Tout d'abord, qu'est-ce que Node.js ? Node.js est une plateforme logicielle open-source qui permet d'exécuter du code JavaScript côté serveur. Avant Node.js, JavaScript était principalement utilisé pour le développement côté client, c'est-à-dire dans les navigateurs Web. Node.js permet aux développeurs de créer des applications Web et des API en utilisant JavaScript, ce qui facilite la communication entre le client et le serveur.

L'utilité de Node.js réside dans plusieurs aspects :

1. Unification du langage de programmation : Node.js permet aux développeurs d'utiliser JavaScript, un langage très populaire, à la fois pour le développement côté client et côté serveur. Cela simplifie le processus de développement et facilite la maintenance des applications.
2. Performance : Node.js est construit sur le moteur V8 de Google Chrome, qui est un moteur d'exécution JavaScript très performant. De plus, Node.js utilise un modèle d'exécution non bloquant et orienté événement, ce qui le rend adapté aux applications en temps réel et à haut débit.
3. Écosystème riche : Node.js dispose d'un vaste écosystème de modules accessibles via npm (Node Package Manager), qui permet aux développeurs d'ajouter rapidement des fonctionnalités à leurs applications sans devoir réinventer la roue.
4. Scalabilité : Node.js est conçu pour être facilement scalable, permettant aux applications de gérer un grand nombre de connexions simultanées avec une faible latence. ( La scalabilité, également connue sous le nom de "mise à l'échelle" ou "extensibilité", fait référence à la capacité d'un système, d'une application ou d'un processus à gérer efficacement l'augmentation de la charge de travail sans compromettre les performances. En d'autres termes, il s'agit de la capacité à s'adapter à une charge croissante en ajoutant des ressources au système, telles que de la mémoire, du stockage ou de la puissance de calcul.   
   Dans le contexte de Node.js, la scalabilité se réfère à la capacité d'une application à gérer un grand nombre de connexions simultanées et à traiter efficacement un volume croissant de requêtes avec des temps de réponse acceptables.)

Passons maintenant au fonctionnement de Node.js :

Node.js utilise un modèle d'exécution basé sur une boucle d'événements. Cela signifie que Node.js attend les événements (comme les requêtes entrantes) et exécute les fonctions associées à ces événements de manière non bloquante. En d'autres termes, Node.js peut traiter plusieurs requêtes simultanément sans attendre la fin d'une tâche avant de passer à la suivante.

Ce modèle d'exécution non bloquant permet à Node.js de gérer efficacement les applications en temps réel et les situations où il y a beaucoup de requêtes simultanées. Il est particulièrement adapté aux applications Web modernes, comme les messageries instantanées, les jeux en ligne, les plateformes de collaboration et les API RESTful.

Pour conclure, Node.js est une plateforme puissante et flexible qui permet aux développeurs de créer des applications Web et des API performantes et évolutives en utilisant JavaScript. Son écosystème riche, sa performance et sa facilité d'utilisation en font un choix populaire pour le développement d'applications Web modernes.

Site permettant de rechercher si un package existe :  
[npm (npmjs.com)](https://www.npmjs.com/)

[npms](https://npms.io/)

Exemple d’installation et d’utilisation de Bootstrap :

Lorsque vous installez Bootstrap à l'aide de npm, les fichiers nécessaires sont téléchargés dans votre projet, généralement dans le répertoire node\_modules/bootstrap/. Pour utiliser Bootstrap dans votre page HTML, vous devez inclure les fichiers CSS et JavaScript appropriés. Voici comment vous pouvez le faire :

1. Inclure le fichier CSS : Vous devez inclure le fichier CSS de Bootstrap dans la balise <head> de votre document HTML. Le fichier se trouve généralement dans node\_modules/bootstrap/dist/css/. Voici un exemple d'inclusion du fichier CSS ::

<!DOCTYPE html>

<html lang="fr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Exemple Bootstrap</title>

<link rel="stylesheet" href="node\_modules/bootstrap/dist/css/bootstrap.min.css">

</head>

<body>

<!-- Votre contenu HTML ici -->

</body>

</html>

1. Inclure les fichiers JavaScript : Pour utiliser les fonctionnalités JavaScript de Bootstrap (comme les modals, les tooltips, etc.), vous devez également inclure les fichiers JavaScript nécessaires. Bootstrap dépend de jQuery et de Popper.js, donc vous devez les inclure avant le fichier JavaScript de Bootstrap. Ajoutez les fichiers JavaScript à la fin de votre balise <body> :

<!DOCTYPE html>

<html lang="fr">

<head>

<!-- ... -->

<link rel="stylesheet" href="node\_modules/bootstrap/dist/css/bootstrap.min.css">

</head>

<body>

<!-- Votre contenu HTML ici -->

<!-- Inclure jQuery et Bootstrap.js -->

<script src="node\_modules/jquery/dist/jquery.min.js"></script>

<script src="node\_modules/bootstrap/dist/js/bootstrap.min.js"></script>

</body>

</html>

Notez que les chemins vers les fichiers CSS et JavaScript de Bootstrap dans les exemples ci-dessus peuvent varier en fonction de la structure de votre projet. Assurez-vous d'utiliser les chemins corrects pour les fichiers dans votre projet.

Exemple d’installation et d’utilisation de jQuery avec npm : [Utiliser le module jQuery NPM | Delft Stack](https://www.delftstack.com/fr/howto/jquery/npm-jquery/?utm_content=cmp-true)