Rapport de projet de technologies informatiques innovantes : Enig'match

Clélie Amiot, Morgane Colle, Corentin Lefebvre et Julie Simonin

M1 SCA

Présentation du site

L'objectif de ce projet étant la réalisation d'un site sous la forme de Single Page Application (SPA), nous avons choisi de créer un site d'énigmes alimenté par les utilisateurs eux-mêmes. Nous pensons qu'il s'agit d'un concept intéressant à exploiter pour ce projet car nous imaginons le site comme un fil où les énigmes sont classées selon différents critères comme la date par exemple, ce qui correspond bien au fonctionnement d'une SPA.

Le principe de base de notre site est simple : un utilisateur créé une énigme, celle-ci est ajoutée à la base de données et est consultable par tous (utilisateurs non-connectés compris), mais seuls les joueurs inscrits peuvent y répondre.

Une énigme est composée de plusieurs informations :

- son contenu (la question, la charrade…)
- sa réponse
- son auteur
- la date à laquelle elle a été postée

Nous proposons aux utilisateurs non-connectés de s'authentifier s'ils ont un compte ou de s'inscrire. L'inscription requiert : un nom d'utilisateur, une adresse e-mail et un mot de passe. Nous avons mis en place une fonction de vérification de mot-de-passe pour vérifier que l'utilisateur ne se trompe pas en remplissant le formulaire.

Lorsqu'un utilisateur se connecte sur notre site, il doit saisir son pseudo ainsi que son mot-de-passe dans les champs appropriés.

Une fois connecté, un utilisateur peut répondre à une énigme, créer des énigmes, accéder à son profil et se déconnecter. La page profil n'est pas totalement fonctionnelle, mais nous y retrouvons les principales catégories que nous souhaitions mettre en place, à savoir : informations générales sur le joueur, achievements, consultation des énigmes qu'il a postées, les énigmes suivies par le joueur, celles qu'il a résolues et celles qu'il n'a pas résolues, après une ou plusieurs tentatives infructueuses.

Techniques

Pour notre site, nous avons réalisé une fausse API avec JSON-server. Nous avons essayé d'ajouter une authentification en jwt avec le module jsonwebtoken : lorsque nous faisons une requête POST sur la page /auth/login, avec le pseudo et le mot de passe nous recevons un token d'authentification qui est valable 1h et qui est nécessaire pour pouvoir accéder aux autres pages du site.

Normalement, il faudrait que la vérification des réponses aux énigmes et la mise à jour des scores soient faites par un serveur pour qu'on ne puisse pas "tricher" avec la résolution des énigmes.

Etant donné que l'objectif du cours n'était pas de faire du back-end et d'interagir avec une vraie base de données côté serveur, nous effectuons les actions du côté de l'application client avec Angular. Ainsi, le client récupère l'énigme et sa solution, puis vérifie que les deux se correspondent. Comme nous l'évoquerons plus tard dans les améliorations possibles, c'est à ce moment qu'aurait eu lieu la mise à jour du score si nous l'avions implémentée.

De ce fait, l'utilisateur aurait pu tricher en utilisant notre API afin de mettre à jour son score sans avoir eu la réponse à l'énigme, ou regarder directement la réponse puisqu'on la récupère en même temps que la question même si elle n'est pas affichée directement à l'écran, mais elle est contenue dans un objet javascript chez le client.

De la même manière, comme le JSON-server permet de récupérer n'importe quelle table de la base de données, lorsque nous effectuons la jointure des tables utilisateurs et énigmes pour vérifier si l'utilisateur a donné la bonne réponse, nous récupérons également toutes les informations relatives au joueur comme son adresse e-mail et son mot-de-passe (que nous avons quand même hashé). Nous avons donc conscience de ces problèmes de sécurité et savons qu'avec un vrai serveur, il ne faudrait récupérer que l'énoncé de l'énigme.

Utilisation

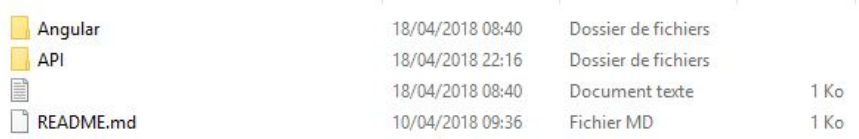
Architecture du site

Pour notre site, nous avons utilisé YAAK Angular Kickstarter qui nous a permis de construire le squelette de notre projet. Notre projet se divise en 2 parties : l'API_server (la base de données et l'API pour y accéder) et Angular, l'application Angular.

Comment lancer notre site ?

1) Récupérer le projet depuis : https://github.com/capybaraking/TII

2) Accéder au dossier du projet :



3) Dossier API :

- ouvrir un terminal à cet endroit et écrire : `npm install`
- attendre la fin du traitement, puis taper : `npm run start`

4) Mêmes instructions dans le dossier Angular > app > public

5) Le site s'ouvre sur la page d'accueil.

Améliorations possibles

Dans la version actuelle de notre projet, nous nous limitons à des énigmes au contenu textuel, mais on peut imaginer une amélioration où on pourrait publier des énigmes sous forme d'images.

Par ailleurs, nous pourrions rendre le site plus ludique afin de stimuler les joueurs et créer plus d'interactions entre eux. Nous avons pensé à des systèmes de points et de récompenses à implémenter dans une version future. Cette idée aurait pu mener à la créations de catégories d'énigmes où une section de la page d'accueil aurait pu être consacrée aux énigmes valant le plus de points

Enfin, si le site présentait beaucoup d'énigmes, nous aurions pu associer des tags aux énigmes afin de les caractériser et rendre possible l'utilisation d'un moteur de recherche.

Conclusion

Au cours de notre réalisation et des cours suivis durant ce module, nous avons découvert un nouveau framework JavaScript. Ce projet nous a donc permis de mettre en pratique les connaissances acquises dans le module sur l'utilisation d'AngularJS et ses spécificités telles que l'utilisation de controllers, services et factory. Il a été enrichissant pour nous d'apprendre à travailler avec un nouvel outil beaucoup utilisé en développement et de nous familiariser avec l'utilisation de frameworks. Nous nous sommes également servis du framework CSS Materialize afin d'opter pour un rendu aussi responsive que possible.

Compte tenu du délai que nous avions et du fait que nous étions totalement novices par rapport à cette technologie, nous sommes arrivés à peu près au point que nous nous étions fixés au départ. Nous avons cependant conscience que plusieurs fonctionnalités auraient pu améliorer notre projet (cf. partie sur les améliorations possibles).