Une image contenant capture d’écran, texte, conception

Description générée automatiquement

# TP2 Première requêtes http vers node.js avec POSTMA ?

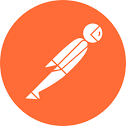


Prérequis , pour réussir ce TP vous devez à voir un server node.js installé ainsi que le gestionnaire de module npm. Vous devez savoir créer un projet node.js avec npm init et installer le module Express . ( Voir TP 1 )

Une image contenant capture d’écran, Bleu électrique, bleu, conception

Description générée automatiquement

Pour suivre correctement ce TP vous devez télécharger Wireshark et l’installer.



Pour suivre correctement ce TOP vous devez télécharger PostMan et l’installer.

## Partie 1 : Préparation du back

En reprenant les indications de l’installation de node.js du TP1, refaire un dossier de projet , connectez vous avec vscode ou samba dans ce dossier . Attention il faut que votre dossier projet ai tous les droits **( chmod 777 dossierProjet** ) et il faut être connecté sur votre terminal en **su – root**

**Attention quand vous démarrer le terminal de bien vous positionner dans votre dossier projet ( cd dossierProjet )**

1. Quelle est la commande npm pour créer un projet node.js dans votre dossier  ?

Lancer cette commande pour initialiser votre projet node.js

1. Quelle est la commande npm pour le module express dans votre dossier projet  ?

Lancer cette commande pour installer le module Express

1. A quoi sert le module Express  ?
2. Quelle est la commande npm pour lancer votre cote coté back ?

Lancer votre code et vérifier que vous arrivez à afficher votre page

## Partie 2 : Création de l’application server en GET

Pour la suite il faut Créer un fichier **application.js** dans votre dossier projet

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

Description générée automatiquement

Pour coder ce fichier application.js on va avoir besoin de :

1. Intégrer le module express
2. Créer une instance d’express
3. Déterminer une variable pour le port
4. Faire une route pour interception la racine du site en get sur l’instance d’express
5. Coder la fonction fléché de la route , elle doit juste afficher un message
6. Mettre le server en listen sur le port déterminé

Les étapes a) à f) peuvent être trouvé dans la leçon ou le tp1

Mais rajouter dans votre code cette partie pour pouvoir afficher des messages d’erreur au cas ou.

// Gestionnaire d'erreurs global

process.on('uncaughtException', (err) => {

console.error(`Une erreur non capturée s'est produite : ${err.message}`);

process.exit(1); // 1 signifie une sortie avec une erreur

});

Faites le code

1. Quelle est la commande node pour lancer l’application.js coté back ?

Lancer votre code et vérifier que vous arrivez à afficher votre page depuis un navigateur

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

Description générée automatiquement

1. Les lignes de mon code sont ici dans le désordre mettez la lettre de l’étape a) à f) associé ?

|  |  |
| --- | --- |
| Ligne de code : | Lettre de l’étape associé de a) à f) |
| app.listen(port, () => {  console.log(`Serveur port ${port}`);  }); |  |
| app.get('/', fonction() ); |  |
| const app = expres(); |  |
| const port = 8080; | Exemple : **c** |
| const express = require('express'); |  |
| (req, res) => {es.send('message'); |  |

Astuce en ajoutant ce module : const os = require('os');

il est possible d’avoir des info sur les interfaces réseau, on peut récupérer l’ip du server avec la ligne : os.networkInterfaces()['ens18'][0].address

il faut bien sur mettre votre bonne interface , sur une VM c’est ens18 mais sur d’autre machine l’interface ( ip a ) aura peut être un nom différent. Cela me permet de faire ceci :

// Démarrer le serveur

const server = app.listen(port, () => {

//console.log(os.networkInterfaces());

console.log(`Serveur est en écoute sur

${os.networkInterfaces()['ens18'][0].address}:${port}`);

});

Ce qui affiche quand on lance le server :



pratique quand on gère plusieurs VM :D

1. Modifier le message de votre route et recharger votre page web coté front sans redémarrer node.js . Est-ce que la modification à eu lieu  ?

Pour le moment si vous voulez mettre à jour votre server , il faut faire dans la console : **ctrl + C** pour stopper le server puis le redémarrer avec la commande **node fichier.js**

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

## Postman Icon Vector SVG Icon - SVG RepoPartie 3 : Découverte d’un client d’api POSTMAN.

Installer PostMan si ce n’est pas déjà fait.

Postman est un outil très populaire utilisé par les développeurs pour tester les API (Interfaces de Programmation d'Applications) RESTful. Il offre une interface utilisateur conviviale permettant de créer, envoyer et recevoir des requêtes HTTP vers des API, ainsi que de visualiser les réponses. Postman est largement utilisé par les développeurs d'API, les testeurs, les architectes d'API et d'autres professionnels du développement logiciel pour simplifier le processus de test, de débogage et de documentation des API.

Une image contenant texte, logiciel, capture d’écran

Description générée automatiquement

L’interface peut changer d’une version à une autre. L’idée et de trouver comment créer sa première requête http GET

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

Description générée automatiquement

Il faut en premier choisir le type de requete ici GET ou POST et saisir l’adresse de votre API. Ici notre API est notre server.js qui ne sait pour le moment répondre qu’a une seul route , la racine de notre ip.

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

Description générée automatiquement

En cliquant sur send, vous envoyez la requête

1. Chercher dans les boutons de cette requêtes l’endroit pour afficher les propriétés en tetes de la réponse http. Citez les toutes et donnes leur définition :

|  |  |
| --- | --- |
| Nom de la propriété d’entête | Définition |
| Exemple : Date | C’estla date de la requête |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Partie 3 : Modification de l’application pour répondre à une requête http POST

Dans la partie 2 nous avont fait une simple requête de type GET. ( même s’il est possible d’envoyer des données au server via la méthode GET ce que nous verrons plus tard ) il est préférable d’utiliser la méthode POST pour envoyer des données a notre server.

Rajouter une route à votre code server.

Voici le code partiel à ajouter :

* L’utilisation du module pour lire le body d’une requête

const bodyParser = require('body-parser');

* Le middleware qui sait décoder un body
* //middleware
* app.use(bodyParser.json()); // décode le body d'une requête
* Et la route :

app.post('/', (req, res) => {

const donneesDuCorps = req.body;

console.log(donneesDuCorps);

res.send('Données reçues et traitées !');

});

**Redémarrer le server après toutes modifications**

1. Que représente req.body ?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Du coté de POSTMAN, créer une nouvelle requête (bouton + ) de type post.

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

Description générée automatiquement

Appuyer sur Send pour envoyer votre requête post.

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, Police, nombre  Description générée automatiquement  Cas 1 : Votre server fonctionne | Une image contenant texte, capture d’écran, Police, nombre  Description générée automatiquementUne image contenant texte, capture d’écran, Police, nombre  Description générée automatiquement  Cas 2 : Votre route n’est pas correct ou alors vous n’avez pas redémarré votre server.js |

Comme du coté server vous avez mis le résultat dans la console du back vous devez observer ceci

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

Undefined car vous n’avez pour le moment envoyer aucune donnée dans votre body de requête post.

Puis trouvez l’onglet pour ajouter des données au body de la requête au format JSON

Et placer dans la zone attendu de postman le json suivant :

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

Description générée automatiquement

Envoyé la requête de nouveau et regarder la console de votre terminal . vous devriez avoir ceci :

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

Description générée automatiquement

Bravo vous avez réussi à envoyer des données du Front ( client Postman ) vers un server d’api ( node.js )

1. Chercher dans les boutons de cette requêtes l’endroit pour afficher les propriétés en tetes de la requêtes http ( pas de la réponse ). Citez les toutes et donnes leur définition :

|  |  |
| --- | --- |
| Nom de la propriété d’entête | Définition |
| 1 )Exemple : Accept \*/\* | indique au serveur quels types de médias (contenus) le client est capable de traiter ou de "prendre en charge" \* = tous |
| 2) |  |
| 3) |  |
| 4) |  |
| 5) |  |
| 6) |  |
| 7) |  |