



PC3R

Projet application web

Groupe :
Sandrine EAR - 21110030
Mebarek Raid BENAKCHA - 21114000
Malik BOUAOUD - 21105730

Table des matières

Dossier	3
Description des cas d'utilisations	3
Fonctionnalités	4
Données	4
Squelette serveur	5
Squelette client	6
Schéma global du système	7
Manuel d'utilisation	8
Lancement de la partie client	8
Lancement de la partie serveur	8
Rapport de développement	9
Développement	9
Technologies utilisées	9
Choix d'implémentation et explication	9
Partie quiz	9
Partie leaderboard	9
Partie authentification et base de données	9
Difficultés rencontrées	10
Conclusion	10
Annexes	11
Page d'accueil	11
Page d'inscription	11
Page de connexion	12
Page du leaderboard	12
Page du quiz	13

Dossier

Le sujet porte sur la réalisation d'une application web de tests proposant des quiz dans le domaine de l'informatique aux utilisateurs en attribuant des points à chaque bonne réponse. L'utilisateur doit d'abord s'authentifier pour jouer ou bien créer un compte s'il ne détient pas encore.

Il choisit de jouer à une partie en choisissant les différentes options du jeu tel que la difficulté ou bien encore la catégorie.

Le joueur peut également consulter le leaderboard contenant le classement des meilleurs scores qu'il soit connecté ou non.

L'API Web externe répondant à nos besoins a été trouvée sur le site suivant : <https://quizapi.io/>.

Elle va proposer des quiz en fonction des données passées à l'URL. Il existe différents types de données pouvant être passés à l'API tel que la catégorie et la difficulté (easy, medium, hard). Par ailleurs, nous avons décidé de limiter le nombre de questions à 10.

Par exemple, nous pouvons envoyer la requête suivante à l'API :

https://quizapi.io/api/v1/questions?apiKey={API_KEY}&limit=10&category={CATEGORY_NAME}&difficulty={DIFFICULTY}

- **API_KEY** représente la clé secrète qui nous authentifie lorsque nous faisons une requête au site.
- **CATEGORY_NAME** : [Linux, DevOps, Docker] et d'autres catégories qui ne sont pas mentionnées dans la documentation de l'API
- **DIFFICULTY** : [Easy, Medium, Hard]

Cet API Web retourne un objet JSON que nous découpons pour récupérer les questions, les différentes possibilités de réponses proposées ainsi que la bonne réponse.

L'API sera contacté à chaque fois qu'un utilisateur décidera de participer à un quiz. Une fois que l'utilisateur aura choisi la catégorie et la difficulté, l'API sera contacté par le serveur pour récupérer un quiz correspondant à la requête exprimée.

Description des cas d'utilisations

Voici la description de deux cas d'utilisations pour notre application web :

- Paulin ouvre l'application Tykuiz et s'authentifie, il choisit de jouer à un quiz.

Il choisit la catégorie "Linux" et la difficulté "Easy".

Paulin répond aux 10 questions.

Paulin se déconnecte.

- David ouvre l'application et s'authentifie, il choisit de consulter le leaderboard.
David se déconnecte.

- Chloé ouvre l'application et consulte le leaderboard.

Fonctionnalités

Notre application propose aux utilisateurs de s'authentifier afin qu'ils puissent jouer en répondant à une série comportant plusieurs questions. Les joueurs pourront également consulter le leaderboard composé du classement des joueurs possédant les meilleurs scores.

Données

Certains types de données ont besoin d'être stockés côté serveur afin d'assurer le fonctionnement de l'application.

Nous avons notamment eu besoin des informations relatives aux utilisateurs afin de pouvoir les utiliser à travers les fonctionnalités de notre application web.

Pour chaque utilisateur, nous stockons leurs informations personnelles, leur pseudonyme de jeu, un mot de passe associé au compte ainsi que leur score dans une base nommée "users".

Schéma de la collection "users"

users	
id	Number
lastname	String
firstname	String
username	String
email	String
password	String
score	Number

La base de données est amenée à réaliser des mises à jour lors de l'ajout d'un utilisateur ou bien la mise à jour du score d'un joueur.

L'API externe est appelée lorsqu'un utilisateur souhaite réaliser un quiz. Lorsqu'il choisit les paramètres du quiz et qu'il appuie ensuite sur le bouton "Commencer", nous faisons d'abord appel au serveur qui va faire appel à l'API afin de générer la série de 10 questions selon la catégorie et la difficulté choisie.

Voici un exemple de description de requête avec sa réponse :

- GET /quiz
réponse : récupérer un quiz
- POST
/register?firstname={**FIRSTNAME**&lastname={**LASTNAME**&email={**EMAIL**&username={**PSEUDO**&password={**PASSWORD**}
réponse : envoyer un formulaire contenant le nom, prénom, email, pseudonyme et mot de passe au côté serveur afin d'enregistrer le nouvel utilisateur dans la base de données.
- POST /login?username={**PSEUDO**&password={**PASSWORD**}
réponse : envoyer les informations concernant l'utilisateur, nom d'utilisateur et mot de passe au côté serveur, si les bonnes informations ont été entrées, l'utilisateur se verra alors connecté.

Squelette serveur

Pour notre application web sur les quiz, nous nous basons sur une approche ressource (REST).

Dans cette approche, les requêtes et les réponses sont construites autour du transfert de représentation de ressources. Une ressource désigne une base de données que nous pouvons interroger, par exemple, nous pouvons interroger la collection "users" de notre base de données. Une représentation est un document qui capture un état de la ressource.

Voici quelques descriptions des requêtes avec leurs réponses :

- GET /home
réponse : récupère la page d'accueil.
- GET /leaderboard
réponse : retourner une page contenant la liste des premiers joueurs ayant les scores les plus élevés.
- GET /play
réponse : retourner une page permettant de faire le quiz.

Les autres requêtes ont déjà été décrites un peu plus dans la partie donnée.

Squelette client

Pour la partie client, nous avons fait le choix d'utiliser la bibliothèque client React avec le module "react-bootstrap" qui ajoute les composants de la bibliothèque Bootstrap au sein de ceux de React.

Bootstrap pour React a été utilisé pour l'ensemble des interfaces graphiques de notre application web.

Plan du site de la partie client

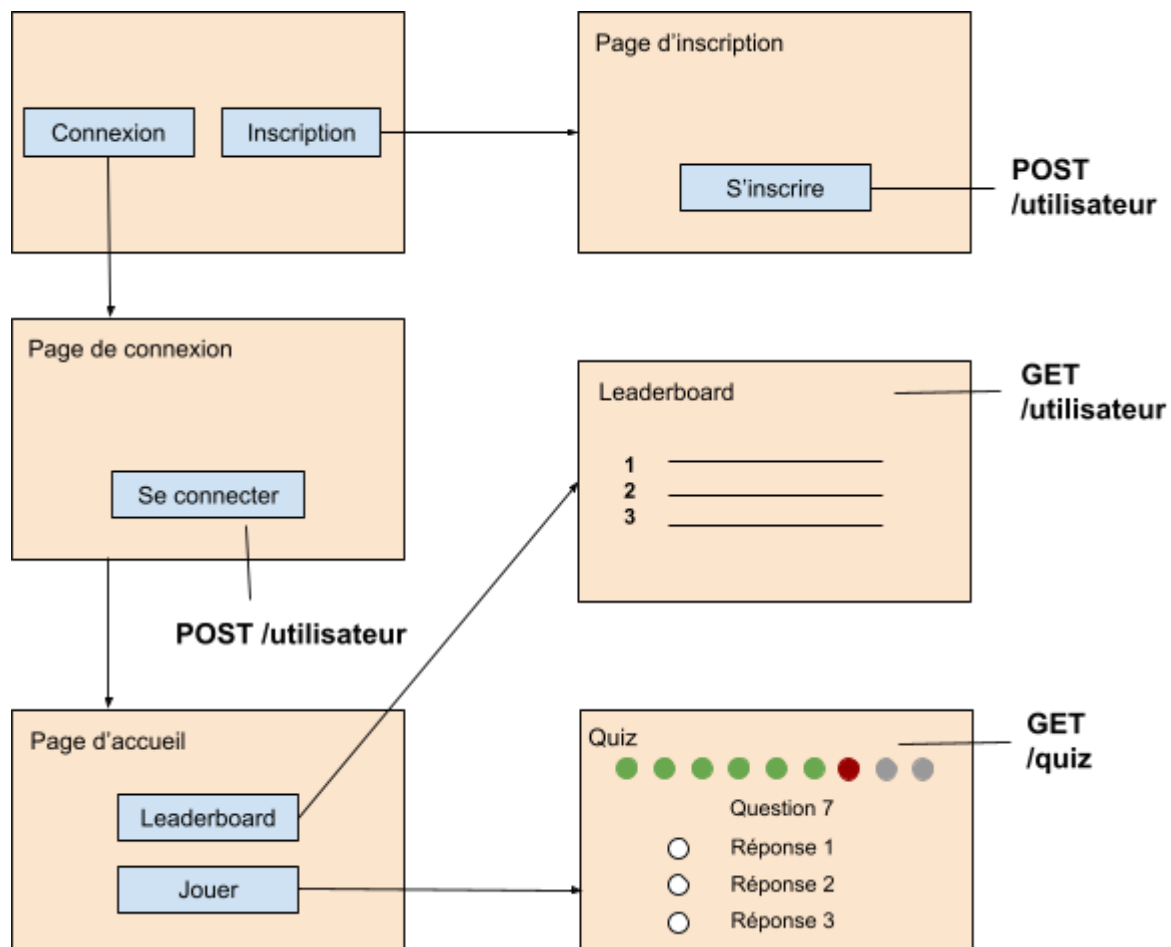
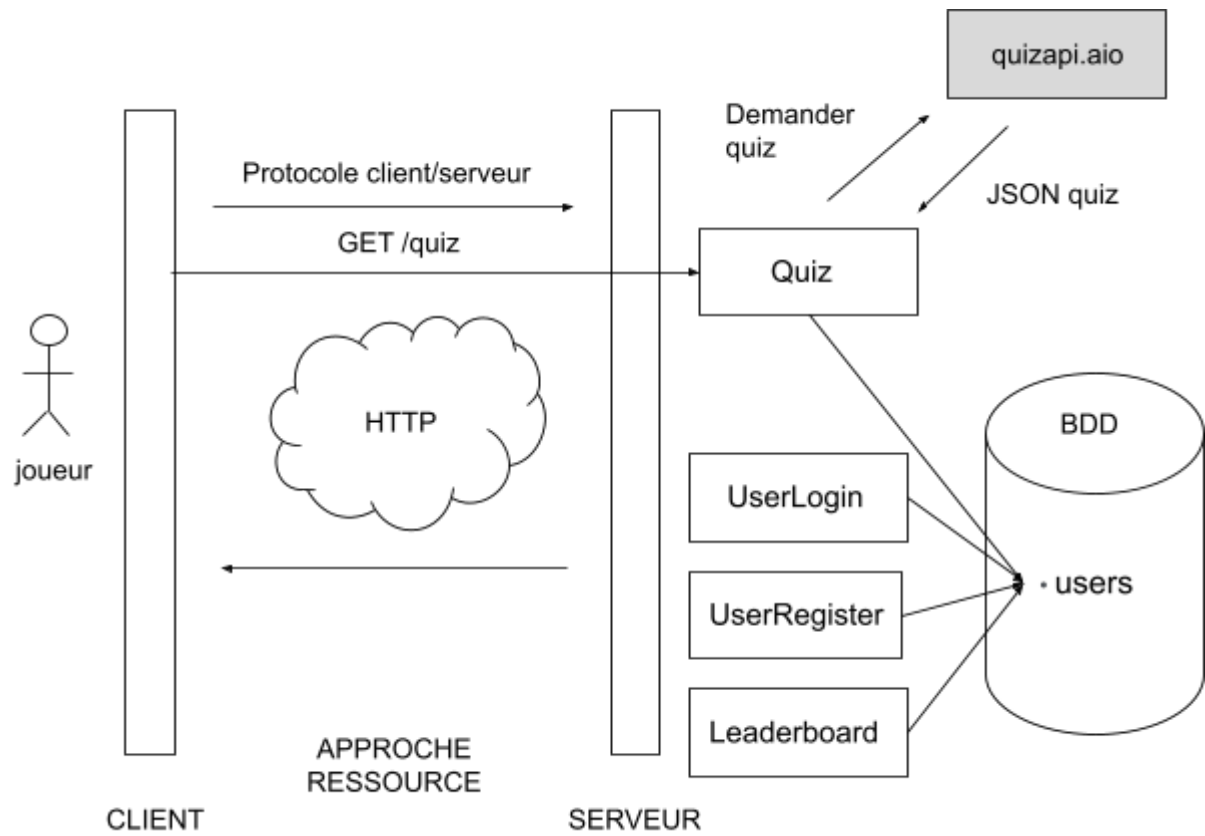


Schéma global du système

Schéma général de l'application



Manuel d'utilisation

Lancement de la partie client

Ouvrir un terminal en se plaçant dans le dossier frontend et lancer

```
> npm install
```

```
> npm run start
```

Lancement de la partie serveur

Prérequis : Tomcat , MongoDB, Eclipse

Sur Eclipse, choisissez d'importer un projet web "**war**". Le dépôt git contient un fichier "Tyquiz.war" contenant toute la partie backend.

L'importer permet de récupérer tout le code des servlets implémentés, une fois importé lancer le serveur tomcat en localhost.

Clique droit sur tykuiz puis "Run" et ensuite "**Run on Server**".

Vous pouvez maintenant utiliser l'application Tykuiz.

Rapport de développement

Développement

Technologies utilisées

La partie client a été réalisée avec React et ses différents modules notamment "react-bootstrap" et "redux". Nous avons choisi cette technologie car elle nous a facilité la réalisation de l'interface comme sur les aspects graphiques des entités se trouvant sur notre application (react-bootstrap) ainsi que la gestion de l'état global du joueur dans l'application et surtout quand il répond aux quiz grâce à redux.

Le serveur de notre application est écrit en Servlets Java utilisant le serveur d'applications Tomcat qui a été imposé.

Nous avons utilisé le système de gestion de base de données MongoDB. Dans cette base de données, nous stockons les informations des utilisateurs.

Choix d'implémentation et explication

Partie quiz

Pour la page des quiz, nous avons décidé d'afficher 10 points illustrant l'état d'avancement du joueur lors d'une partie.

Pour chaque réponse correcte, le score du joueur augmente d'un point. Pour réaliser cela, nous avons utilisé la bibliothèque "redux" qui est une bibliothèque Javascript permettant de gérer les différents états d'une application web. Elle nous a permis de mémoriser les scores de joueurs et leurs informations de connexion.

Partie leaderboard

Le leaderboard est un tableau qui trie les dix premiers joueurs selon leurs scores. Comme l'indique le schéma de la collection 'users', chaque utilisateur a un champ 'score' qui est mis à jour après chaque partie du quiz. Le leaderboard se crée en faisant une requête au serveur qui de son côté sollicite la base de données pour récupérer la liste des dix premiers joueurs triée d'une manière décroissante selon la valeur du champ 'score' de chaque joueur. Ensuite, on affiche le pseudo du joueur avec son score à côté, en plus des médailles pour les trois premiers joueurs.

Partie authentification et base de données

Pour cette partie du projet, nous avons créé deux pages d'authentification, une page d'inscription et une page de connexion.

Pour garder une trace de l'état d'un utilisateur s'il est connecté, nous avons utilisé tout comme dans la partie quiz, "redux" pour mémoriser ces états.

Nous avons choisi d'utiliser une base de données MongoDB de type NoSQL puisqu'elle permet une plus grande flexibilité lors des insertions de données ou leurs extractions.

Lorsqu'un utilisateur décide de s'inscrire, une requête vers le côté serveur est émise avec les différentes informations de l'utilisateur. Nous ajoutons un utilisateur à la base de données uniquement si le pseudonyme (qui sert d'identifiant de la table "users") n'est pas présent.

Même principe pour la partie connexion, sauf que nous faisons une recherche à la place d'une insertion dans la base de données. Si un nom d'utilisateur et un mot de passe correct ont été entrés alors celui-ci est connecté et peut donc accéder au contenu que propose l'application web.

Difficultés rencontrées

Nous avons rencontré des problèmes lors de la réalisation du backend, notamment lors de l'installation du serveur d'applications Tomcat ou bien encore avec des problèmes au lancement du serveur.

L'export du backend était un élément très délicat dans le sens où une fois après avoir mis à jour le backend il devenait parfois inutilisable pour les autres membres du groupe. C'est pourquoi avant chaque push nous exportons le côté serveur en fichier .war.

L'utilisation du langage Javascript a été plus difficile pour les membres n'ayant pas été amené à manipuler beaucoup ce langage, en effet étant dans une formation très théorique l'univers Javascript permet de réaliser plus de choses que la plupart des langages étudiés à l'université ce qui a créé des blocages lors de cette transition chez certains d'entre nous.

Conclusion

Ce projet nous a permis de manipuler de nouvelles technologies, Javascript ainsi que ses différents frameworks et bibliothèques React, redux, Bootstrap.

Nous avons pu appliquer les enseignements théoriques de l'UE MLBDA dans un vrai contexte d'application en utilisant des bases de données de type NOSQL et tester leurs flexibilités en dehors des séances de TD.

Nous avons également appris à développer des serveurs en choisissant le type d'approche que nous utilisons, les composants présents et leurs comportements.

Une application comme TYKuiz peut être développée afin d'éviter le stress et la perte de temps des entretiens techniques pour un poste dans l'informatique en testant directement les connaissances d'un candidat sur une plateforme en ligne simple et joviale.

Annexes

Page d'accueil



Page d'inscription

The screenshot shows the registration page of the TYKuiz website. It has the same navigation bar as the homepage. The main content area is light orange and features a large heading 'Inscription'. Below the heading, there are five input fields for registration: 'Nom' (with placeholder 'Entrez votre nom'), 'Prénom' (with placeholder 'Entrez votre prénom'), 'Pseudonyme' (with placeholder 'Entrez votre pseudonyme'), 'Adresse mail' (with placeholder 'Entrez une adresse mail'), and 'Mot de passe' (with placeholder 'Entrez un mot de passe'). A green button labeled 'S'inscrire' is positioned below the password field. The footer at the bottom contains 'TYKuiz ©' and the tagline 'Challenge your IT knowledge with us'.

Page de connexion

TYKuiz

Accueil Jouer Classement

Inscription Connexion

Authentification

Pseudonyme

Entrer votre pseudonyme

Vos informations personnelles ne seront jamais communiquées

Mot de passe

Entrer un mot de passe

Se connecter

TYKuiz ©

Challenge your IT knowledge with us












Page du leaderboard

TYKuiz

Accueil Jouer Classement

Déconnexion

Leaderboard

		maycs	
		Sandrine	
		raid	
		Alex	



In which directory by default user home directories are created?

- ☐ /home
- ☐ /usr
- ☐ /tmp
- ☐ /etc
- ☐ /user

2

Suivant