Samba Joan Roger Mabanza

Résumé

Explication de l’hébergement de Node.JS

Documentation TP2

Application Web II

Table des matières

[Hébergement de Node.JS 2](#_Toc149306175)

[Référence : 6](#_Toc149306176)

# Hébergement de Node.JS

L’hébergement de node.js est fait sur le site <https://render.com/docs/deploy-node-express-app>. D’abord, on doit créer un compte ou utiliser un compte GitHub ou GitLab ou Google.

Pour notre cas, on a utilisé nos comptes GitHub pour se connecter. Après, on clique sur le bouton « New + » puis Web Service puis cliquer sur « Build and deploy from a git repository ». Puis, on connecte le projet qui est sur GitHub. On va avoir une page avec des champ qu’on va remplir. On donne un nom à notre projet déployé puis dans le « Root Directory » on met le nom du dossier à déployé qui est « node » dans notre cas. Puis on sélectionne Node dans Runtime. Dans « Build Command » on écrit yarn puis dans « Start Command » on met node server.js qui représente le nom du fichier où le code en node est écrit. Le déploiement est gratuit. Donc, on clique sur le bouton « Create Web Service » et on attend jusqu’à qu’il soit déployé.

Une image contenant texte, capture d’écran, Police, conception

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Une image contenant texte, capture d’écran, logiciel, nombre

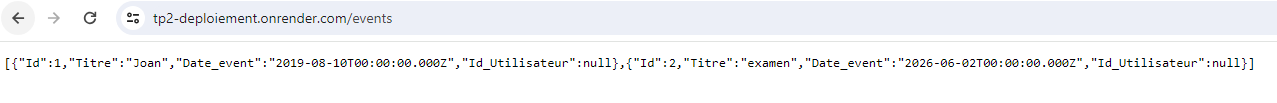
Description générée automatiquement

Une image contenant texte, capture d’écran, nombre, document

Description générée automatiquement

Une image contenant texte, capture d’écran, logiciel, nombre

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Une image contenant texte, capture d’écran, nombre, ligne

Description générée automatiquement

# Utiliser FullCalendar :

On utilise le calendrier pour créer, afficher et supprimer des événements à une date précise. On utilise des méthodes qui envoient des requêtes du service REST par le biais du serveur Node pour communiquer avec la base de données évènements. On a changé aussi la langue du calendrier de l’anglais au français.

La fonction **chargerEvenements** permet de charger les événements stockés en base de données lorsque l’utilisateur est connecté. On crée de nouveaux objets en mappant les résultats de la requête envoyée afin de contenir le titre et la date d’événement, cette dernière est formatée de façon à ressortir « année-mois-jour » en chaine de caractères grâce à la librairie « moment » qui permet de spécifier l’horodatage avec un format précis, pour notre cas, nous aurons besoin que de la date du jour.

La fonction **useEffect** est une fonction React pour les composants de type fonction, elle permet, à chaque rafraichissement de page, de recharger les événements gardés dans la base de données afin qu’ils soient présents dans le calendrier.

La fonction **ajouterEvenement** permet d’ajouter un événement en écrivant son titre et sa date en prompt dans le navigateur. Si le titre et la date ne sont pas nuls, l’ajout est automatique, puis le chargement d’événements s’en suit. Ce procédé nécessite de cliquer sur le bouton nommé « *Ajouter un événement* ».

La fonction **supprimerEvenement** permet de supprimer un événement lorsqu’on clique gauche sur lui. Le pop-up demandant la suppression d’événement apparaitra, dépendant du choix de l’utilisateur, l’événement sera d’abord supprimé dans la base de données, et ensuite, lors du rechargement d’événements dans l’application, il disparaitra du calendrier.

Le FullCalendar contient une propriété « eventClick » qui appelle une fonction fléchée qui prend un argument info qui est un objet contenant des informations sur l'événement sur lequel l'utilisateur a cliqué. Celle-ci sera utilisée pour appeler la fonction **supprimerEvenement.**

# Local Storage :

Le localStorage est une propriété de l’interface « window » pour le stockage et l'accès de données depuis un objet de type *Storage*, elle marche avec le principe de clé/valeur lors du stockage d’une donnée.

La gestion de l'ID de session dans un environnement web peut se faire de différentes manières, et l'utilisation du Local Storage ou des variables de session présente chacune des avantages et des inconvénients distincts :

|  |  |
| --- | --- |
| localStorage | sessionStorage |
| Avantages | Inconvénients |
| **Plus persistant** dans les données car elles sont gardées même en réouvrant le navigateur pour une session ultérieure. Suffit de supprimer tous les éléments pour perdre ses données | **Moins Persistant** dans les données car elles sont effacées à la fermeture du navigateur. |
| **Stockage de données** jusqu’à 5 ou 10 Mo | **Stockage de données** dépendant de la portée du navigateur et du serveur utilisés |
| **Accessibilité des données** depuis l’API Web Storage | **Accessibilité restreinte** car les données stockées dans le serveur surchargent cette dernière et limitent ses capacités |
| Inconvénients | Avantages |
| **Très peu sécurisé** au niveau des requêtes http car les données localStorage sont autant accessibles du côté-serveur que du côté-client, ce qui peut donner occasion à des attaques XSS. | **Mieux sécurisé** au niveau des requêtes http car les variables de session sont stockées côté serveur, limitant ainsi les attaques XSS |
| **Expiration automatique inexistant,** pour faire disparaitre des données automatiquement il faut procéder à des fonctionnalités supplémentaires | **Expiration automatique intégrée** afin de permettre à des données, pour une question de sécurité, d’expirer à une certaine période. |
| **Aucune synchronisation avec serveur** car ses données ne dépendent pas de l’état de la session côté serveur | **Synchronisation avec serveur** puisque les applications dépendent de l‘état de la session côté serveur |

### Conclusion

Le choix entre le Local Storage et les variables de session dépend des besoins spécifiques de l'application, notamment en termes de sécurité, de durée de vie des données, de capacité de stockage et de synchronisation avec le serveur. Pour les applications nécessitant une haute sécurité et une synchronisation serveur, les variables de session sont préférables. Pour une expérience utilisateur persistante et facile d'accès, le Local Storage peut être plus adapté.

# Référence :

Render :

<https://render.com/docs/deploy-node-express-app>

[Stockage des données locales : Web Storage - Alsacreations](https://www.alsacreations.com/article/lire/1402-web-storage-localstorage-sessionstorage.html)

ChatGpt