

Bloc 2

Session de septembre 2021

Examen de Javascript / Node.js

Professeurs : Raphaël Baroni, Jean-Luc Collinet

Date et heure : 25 août de 13h à 16h

Lieu : à domicile

Durée de l'examen : 3h

Modalités : cf. **Directives** ci-dessous

Cotation : Javascript sur 20 points qui valent 75 % de la note finale de l'UE BINV2150.

Directives

Vous trouverez sur <https://evalmoodle.vinci.be/course/view.php?id=356>

un fichier **examenJS.zip**.

Cette archive contient :

- l'énoncé dans le document **2021.09-Session2-Examen.v.Finale.pdf**,
- un fichier JS, utile pour réaliser les fonctionnalités demandées, **default_proposals.js**.

Bon travail à tous !

Remise le 25 août avant 16h

1. Créez un fichier **.zip** nommé **NOM_PRENOM.zip**
par exemple : STALLMAN_RICHARD.zip
2. Vérifiez bien votre **.zip** avant de le poster.
3. Remettez ce fichier **.zip** sur Evalmoodle dans le devoir Examen de Javascript – Août 2021 <https://evalmoodle.vinci.be/mod/assign/view.php?id=16691>

La collaboration entre les étudiants est interdite et sera donc lourdement sanctionnée si elle se produit. Un outil de détection de plagiat sera utilisé.

Nous vous recommandons d'abord de lire l'énoncé en entier.

Nous vous demandons de séparer vos réponses en 2 répertoires bien distincts, chaque répertoire contenant votre réponse à une des 2 questions de l'énoncé :

1. Le dossier **NOM_PRENOM/question01** doit contenir tout le code concernant la question 1 : « frontend only »
2. Le dossier **NOM_PRENOM/question02** doit contenir tout le code concernant la question 2 : « RESTFul API et SPA »

Si une question d'examen ne s'exécute pas ou ne donne aucun résultat fonctionnel, vous aurez d'office moins de 50% des points pour cette question.

Enoncé

1 Contexte

Nous allons développer une petite application permettant de gérer des demandes d'étudiants, au sens large : cela peut être des propositions d'améliorations dans le cadre scolaire.

Plus particulièrement, durant ces 3 prochaines heures, vous allez offrir les moyens à des professeurs de voir des demandes d'étudiants et de faire évoluer le statut de leurs demandes. Une demande est d'abord « soumise », puis peut être « acceptée, en cours d'élaboration » ou « refusée, ne sera pas mise en œuvre », et, en fin de travail après avoir été acceptée : « terminée ».

2 Question 1 : « *frontend only* » (8 points)

- **Programmez dans le dossier NOM_PRENOM/question01**
- Vous pouvez utiliser n'importe quelle librairie JS (Bootstrap, JQuery, etc.).
- Seuls des fichiers .html, .css, .js, .json, .png et .jpg sont permis.
- Webpack n'est pas nécessaire ici, mais si vous l'utilisez, veuillez ne pas fournir le répertoire node_modules, mais fournissez le fichier package.json référençant toutes les dépendances.

2.1 Développement d'une version statique des composants de votre SPA et de la navigation entre ces composants (4 points)

Dans un premier temps, vous allez développer une version statique des composants visuels de votre SPA et vous assurer que la navigation entre ces composants fonctionne. Par la suite, vous allez donner vie à chacun des composants de votre SPA.

Veuillez prévoir deux boutons qui seront toujours affichés sur votre SPA. Le bouton « Demandes » permet d'afficher le composant visuel permettant la saisie d'une demande d'un étudiant.

Le bouton « Voir les demandes » amène l'affichage des demandes afin de suivre leurs évolutions.

Voici un exemple :

Poser une demande

Voir les demandes

Au chargement de la SPA, seuls les deux boutons sont affichés.

Lors d'un clic sur le bouton « Poser une demande », veuillez afficher le composant visuel permettant la saisie d'une demande :

Je pose une demande :

Dans cette partie, nous ne traitons pas l'envoi de la demande.

Ce composant visuel de saisie d'une demande comprend :

- un texte : « Je pose une demande : »,
- un input pour pouvoir saisir une demande qui sera soumise par un étudiant. Ce champ de saisie doit être suffisamment grand pour saisir un long texte,
- un bouton pour envoyer une demande.

Lors d'un clic sur le bouton « Voir les demandes », veuillez afficher le composant visuel permettant de voir les demandes et leurs statuts :

Voici les demandes et leurs statuts :

id	proposal	status
1	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	submitted
2	Puis-je vous demander de rajouter du papier dans les toilettes du 1er étage du bâtiment B ?	accepted
3	Puis-je vous demander un distributeur de friandises dans le foyer des étudiants ?	refused
4	Puis-je vous demander d'augmenter la puissance du Wifi dans le local 026 ?	done

Ce composant visuel pour voir les demandes comprend :

- un texte : « Voici les demandes et leurs statuts : »,
- une table HTML dont les titres de colonnes sont les clés de la structure JSON dans le **default-proposals.js**, les valeurs sont les exemples de demandes dans cette même structure.

Contraintes d'implémentation :

- Vous développez une SPA. Pour rappel, on ne charge qu'une seule fois le frontend au niveau d'une SPA.
- Ce que vous développez ne doit pas être stylé.
- Vérifiez que votre navigation, via vos deux boutons, affiche bien l'un ou l'autre composant visuel ;

- Veuillez intégrer le fichier **default-proposals.js** à votre projet en l'utilisant comme module. Ce fichier retient les demandes et leurs statuts. Vous pouvez mettre à jour le contenu de ce fichier pour pouvoir l'utiliser dans votre code JS.

2.2 Changement dynamique des statuts (4 points)

Vous allez rendre dynamique le composant visuel de saisie pour voir les demandes, développé au point précédent, en deux étapes.

Voici les demandes et leurs statuts :

id	proposal	status
1	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	submitted ▼
2	Puis-je vous demander de rajouter du papier dans les toilettes du 1er étage du bâtiment B ?	accepted ▼
3	Puis-je vous demander un distributeur de friandises dans le foyer des étudiants ?	refused ▼
4	Puis-je vous demander d'augmenter la puissance du Wifi dans le local 026 ?	done ▼

Dans cette partie, nous ignorons entièrement l'authentification de l'utilisateur professeur.

Contraintes d'implémentation :

- Les statuts sont présentés avec des SELECT en HTML,
- Le statut de chaque demande est correctement affiché par défaut dans chaque SELECT.

Voici les demandes et leurs statuts :

id	proposal	status
1	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	accepted ▼
2	Puis-je vous demander de rajouter du papier dans les toilettes du 1er étage du bâtiment B ?	accepted ▼
3	Puis-je vous demander un distributeur de friandises dans le foyer des étudiants ?	refused ▼
4	Puis-je vous demander d'augmenter la puissance du Wifi dans le local 026 ?	done ▼

Vous avez sélectionné : accepted pour la demande dont l'ID est : 1

Dans cette partie, nous ne faisons pas persister les changements de statut.

Contraintes d'implémentation :

- après avoir changé un statut, par exemple le statut de la demande dont l'id est 1, une notification s'affiche,
- cette notification s'affiche lors de chaque événement « onchange » d'un SELECT.

3 Question 2 : RESTFul API et SPA (12 points)

Vous allez maintenant développer une RESTFul API et une SPA monolithique consommant cette API.

Contrainte d'implémentation : votre SPA doit être accessible localement sur votre machine, via l'URL <http://localhost:2021>

- **Programmez dans le dossier NOM_PRENOM/question02**
- Vous pouvez créer le projet JS sous VS Code en utilisant le générateur d'application d'Express pour votre backend.
- Vous pouvez utiliser n'importe quel package côté serveur (Express, etc.).
- Seuls des fichiers .html, .css, .js, .json et .png sont permis.
- Veuillez ne pas fournir le répertoire node_modules, mais fournissez le fichier package.json référençant toutes les dépendances.

3.1 RESTFul API (6 points)

Au niveau de la RESTFul API, tout en respectant les conventions REST appliquées dans le cadre de notre cours, veuillez développer ces trois opérations sur des ressources de type « demande » :

- **Lecture des demandes (GET /api/proposals)**
Toutes les demandes sont envoyées au navigateur client depuis le serveur.
- **Création d'une demande (POST /api/proposals)**
Une demande comprend un identifiant, un contenu textuel (le texte de la demande) et le statut qui est initialisé par défaut à « submitted »
- **Mise à jour d'une demande (POST /api/proposals/:id)**
Une demande peut être mise à jour en connaissant son identifiant, seul le statut est à changer.

Contraintes d'implémentation :

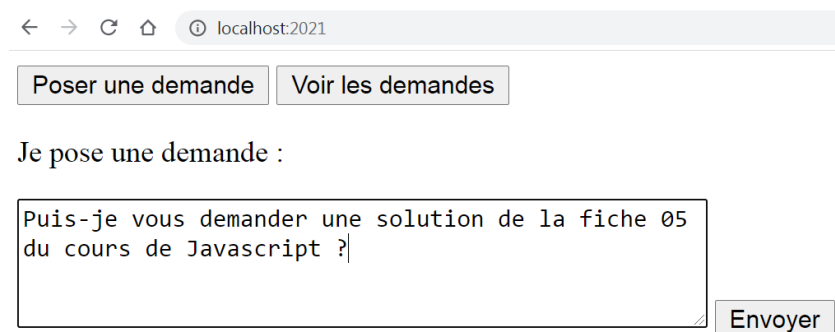
- Les demandes qui sont créées doivent persister et donc survivre au redémarrage de votre application. Vous pouvez faire persister les demandes côté serveur de la manière que vous voulez, du moment que cela soit en JSON.
- Vous pouvez choisir un système pour attribuer les identifiants automatiquement.

3.2 SPA et RESTFul API (6 points)

Nous allons maintenant consommer les trois opérations de la RESTFul API au niveau de la page chargée par défaut lors de l'accès à l'application.

Vous pouvez intégrer le frontend que vous avez développé dans le cadre de la question 1.

Lors d'un clic sur le bouton « Poser une demande », veuillez afficher le composant visuel permettant la saisie d'une demande :



← → ↻ 🏠 localhost:2021

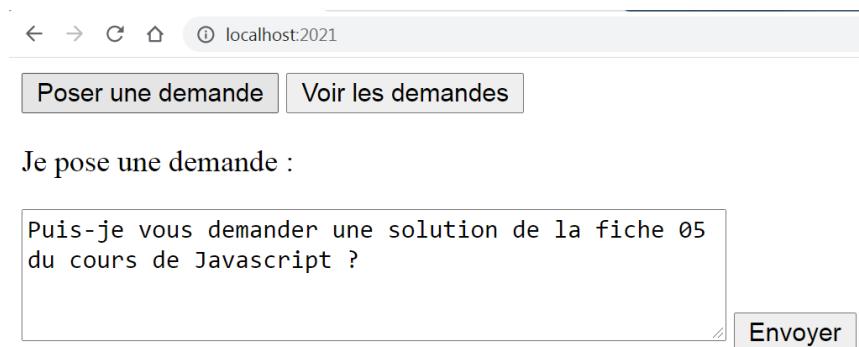
Poser une demande Voir les demandes

Je pose une demande :

Puis-je vous demander une solution de la fiche 05
du cours de Javascript ?

Envoyer

Lors du clic de l'utilisateur sur le bouton Envoyer, la demande est envoyée et il faut notifier l'utilisateur :



← → ↻ 🏠 localhost:2021

Poser une demande Voir les demandes

Je pose une demande :

Puis-je vous demander une solution de la fiche 05
du cours de Javascript ?

Envoyer

Votre demande a été envoyée

Haute Ecole Léonard de Vinci
Paul Lambin - Baccalauréat en Informatique
Examen de Javascript

En cliquant sur le bouton « Voir les demandes », la demande est maintenant visible :

A screenshot of a web browser window with the address bar showing 'localhost:2021'. Below the address bar, there are two buttons: 'Poser une demande' and 'Voir les demandes'.

Voici les demandes et leurs statuts :

id	proposal	status
KS6kBPrlF	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	submitted ▾

Le statut est par défaut « submitted ».

Cette seconde étape permet d'afficher un tableau, par exemple ainsi :

A screenshot of a web browser window with the address bar showing 'localhost:2021'. Below the address bar, there are two buttons: 'Poser une demande' and 'Voir les demandes'.

Voici les demandes et leurs statuts :

id	proposal	status
KS6kBPrlF	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	submitted ▾
Uee07vQTR	Puis-je vous demander de rajouter du papier dans les toilettes du 1er étage du bâtiment B ?	accepted ▾
NOelwNAAV	Puis-je vous demander un distributeur de friandises dans le foyer des étudiants ?	refused ▾
-5FL4FfhH	Puis-je vous demander d'augmenter la puissance du Wifi dans le local 026 ?	done ▾

Contraintes d'implémentation :

- Dans le cas où il n'y a pas encore de demandes, affichez une notification adéquate.

A screenshot of a web browser window with the address bar showing 'localhost:2021'. Below the address bar, there are two buttons: 'Poser une demande' and 'Voir les demandes'.

Voici les demandes et leurs statuts :

Aucune demande pour l'instant.

L'étape finale est de rendre fonctionnel le changement de statut :

[←](#) [→](#) [↺](#) [🏠](#) [localhost:2021](#)

[Poser une demande](#) [Voir les demandes](#)

Voici les demandes et leurs statuts :

id	proposal	status
KS6kBPrlF	Puis-je vous demander une solution de la fiche 05 du cours de Javascript ?	accepted ▾
Uee07vQTR	Puis-je vous demander de rajouter du papier dans les toilettes du 1er étage du bâtiment B ?	accepted ▾
N0elwNAAV	Puis-je vous demander un distributeur de friandises dans le foyer des étudiants ?	refused ▾
-5FL4FfhH	Puis-je vous demander d'augmenter la puissance du Wifi dans le local 026 ?	done ▾

Vous avez changé : [accepted](#) pour la demande dont l'ID est : KS6kBPrlF

Si par exemple l'utilisateur choisit « [accepted](#) », le statut est mis à jour directement et une notification s'affiche.

Contraintes d'implémentation :

- Vous ne devez pas gérer l'authentification. C'est bien sûr un professeur qui pourra changer les statuts et non un étudiant. Cette gestion des rôles ne fait pas partie de l'examen.
- Vous ne devez pas vérifier que le workflow est respecté, en d'autres termes, l'utilisateur peut changer le statut comme il le désire.
- Les quatre statuts sont : « [submitted](#) », « [accepted](#) », « [refused](#) » et « [done](#) ».