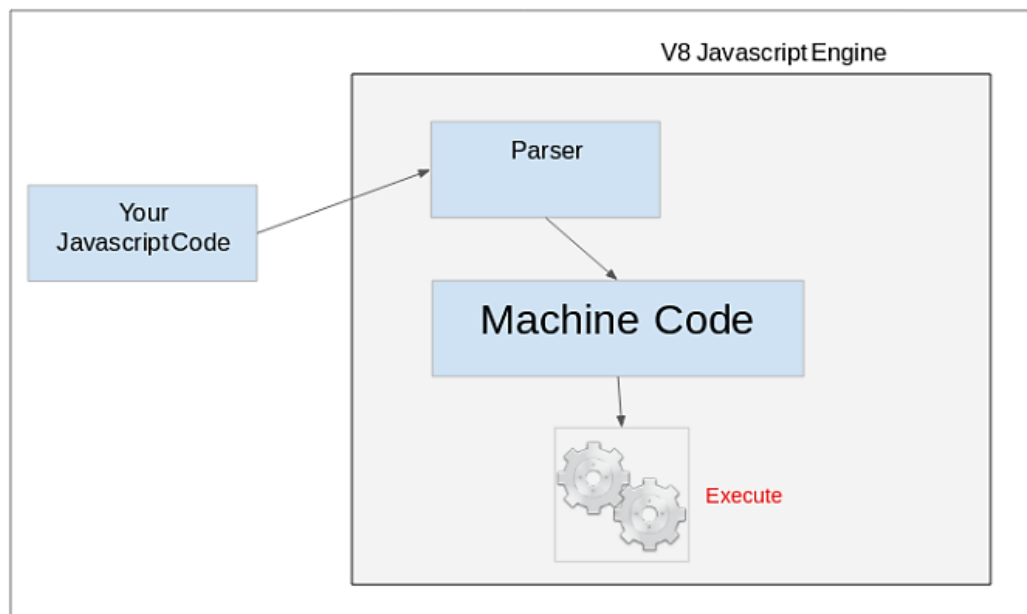


## OBJECTIF :

Appliquer JavaScript sur des technologies **back-end** (Node, Express et MongoDB) afin de créer des serveurs complets permettant d'interagir avec des bases de données.

## LES OUTILS BACK-END :

- Installation Node installe également **Node Package Manager** ou npm : c'est l'outil précieux pour l'installation des packages nécessaires à la création des projets.
- Chrome est un navigateur Google reconnu et gratuit avec la première version publiée en décembre 2008, dont V8 JavaScript Engine (La machine Javascript) est un programme écrit sur C++, le code de source ouverte, utilisé dans Google Chrome pour analyser et exécuter le code Javascript à hautes performances. Il analyse (parse) syntaxe Javascript et l'interprète en code de l'ordinateur pour l'exécution.



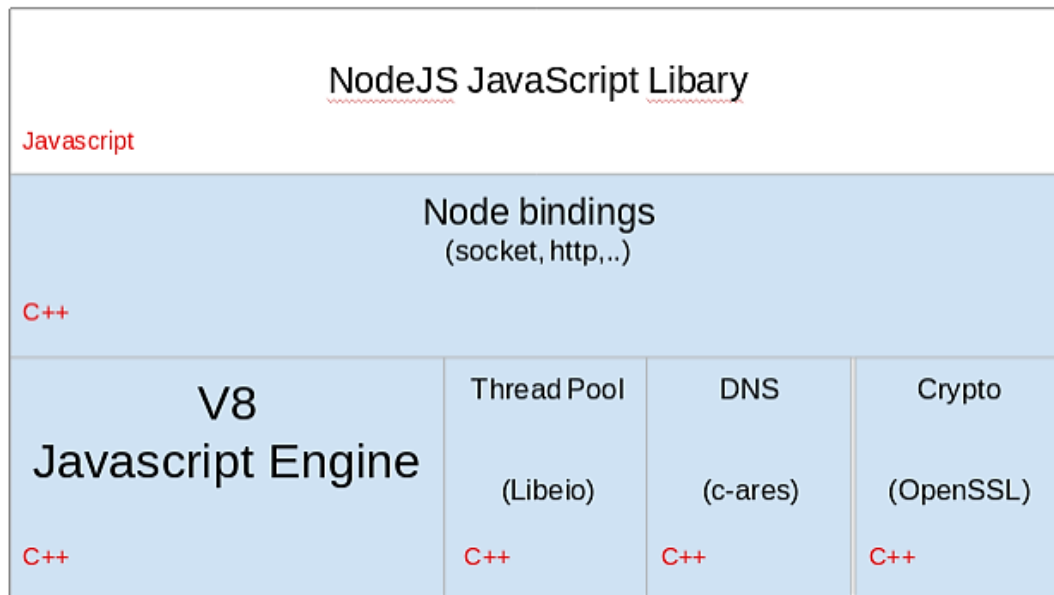
## NODEJS

**Node** est le *runtime* qui permet d'écrire toutes les tâches côté serveur, en JavaScript, telles que le logique métier, la persistance des données et la sécurité. Node ajoute également des fonctionnalités que le JavaScript du navigateur standard ne possède pas, comme par exemple l'accès au système de fichiers local.

Exemple :

```
const mysql = require('mysql');
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'user',
  password: 'password',
  database: 'database_name'
});
connection.connect((err) => {
  if (err) throw err;
  console.log('Connected!');
});
```

Et **NodeJS** était né. Le V8 Javascript Engine est l'un des composants du **NodeJS**. Voici l'image de l'architecture du NodeJS :



En bref, le **NodeJS** peut remplacer PHP, Java pour construire des applications web du côté Server. Au lieu d'écrire du code par PHP, Java utilise la syntaxe de Javascript.

Le NodeJS est l'environnement d'exécution JavaScript (JavaScript Runtime Environment) de l'extérieur navigateur. Le NodeJS comprend également autres composants et autres bibliothèques pour qu'il puisse fonctionner comme un Web Application Server.

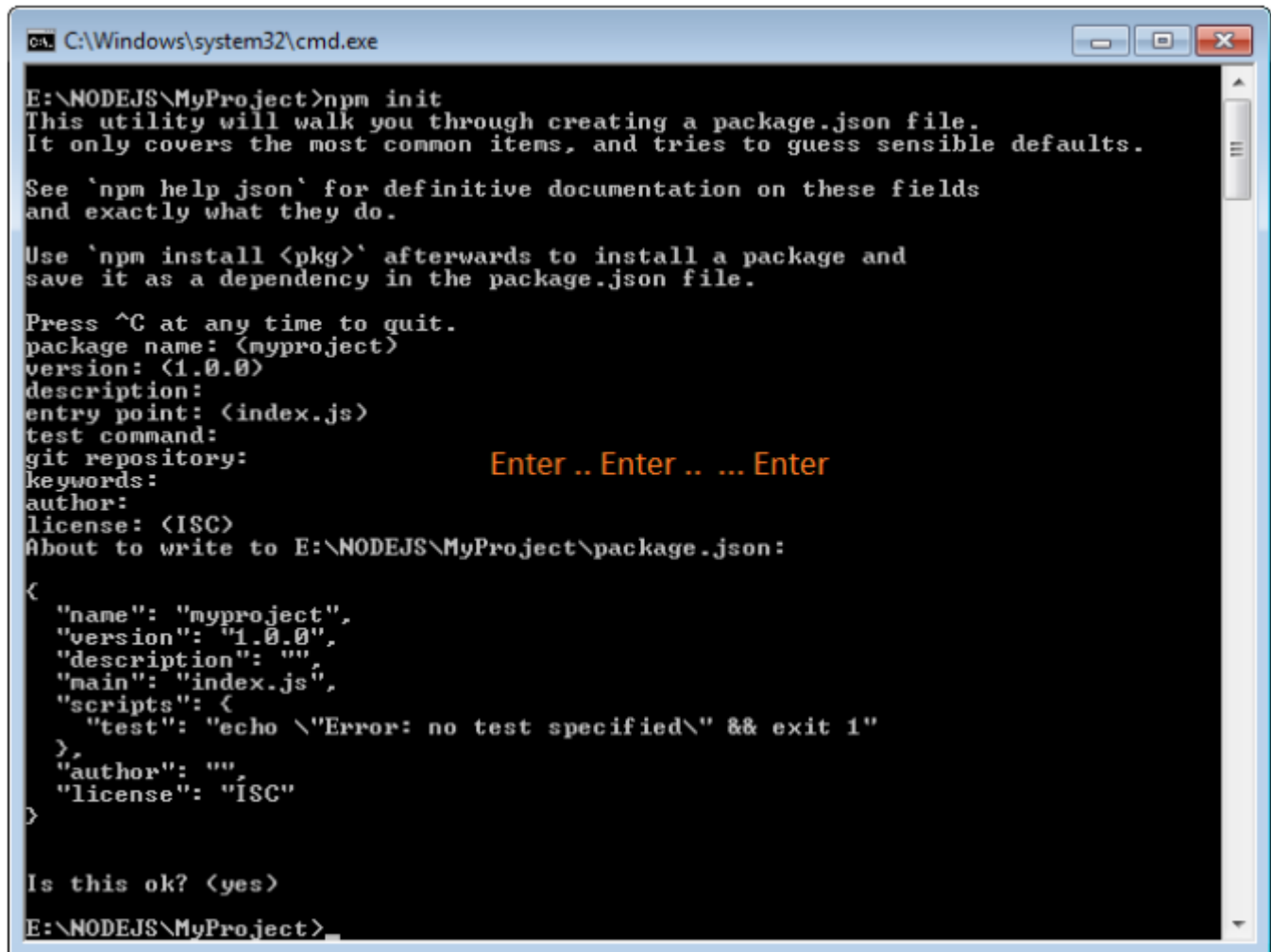
Remarque :

NPM est l'abréviation de Node Package Manager, qui est un outil (programme) gérant les bibliothèques de programmation Javascript pour Node.js. Cet outil est réellement nécessaire pour le monde open source. Dans la communauté Javascript, les développeurs partagent des centaines de milliers des extraits de code qui aident les nouveaux projets à éviter de réécrire des composants de base, des bibliothèques de programmation ou même des frameworks.

**Express** est un **framework** reposant sur Node, qui facilite la création et la gestion des serveurs Node.

## CREATION D'UN PROJET NODEJS

- Tout d'abord, il faut créer un dossier nommé MyProject, par exemple,
- Ouvrez des fenêtres CMD et CD et allez vers le dossier que vous avez créé :



```
C:\Windows\system32\cmd.exe

E:\NODEJS\MyProject>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help json' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: <myproject>
version: <1.0.0>
description:
entry point: <index.js>
test command:
git repository:
keywords:
author:
license: <ISC>
About to write to E:\NODEJS\MyProject\package.json:
{
  "name": "myproject",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this ok? <yes>
E:\NODEJS\MyProject>
```

Un fichier nommé package.json a été créé sur votre ordinateur.

#### Remarque :

- on peut initialiser un dépôt Git en exécutant : git init depuis le dossier backend.
- créer un fichier .gitignore contenant la ligne node\_modules afin de ne pas envoyer ce dossier, qui deviendra volumineux, vers le dépôt distant.

#### EXPRESS/EJS

- Express.js (Ou tout simplement Express) est un Web Application Framework de NodeJS. Il fournit un ensemble de fonctionnalités puissantes aux applications web et mobiles.
- EJS est l'abréviation de "Embedded JavaScript templating", qui est une bibliothèque, utilisée pour analyser les fichiers ejs et créer du HTML pour retourner au client (navigateur).

Commande : npm install express ejs

```
C:\Windows\system32\cmd.exe

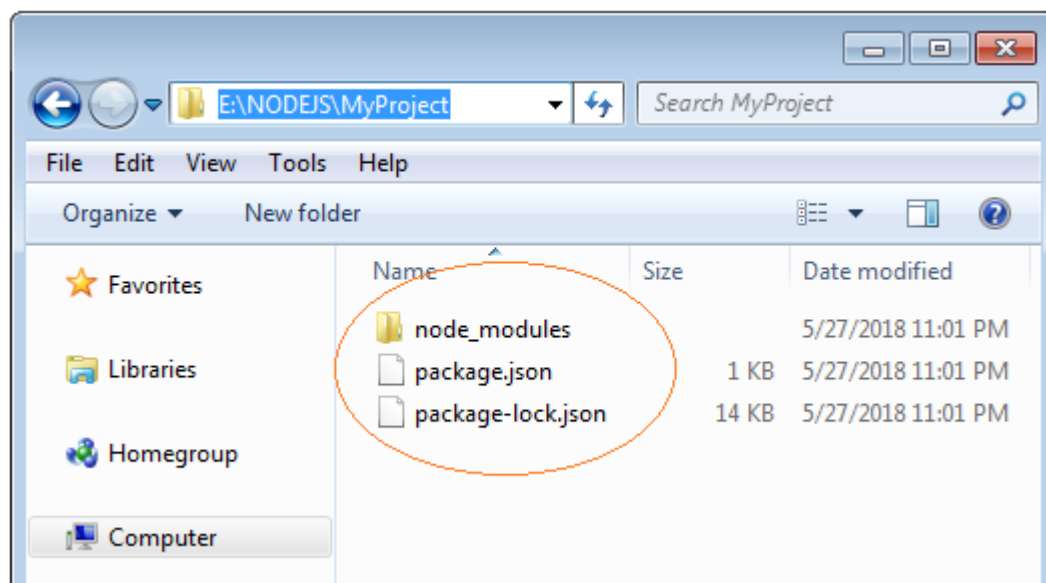
E:\NODEJS\MyProject>npm install express ejs
npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN myproject@1.0.0 No description
npm WARN myproject@1.0.0 No repository field.

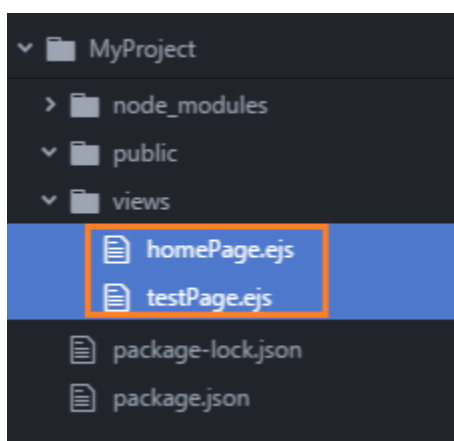
+ ejs@2.6.1
+ express@4.16.3
added 51 packages in 5.641s

E:\NODEJS\MyProject>_
```

Après la fin de l'installation,



## STRUCTURE DU PROJET



Sur le projet, On aura deux sous dossiers public & views :

- public : Est le dossier comprenant tous les fichiers que des utilisateurs peuvent accéder, par exemple image, video,...
- views : Votre site web contient plusieurs pages, par exemple, page d'accueil, page de connexion, ... Ce dossier est l'endroit contenant toutes vos pages.

Dans le dossier views, nous avons deux fichiers : homePage.ejs et testPage.ejs

```
//homePage.ejs
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Home</title>
  </head>
  <body>

    <h1>This is Home Page</h1>

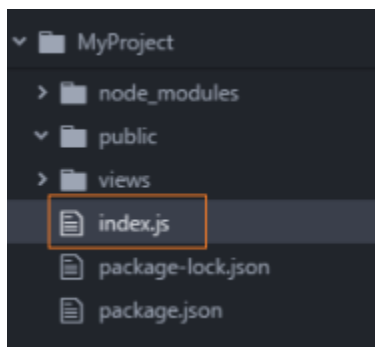
  </body>
</html>
```

```
//testPage.ejs
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Test</title>
  </head>
  <body>

    <h1>This is Test Page</h1>

  </body>
</html>
```

Ensuite, créez un fichier baptisé index.js :



```

1  var express = require("express");
2
3  var app = express();
4
5  app.use(express.static("public"));
6
7  app.set("view engine", "ejs");
8  app.set("views", "./views");
9
10 app.listen(3000);
11
12 app.get("/", function(request, response) {
13
14     response.render("homePage");
15 });
16
17 app.get("/test", function(request, response) {
18
19     response.render("testPage");
20 });

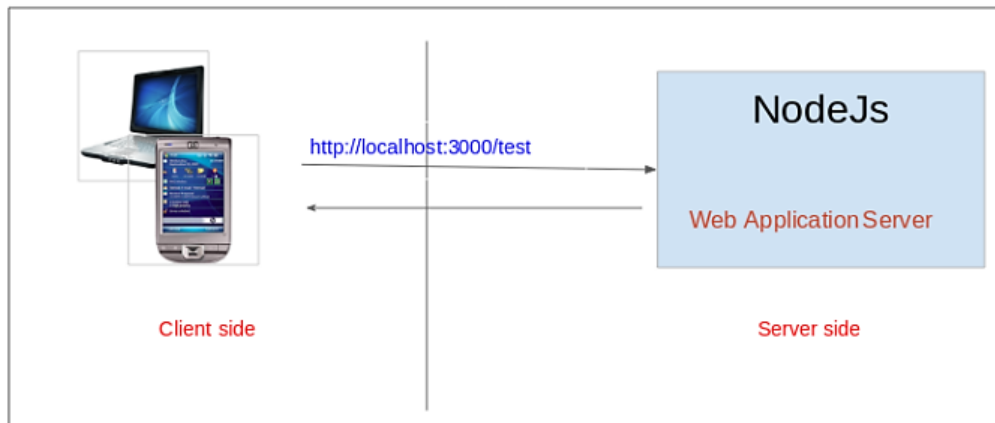
```

Code	Description
var express = require("express");	<b>ExpressJS</b> est un <b>Web Application Framework</b> , cette ligne de code indique que vous voulez l'utiliser.
var app = express();	Créer un objet <b>Express</b> .
app.use(express.static("public"));	Cette ligne de code indique <b>Application Server</b> que vous voulez utiliser le dossier <b>public</b> pour stocker des données statiques, l'utilisateur peut accéder aux fichiers dans ce dossier.
app.set("view engine", "ejs");	Cette ligne de code indique <b>Application Server</b> que vous voulez utiliser la bibliothèque <b>EJS</b> . C'est une machine pour gérer votre page. Le <b>EJS</b> créera du <b>HTML</b> pour retourner au navigateur de l'utilisateur.
app.set("views", "./views");	Cette ligne de code indique <b>Application Server</b> le chemin d'accès au dossier contenant vos pages.
app.listen(3000);	Votre application écoutera sur le port 3000 quand elle est déployée.
app.get("/test", function(req, res) { ... });	Définir le chemin d'accès à une page

EXECUTION :

```
C:\Windows\system32\cmd.exe - node index.js
E:\NODEJS\MyProject>node index.js
```

Cette commande démarrera le Web Application Server, et déploiera votre application sur ce Web Server. En ce moment, il est prêt à servir la demande envoyée par le client.



**Remarque** : Ne fermez pas de la fenêtre CMD, parce que votre Application Server est en cours d'exécution.

Ouvrez le navigateur et accédez au lien :

<http://localhost:3000/>

<http://localhost:3000/test>

