continents CASCADE`;
    await client.sql`DROP TABLE IF EXISTS score CASCADE`;
    console.log('Dropped existing tables');
  } catch (error) {
    console.error('Error dropping tables:', error);
    throw error;
  }
}
```

Message dans le terminal pour affirmer la connexion avec la base de données :

```
PS C:\wamp64\www\Album-Travel> node scripts/testDbConnection.js
Connected to the database
```





PROBLÈMES RENCONTRÉS

04

Authentification

Lors de l'installation du projet, l'importation d'un dossier src à la racine a causé des problèmes de redirection côté serveur avec next-auth (= un module de système d'authentification de next.js). Pour résoudre ce problème, nous avons restructuré notre application pour éliminer le dossier src à la racine, ce qui a résolu les problèmes de redirection côté serveur. La solution nous a pris beaucoup de temps à trouver puisque ce problème n'était pas documenté.

Ensuite, nous n'avons pas pu utiliser la fonction `useSession()` de next-auth pour stocker et récupérer les sessions des utilisateurs en raison de problèmes de compatibilité de version. Pour résoudre cela, nous avons créé manuellement un système de session pour gérer les informations des utilisateurs en stockant l'email, le pseudo et l'id de l'utilisateur dans le `localStorage` après la connexion réussie. Enfin, nous avons créé une API interne pour récupérer les informations de l'utilisateur connecté. Ces informations sont ensuite stockées dans le contexte global de l'application grâce à un fichier `userContext.ts`, permettant ainsi de les utiliser partout dans l'application. Cela a été nécessaire car nous ne pouvions pas utiliser `useSession()` de next-auth, qui permet de stocker la session de l'utilisateur dans le `localStorage` et de récupérer ses informations.

Déconnexion

Pour la déconnexion des utilisateurs, nous avons dû gérer la suppression des données de l'utilisateur du `localStorage` et des cookies avec une fonction de next-auth mais aussi en ajoutant une vérification de fin de session manuellement. Nous avons créé un script `signOutAction.ts` contenant une fonction `handleSignOut` qui utilise la fonction `signOut()` de next-auth pour terminer la session, malgré le problème avec `useSession()`, cette fonction fonctionnait correctement. Puis, nous effaçons manuellement les cookies et le `localStorage` pour être sûr que la session soit bien effacée. Nous avons également ajouté un bouton de déconnexion dans le composant `SideNav` et utilisé `useRouter` de Next.js pour rediriger l'utilisateur après la déconnexion.



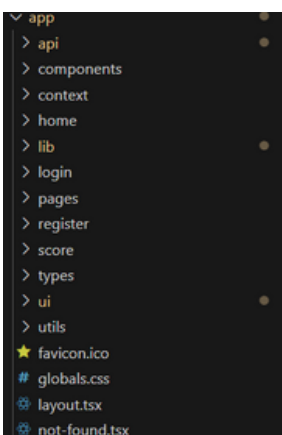
PROBLÈMES RENCONTRÉS

04

Redirection

La configuration de next-auth a nécessité des ajustements pour gérer correctement les sessions et les redirections. Nous avons configuré next-auth dans `auth.config.ts` et `auth.ts`, en définissant les pages de connexion et les callbacks pour la gestion des sessions, et en créant un middleware pour bloquer l'accès aux pages non autorisées et rediriger en conséquence. De plus, nous avons également rencontré un problème avec les utilisateurs qui pouvaient accéder à des pages 404 sans être connectés. Pour résoudre cela, nous avons configuré un middleware qui vérifie les cookies de session et redirige les utilisateurs non authentifiés vers la page de connexion ou d'inscription et créé une page pour l'erreur 404 afin que la redirection fonctionne. Avant, pour créer cette page 404 il fallait créer un fichier `404.tsx` dans un dossier `page`, cependant cela a changé avec la nouvelle version de Nextjs, donc après plusieurs recherche nous avons trouvé que pour cette version il fallait un fichier `not-found.tsx` à la racine du dossier `app` pour personnaliser cette page d'erreur. Par ailleurs, le middleware bloque toutes les tentatives d'accéder à une autre page que la connexion ou l'inscription si l'utilisateur n'est pas authentifié et redirige vers l'accueil si l'authentification a bien fonctionné.

Structure :



Code du fichier middleware.ts

```
1 import { NextResponse } from 'next/server';
2 import type { NextRequest } from 'next/server';
3
4 export async function middleware(req: NextRequest) {
5   console.log('Checking session token...');
6   const sessionCookie = req.cookies.get('authjs.session-token') || req.cookies.get('__Secure-next-auth.session-token');
7   const isAuthPage = req.nextUrl.pathname === '/login';
8   const isProtectedPage = ['/', '/login', '/register', '/api/auth', '/api/register'].includes(req.nextUrl.pathname);
9
10  if (isAuthPage && sessionCookie) {
11    return NextResponse.redirect(new URL('/', req.url));
12  }
13
14  if (isProtectedPage && !sessionCookie) {
15    return NextResponse.redirect(new URL('/login', req.url));
16  }
17
18  return NextResponse.next();
19 }
20
21 export const config = {
22   matcher: ['/((?!_next/static|_next/image|favicon.ico).*)'],
23 };
24
```

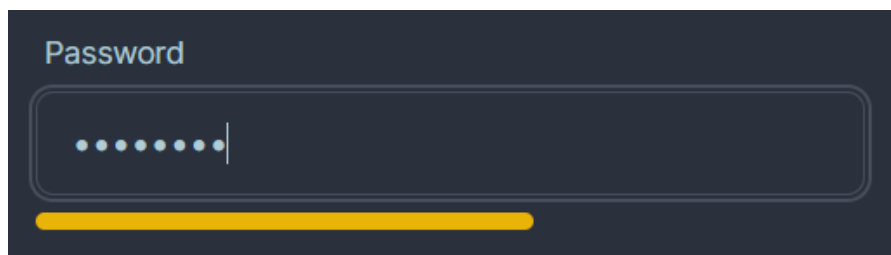


PROBLÈMES RENCONTRÉS

04

Insertion des utilisateurs

Nous avons eu des difficultés pour valider la sécurité des mots de passe. Pour résoudre cela, nous avons utilisé le module `zxcvbn` pour vérifier la sécurisation des mots de passe et ajouté des règles pour obliger l'utilisation d'une majuscule et d'un caractère spécial. Le processus d'inscription nécessitait de hacher les mots de passe avant de les insérer dans la base de données. Nous avons créé une API d'inscription qui reçoit les données, les valide, hache le mot de passe de l'utilisateur et insère les informations dans la base de données.



Aperçu de l'interface avec le module `zxcvbn` pour le mot de passe sécurisé

Ensuite grâce à la session créer manuellement où lors de la connexion nous récupérons l'email de l'utilisateur, son pseudo et l'id, nous pouvons insérer dans la base de données des données liée à l'utilisateur avec une requête sql. Ce fonctionnement avec l'authentification a permis de bien insérer les données selon l'utilisateur et par la suite de faire par exemple un tableau de score réelle.



PROBLÈMES RENCONTRÉS

04

Déploiement Vercel

Le déploiement sur Vercel a été entravé par des problèmes d'installation de modules. Nous avons vérifié et corrigé les configurations des modules pour nous assurer que tous étaient correctement installés et configurés.

Nous avons également rencontré des problèmes avec l'utilisation des variables d'environnement. Nous avons vérifié et corrigé les variables d'environnement dans le fichier .env et ajusté leur utilisation dans l'application.

Enfin, des pages vides et des problèmes d'échappement de caractères ont été résolus en vérifiant et ajustant les composants rendus, et en utilisant des méthodes appropriées pour gérer les caractères spéciaux comme par exemple "'" pour " ' ".

En résolvant ces problèmes un à un, nous avons réussi à déployer le projet sur Vercel.

Exemple de problèmes lors du déploiement avec Vercel :

```
14:48:48.284 Failed to compile.
14:48:48.285
14:48:48.285 ./auth.ts:8:15
14:48:48.285 Type error: '"./app/lib/definitions"' has no exported member named 'User'. Did you mean 'Users'?
14:48:48.285
14:48:48.285     6 | import { z } from 'zod';
14:48:48.286     7 | import { sql } from '@vercel/postgres';
14:48:48.286 >    8 | import type { User } from './app/lib/definitions';
14:48:48.286         |         ^
14:48:48.286     9 | import bcrypt from 'bcrypt';
14:48:48.287    10 |
14:48:48.287    11 | async function getUser(email: string): Promise<User | undefined> {
14:48:48.326 Error: Command "npm run build" exited with 1
```

```
13:36:20.766 ./app/home/page.tsx
13:36:20.766 55:94 Error: '' can be escaped with '&apos;','&lsquo;','&#39;','&rsquo;'. react/no-unescaped-entities
13:36:20.766
13:36:20.766 info - Need to disable some ESLint rules? Learn more here: https://nextjs.org/docs/basic-features/eslint#disabling-rules
```



PERSPECTIVES D'ÉVOLUTION

05

Idée 1 - Indice image supplémentaire pour le défi de précision

Dans le défi de précision, si l'utilisateur dépasse 1 minute 30 secondes, une image du pays lui est proposée comme indice supplémentaire pour l'aider à deviner le pays. Pour cela, nous devons intégrer une base de données associée à une banque d'images afin d'avoir une image disponible pour chaque pays.



Idée 2 - Mode Histoire ou Géographie

Pour améliorer le jeu, une idée serait d'ajouter des catégories où les indices ne se limiteraient pas à la géographie. Par exemple, selon le mode de jeu, les indices pourraient également inclure des aspects tels que l'histoire (date de fondation, pays colonisateur), le régime politique actuel, ou encore la religion majoritaire du pays.

Idée 3 - OAuth

Nous envisageons d'intégrer la possibilité pour les utilisateurs de s'inscrire et de se connecter via des services tiers tels que Google, GitHub, Facebook, et d'autres fournisseurs de services. Cette fonctionnalité offrira une méthode d'inscription pratique et sécurisée, simplifiant ainsi l'accès au jeu pour nos utilisateurs.

Idée 3 - Responsive

Nous prévoyons d'adapter le design pour rendre le jeu accessible et convivial sur les téléphones mobiles. Cela inclura des ajustements pour optimiser l'expérience utilisateur, en rendant les éléments de jeu et les interactions facilement navigables sur les écrans de taille réduite des smartphones.

