3CSI - Web Avancé

Projet

Vous êtes étudiant en 3CSI et vous êtes un bon élève. En cours de web avancé, votre professeur vous donne des devoirs sur papier. Vous et vos camarades êtes conscient des enjeux écologiques de notre époque et décidez de prendre les choses en main.

En équipe de 3 et 4 personnes, développez une application web qui permet aux élèves de passer des examens sous forme de questionnaire. L'application sera développée en NodeJS en utilisant le framework ExpressJS. Utiliser Bootstrap pour le design, Pug pour le templating et Socket.IO pour les communications temps réel entre le client et le serveur.

Le projet devra être rendu dans un ZIP par mail avant le 26/02/2017 23:59. Tous projet rendu en dehors de ce délais aura la note de 0. Vous indiquerez dans le mail le nom de chaque membre du groupe ayant travaillé sur le projet, un checksum md5 du ZIP et toutes remarques pouvant être utile à la correction.

Process de candidat

- Le candidat ouvre la page d'accueil « `/` » de l'application.
- La candidat indique son nom et son prénom. Il est redirigé vers l'espace de test « `/play` ».
- L'espace de test est constitué du texte de la question, d'un minuteur, d'un bouton « suivant » et de la zone de réponse (zone libre ou réponse QCM).
    - Lorsque le candidat a fini, il clique sur suivant, sa réponse est soumise et la question suivante s'affiche sans rechargement de la page.
    - Lorsque le temps autorisé pour la question est écoulé, une popup s'affiche avec le bouton « suivant ». La popup indique au candidat que le temps est écoulé et que sa réponse sera soumise en l'état. Lorsque la popup est affiché le candidat ne peut plus modifier sa réponse à la question.
- Une fois le formulaire terminé le résultat est enregistré sur le disque dur du serveur. La note du candidat est affiché à l'écran.

Process de professeur

La première version de cette application est très simple et ne nécessite pas d'administration. Le professeur décrira son devoir dans un fichier JSON qu'il uploadera depuis la page « `/update` ». L'application utilisera ce fichier comme référence pour savoir quoi poser comme question.

- Le professeur ouvre la page « `/upload` » de l'application.
- La page est composé d'un formulaire de téléchargement n'acceptant que les fichiers JSON.
- Le professeur upload le fichier de configuration en JSON.
- L'application affiche un aperçu des questions et réponses récupérées de ce fichier JSON.

Le fichier de configuration doit respecter le format suivant :

```
[
  {
    “type”: “choice”,
    “time”: 60,
    “question”: “Quelle est la couleur du cheval blanc d’Henri IV ?”,
    “choices”: [
      “Rouge”,
      “Blanc”,
      “Bleu”,
      “Vert”
    ],
    “responses”: [
      “Blanc”
    ]
  },
  {
    “type”: “free”,
    “time”: 45,
    “question”: “Citer une des trois couleurs du drapeau français”,
    “responses”: [
      “Rouge”,
      “Blanc”,
      “Bleu”
    ]
  }
]
```

Les fichiers de résultats doivent respecter le format suivant :

```
{
  “name”: “Jean Duval”,
  “date”: “20170217”,
  “score”: “75”,
  “responses”: [
    {
      ”question“: “…”,
      “success”: true,
      “responses”: [
        “…”
      ]
    },
    {
      ”question“: “…”,
      “success”: false,
      “responses”: [
        “…”
      ]
    }
  ]
}
```

Barème

	Critère	Points
	L’application possède une page « `/` » demandant au candidat de s’identifier	1
	L’application redirige le candidat sur l’espace de test	1
	L’espace de test de l’application correspond au slug « `/play` »	1
	L’espace de test contient le texte de la question, le minuteur, la zone de réponse et le bouton « suivant »	1

	Critère	Points
CLIENT	Le type de réponse de chaque question respecte celui indiqué dans le fichier de configuration	2
	Au clic sur le bouton « suivant » le résultat est enregistré	1
	Au clic sur le bouton « suivant » l'espace de test affiche la question suivante	1
	Le changement de question se fait sans rechargement de la page	3
	Le temps affiché par le minuteur de chaque question respecte celui indiqué dans le fichier de configuration	2
	Le temps du minuteur est décompté seconde après seconde	1
	Lorsque le minuteur affiche 0 seconde une popup s'affiche	1
	Pendant que la popup est affiché le test ne passe pas à la suite	1
	Pendant que la popup est affichée il est impossible de modifier la réponse du candidat	2
	Au clic sur le bouton « suivant » de la popup, la question suivante s'affiche	1
	Une fois le test terminé le candidat peut voir le résultat du test	1
	Une fois le test terminé le résultat est enregistré dans un répertoire du projet	2
	Le candidat peut passer plusieurs fois le test mais les résultats ne doivent jamais être effacés	2
	Lorsque le candidat repasse le test, l'application redemande le nom et le prénom du candidat	1
	Les questions affichées lors du test sont celles écrites dans le fichier de configuration	2
	Si aucun fichier de configuration n'est trouvé par l'application, un message l'indique au candidat sur la page d'accueil « `/` ». Aucun test n'est lancé	1
	Si le candidat s'arrête au milieu du test, le test est considéré comme abandonné et les réponses restantes comme échouées.	3
SERVEUR	L'application possède une page « `/upload` »	1
	L'espace de téléchargement « `/upload` » contient un formulaire de téléchargement et une zone de visualisation du test actuel	1
	Le formulaire de l'espace de téléchargement modifie le fichier de configuration	1
	Après le téléchargement d'un nouveau fichier de configuration, la zone de visualisation se met à jour avec les nouvelles informations sans rechargement de la page	2
	La zone de visualisation de l'espace de visualisation est une liste des questions du test actuel.	1
	La zone de visualisation de l'espace de visualisation affiche pour chaque question, les réponses, le temps et les choix correspondants	1
	L'application calcule automatique la note du candidat en fonction de ses réponses et des réponse attendues	3
	Le professeur doit pouvoir récupérer les résultats de tous les tests passés par ses élèves triés par date dans un dossier de l'application.	1
	Seuls les fichiers JSON sont acceptés par le formulaire de l'espace de téléchargement	1

	Critère	Points
	Les fichiers de résultat respectent le format demandé	1
	Le temps pour chaque question est géré côté serveur	4
BOTH	Boostrap est correctement utilisé pour le design	1
BOTH	Pug est correctement utilisé pour le template	2
BOTH	ExpressJS est correctement utilisé	2
BOTH	Socket.IO est correctement utilisé pour la gestion du temps réel	3
BONUS	Le projet utilise BabelJS	0,5
BONUS	Le projet utilise les commandes de lifecircle de NPM	0,5
BONUS	Le projet est servi dans un container Docker et utilise les volumes	0,5
BONUS	Le code de traitement doit se situer côté serveur (NodeJS), le code Javascript coté client ne sert qu'à l'animation et à la transmission des informations au serveur.	0,5
BONUS	Le code est écrit en module	0,5
BONUS	Le code est écrit en ES6 ou ultérieur	1
BONUS	Le projet fourni un README expliquant l'installation et le lancement du projet	0,5
BONUS	L'application affiche les choix de QCM de façon aléatoire	1
BONUS	L'application affiche les questions dans un ordre aléatoire	2
	TOTAL	**56**

Les point bonus seront comptabilisés si et seulement si la note est déjà supérieur à 28/56.