DOCUMENTATION INITIALISATION & TESTS

Mehdi Frikha

Jiaxuan Yang

Samuel VANIÉ

Emmanuel FEZEU

Mouahé

TABLE DES MATIÈRES

I. INITIALISATION

1. Backend

Les différentes étapes pour pouvoir mettre en place l'application sont:

- Installer les différents packages nécessaires pour le projet :
  npm run setup
- Initialiser la base de données :
  - L'url de la base de données se trouve dans root/backend/.env. Deux cas de figures possibles :
    - ❖ On a une base de données distante sur MongoDB.
    - ❖ SI vous voulez travailler localement , vous devez installer mongodb en local sur le PC. Il est aussi possible d'installer l'image docker de mongoDb à la place . Ensuite on peut utiliser ***compass*** pour avoir une interface graphique au lieu d'utiliser mongocli. Dans ce cas , vous devrez modifier l'url MONGO_DB dans le fichier root/backend/.env et mettre:***mongodb://localhost:27017/<nom-databse>***
  - Si vous voulez travailler en local , pour initialiser la base de données avec des valeurs par défaut , il suffit de run le script : ***init_database.sh***.
- Pour lancer le projet , on a 3 possibilités:
  - ***npm run start***
  - ***npm run startdev :*** Lance le projet avec nodemon.
  - ***npm run debugdev :*** Lance le projet et affiche les informations de débogage pendant l'exécution.
- Une brève documentation sur les différentes évenements géres par les sockets utilisés dans l'application :
  Les évenements déclenchés par les joueurs/côté client
  - ***rejoindre-jeu :*** Un joueur rejoint un jeu. L'événement rejoindre-jeu est emi coté client et traité par le serveur.
  - ***disconnect :*** Si un joueur se déconnecte , l'évènement disconnect est émis et traité par le serveur.
  - ***leave-game :*** Si un joueur quitte la partie , l'événement leave-game est émis et traité par le serveur.
  - ***Pouvoir-Spiritisme,request-Voyance,request-Loup-alpha :*** *Ce sont des événements que les joueurs déclenchent quand ils utilisent leurs pouvoirs, le serveurs fait ensuite le traitement associé.*
  - ***vote-jour, vote-nuit :*** Événement déclenché par lorsqu' un joueur effectue un vote. Ceux-ci sont traités par le serveur une fois l'action déclenchée.
  - ***send-message-game:*** Lorsqu'un joueur souhaite envoyer un message c'est cet événement qui est lancé côté client et traité par le serveur.

  Les événements déclenchés par le serveur
  - ***new-message :*** Notifie un joueur qu'il a reçu un message.
  - ***status-game :*** Envoie le statut du jeu aux différents joueurs.
  - ***player-info:*** Envoie ses informations courantes à un joueur.
  - ***notif-vote***: On informe aux joueurs qu'un vote a été fait contre x
  - ***notif-vote-final*** : On annonce aux joueurs qui a été tué à l'issue du vote.
  - ***send-Player-Data-Voyante/send-Player-Data-contamination***: On envoie au joueur qui possède le pouvoir les informations que son pouvoir renvoie.
  - ***new-custom-chat*** *: On notifie au deux joueurs concernés par le spiritisme qu'ils peuvent communiquer*

- *Le backend a été déployé sur* ***render. Le lien de déploiement est donné ci après :*** https://loup-garoup-app.onrender.com.
-

2. Frontend

- Afin de mettre en place le front end, il faut initialement faire un **npm install** dans le dossier root/frontend/loup-garou_front . Après avoir cela il faut il suffit de faire un **npx expo** start si l'on veut lancer l'application localement ou sur le web. Si vous voulez testez avec un téléphone c'est conseillé de faire un npx expo start –tunnel.
- Après avoir fait cela, il faut maintenant lier l'application à un backend. Par défaut l'application sera lié à un backend sur le site d'hébergement mais il est possible de le changer et de le lier à un backend local, il faut, pour cela, aller au fichier : /root/frontend/loup-garou_front/constants/links.js. Dans ce fichier il faut mettre l'élément backend à l'adresse souhaité pour utiliser le backend il ne faut pas mettre http://localhost:3000 il faut mettre l'adresse local de la machine dans le réseau local par exemple 'http://192.168.0.14:3000' dans mon réseau.
- Après avoir fait cela, vous pouvez lancer l'application en appuyant sur w qui lancera l'application web et pour lancer la version android, l'application doit être lancée en tunnel et il suffit de télécharger l'application de expo ensuite prendre une image du Qr code dans l'application. Il est aussi possible d'utiliser un émulateur android installable grâce à avd en ligne de commande, ou en téléchargeant android-studio, via son interface.
- Une fois cela fait, vous rencontrerez la figure suivante (figure 1). Il suffit de créer un compte dans la sous-fenêtre de registre , puis de vous connecter. S'il y a un problème lors de la connexion (compte qui n'existe pas ou création de compte impossible), l'application Android affichera une erreur, mais sur le web, ces erreurs ne seront pas visibles. Pour que l'application soit bien dimensionnée sur le web et si vous utilisez Firefox, vous pouvez ouvrir "Inspect Element" et appuyer sur le troisième bouton à gauche dans le coin  en haut à droite, près de la croix. L'image d'Inspect element de firefox est dans la figure 2 sur laquelle l'on peut voir ce button.

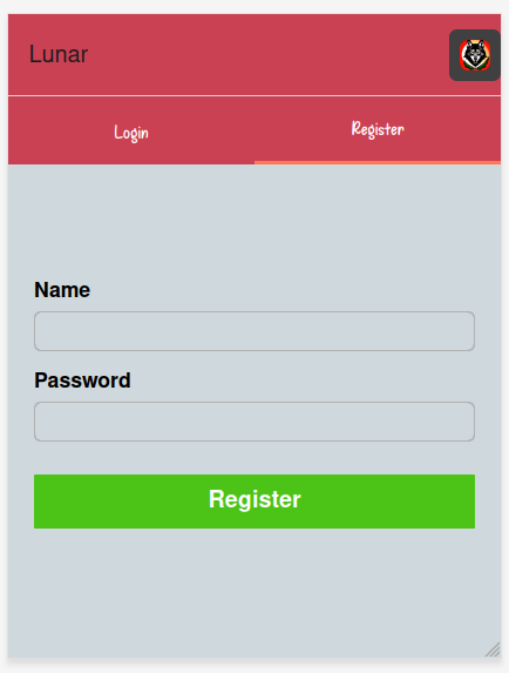

Figure 1 Page d'accueil.

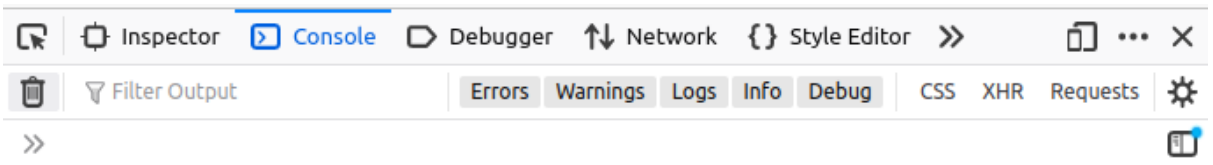

Figure 2 Inspect element firefox

- Pour une explication détaillée du fonctionnement, il suffit de voir le document de rendu de l'acol disponible dans le dossier root/docs/renduAcol.pdf . Il faut lire la partie 3 du document dans le manuel utilisateur.

Lancer une partie à plusieurs pour tester :

- L'astuce qu'on a utilisé pour réaliser cette opération est d'ouvrir npx expo plusieurs fois pour simuler la connexion de plusieurs utilisateurs. Le problème qui trouve sa solution en réalisant cette opération réside dans le fait que le localstorage est intrinsèque à l'instance du navigateur qui est ouvert, donc dans le même navigateur l'on ne pouvait pas juste lancer plusieurs fois le même lien de expo et s'attendre à ce que les fenêtres ou clients du jeu soient indépendants.
- Voici un exemple de lancement d'une application de 4 personnes:
- Connexion au compte: pour avoir quatre instances différentes on peut lancer expo 4 fois avec les liens localhost:1900-0/1/2/3. Chaque lien sera associé à un utilisateur. Lancé 4 expo peut se révéler trés trés lourd, surtout sur ma machine donc une solution que j'ai trouvé est d'ouvrir 2 expo et ensuite ouvrir les deux sites localhost:19000-0/1 sur firefox et en même temps sur chrome afin de ne pas avoir le problème de local storage.
- Après avoir fait cela, il faut créer un compte par utilisateur et se connecter. Ensuite il faut créer un jeu,il faut bien mettre une date qui est proche par exemple mettre la date actuelle plus de deux minutes. (Si la date mise est inférieure à la date actuelle le jeu ne sera pas lancé il bien regarder la date) Ensuite quand vous entrez dans la salle d'attente de la première page il faut prendre l'id de la partie et rejoindre la partie dans les différents comptes.
- Maintenant normalement tous les comptes se sont connectés et il faut juste attendre que le jeu commence et si vous avez mis le nombre de joueur souhaité égal à 4 l'application lancera directement quand 4 personnes se connectent.
- Enfin, le manuel explique comment le jeu fonctionne et quelles sont les différentes fonctionnalités (comment voter, utiliser un pouvoir, parler dans le chat). Le menu est assez intuitif donc la lecture du manuel n'est pas nécessaire mais s'il y a quelque chose qui n'est pas clair vous pouvez le consulter.
- Note : Si le nombre de personnes connectées est inférieur au nombre souhaité, la partie commencera quand le temps de début arrive sa fin et au moins deux joueurs se sont connectés.

II. TESTS

1. Backend

Les tests backend ont été fait avec ***jest , supertest.***
Nous avons créé un environnement de test. Ainsi on a une base de données pour l'environnement de test, et une autre pour l'environnement de production. Dans le fichier root/Backend/.env, la variable NODE_ENV définit dans quelle environnement on se trouve. Elle est par défaut initialisée à prod.

La commande pour lancer les tests est : ***npm run test***

Description des tests effectués :
Ils sont situés dans le répertoire root/Backend/src/___tests___

- Dans le fichier user.test.js : On teste les différents endpoints associés au traitement des utilisateurs. On vérifie aussi que les middlewares implémentés se comportent comme attendu.
- Dans le fichier partie.test.js : On teste les différents endpoints associés au traitement d'une partie i.e la création de la partie, l'ajout de joueurs dans une partie.
- socket.test.js : On crée une partie, deux joueurs rejoignent la partie, on essaie de connecter les deux joueurs à la socket de la partie et on vérifie si une fois connectés ils reçoivent bien les informations sur le statut de la partie**.**

2. Frontend

Pour lancer les tests du frontend, vous devez d'abord vous assurer que Expo a été lancé avec la commande “npx expo start”. Ensuite, il faut appuyer sur la touche "w" pour lancer le site web correspondant.
Il est aussi important de vérifier que le backend, avec lequel l'application est connectée, fonctionne correctement.
Après, vous pouvez lancer les tests automatiques en exécutant la commande “npx cypress run”. Vous pouvez également ouvrir le menu Cypress avec la commande “npx cypress open”, qui vous permettra de lancer les tests individuellement.

Globalement, on a créé deux fichiers cypress pour tester les fonctionnalités de front-end, ils sont auth.cy.js et configJeu.cy.js.

1) **describe('Log the user scenario'....)** : ce test vérifie le scénario d'authentification de l’utilisateur, il simule la connexion en utilisant un intercept pour intercepter la requête POST à l’api de connexion. Après avoir saisi les informations de connexion, il vérifie si la vue principale est affichée, ce qui signifie que l’utilisateur est bien connecté.
2) Test de création de compte utilisateur : **describe('Create user account scenario'.....):** ce test vérifie le scénario de création de compte joueur, Il simule la création d’un compte en utilisant un intercept, l’idée est pareil que l’intercept qu’on a expliqué avant. Après avoir saisi les informations d’inscription, on vérifie si le bouton de connexion est présent, ce qui signifie l’utilisateur a été rediriger vers la page de connexion (à côté de la page ‘register’)
3) test des fonctionnalitées de l’interface de configuration avant la création d’un jeu: **describe('test the functionalities of configuration interface before creating a game':** ce test effectue plusieurs actions pour configurer un jeu, comme la définition du nombre minimum de joueurs, la durée du jour et de la nuit, la date de début du jeu, le pouvoir des loups et la proportion des loups. Il vérifie également que l'utilisateur peut créer le jeu et revenir à la page d'accueil.