

## O que é Node.js?

- O **Node.js** é uma runtime de JavaScript;
- Ou seja, **uma biblioteca utilizada por um compilador** durante a execução do programa;
- Que está construída na **V8 engine** (escrita em C++) da Google;
- Possibilitando criar softwares em JS no lado do servidor;
- Temos então um código JS rodando em C++ para garantir **alta performance**;



O Que é npm?

O gerenciador de pacotes Node (npm) é um dos maiores repositórios de software do mundo. Ele vem acompanhado do node.js, um ambiente de servidor de código aberto.

## O que é npm?

- O **npm** é um gerenciador de pacotes do Node;
- Vamos poder utilizar **bibliotecas de terceiros**, baixando elas pelo npm;
- E também **executar determinados scripts** no nosso programa;
- Dificilmente um software em Node.js não utiliza o npm;
- Os módulos externos ficam numa pasta chamada **node\_modules**;
- Ela deve ser descartável, ou seja, a cada instalação do projeto baixamos todos os pacotes novamente;



O Que é package.json?

Todo projeto npm contém um package.json, um arquivo localizado no diretório raiz. Ele contém os metadados de projetos ou pacotes npm, como versões de pacotes e colaboradores.

O arquivo package.json simplifica a identificação, gerenciamento e instalação de um pacote. É por isso que é essencial incluir um arquivo package.json antes de publicar projetos no registro npm.

Agora vamos usar o npm para começar os nossos projetos, para começar limpe os projetos Nodes anteriores, para não ficarmos com a máquina cheia de pastas que não vamos usar mais.

Agora crie uma pasta com a seguinte sintaxe "seunome\_turma" exemplo "Evandro\_355".

Abra o **cmd** e vamos criar nosso primeiro arquivo dentro da nossa pasta.

```
npm init -g
```

Abra a pasta com VSCODE

Instalando o Express

O Express é um framework para aplicativo da web do Node.js mínimo e flexível que fornece um conjunto robusto de recursos para aplicativos web e móvel.

Instalando "express"

No cmd

```
npm install express --save
```

Vamos fazer a nossa primeira conexão.

Crie o arquivo **index.js**

```
const express = require("express");
app = express();

app.listen(80, () =>{
  console.log("SERVIDOR RODANDO COM SUCESSO")
})
```

Criando as primeiras rotas para teste.

```
const express = require("express");
app = express();

app.get("/", (req, res) =>{
  res.send("AQUI É INDEX")
})
app.get("/usuario", (req, res) =>{
  res.send("AQUI É ARQUIVO USUÁRIO")
})

app.listen(80, () =>{
  console.log("SERVIDOR RODANDO COM SUCESSO")
})
```

Usando parâmetros aqui vão ser obrigatórios

```
const express = require("express");
app = express();

app.get("/", (req, res) => {
  res.send("AQUI É INDEX")
})
app.get("/usuario/:nome", (req, res) => {
  //REQ --> DADOS ENVIADOS PELO USUÁRIO
  //RES --> RESPOSTA QUE VAI SER ENVIADA PARA O USUÁRIO
  let nome = req.params.nome
  res.send("<h3>Ola como vai ? " + nome + "<h3>")
})

app.listen(80, () => {
  console.log("SERVIDOR RODANDO COM SUCESSO")
})
```

Usando parâmetros aqui não vão ser obrigatórios

```
const express = require("express");
app = express();

app.get("/", (req, res) => {
  res.send("AQUI É INDEX")
})
app.get("/usuario/:nome", (req, res) => {
  //REQ --> DADOS ENVIADOS PELO USUÁRIO
  //RES --> RESPOSTA QUE VAI SER ENVIADA PARA O USUÁRIO
  let nome = req.params.nome
  res.send("<h3>Ola como vai ? " + nome + "<h3>")
})
app.get("/blog/:artigo?", (req, res) => {
  //REQ --> DADOS ENVIADOS PELO USUÁRIO
  //RES --> RESPOSTA QUE VAI SER ENVIADA PARA O USUÁRIO
  let artigo = req.params.artigo;
  if(artigo){
    res.send("<h2>Artigo: " + artigo + "</h2>")
  }else{
    res.send("<h3>Bem vindo ao blog <h3>")
  }
})

app.listen(80, () => {
  console.log("SERVIDOR RODANDO COM SUCESSO")
})
```

Aqui mostra a importância do package.json . Deletar a pasta "node\_modules" , fazer teste  
Testar "npm update"

Usando o EJS

O EJS é uma engine de visualização, com ele conseguimos de uma maneira fácil e simples transportar dados do back-end para o front-end, basicamente conseguimos utilizar códigos em javascript no html de nossas páginas.

Crie uma nova pasta "evandro\_express\_ejs\_255"

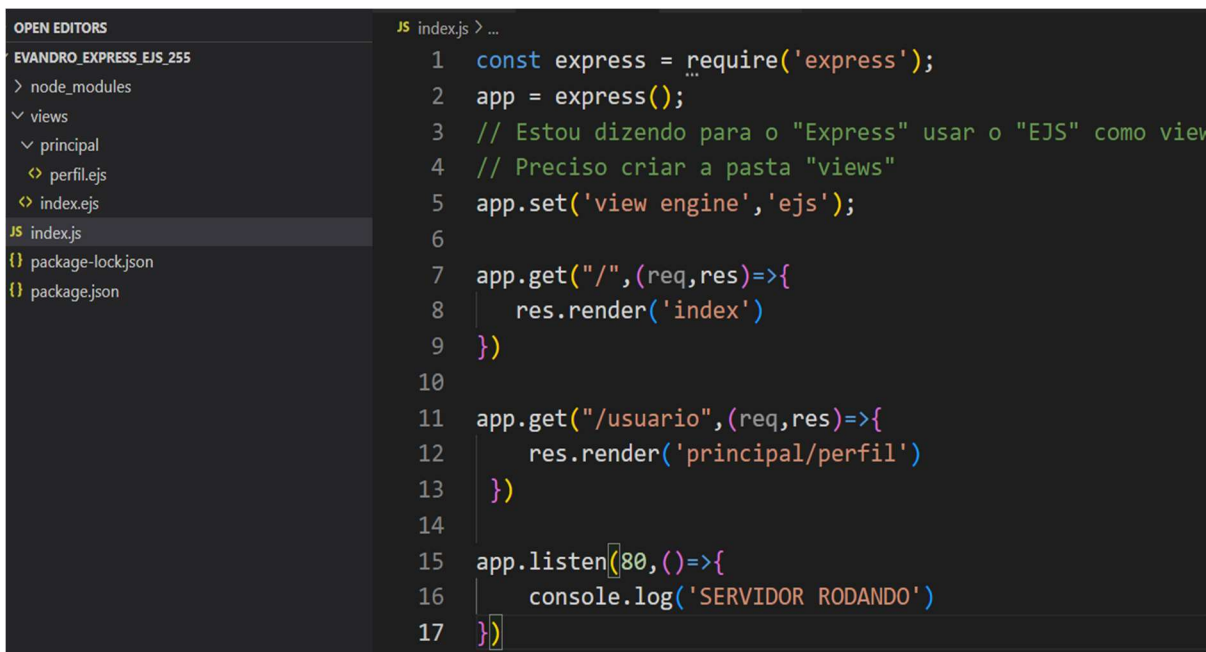
```
npm init
```

```
npm install express --save
```

```
npm install ejs --save
```

Vamos instalar mais um repositório do node o “nodemon” vai servir para não precisarmos mais, ficar desligando e reiniciando nosso servidor NODEJS

```
npm install nodemon -g
```



```
OPEN EDITORS
EVANDRO_EXPRESS_EJS_255
> node_modules
  views
    principal
      perfil.ejs
      index.ejs
  JS index.js
  {} package-lock.json
  {} package.json

JS index.js > ...
1  const express = require('express');
2  app = express();
3  // Estou dizendo para o "Express" usar o "EJS" como view
4  // Preciso criar a pasta "views"
5  app.set('view engine', 'ejs');
6
7  app.get("/", (req, res) => {
8    res.render('index')
9  })
10
11 app.get("/usuario", (req, res) => {
12   res.render('principal/perfil')
13 })
14
15 app.listen(80, () => {
16   console.log('SERVIDOR RODANDO')
17 })
```

## Exercicio

Crie a pasta `exercicio_express_ejs_01`.

Faça todas as inicializações necessárias em `author` → coloque seu nome ou dupla

Vais existir três arquivos.

Na pasta “views”

`index.ejs`

```

<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>INDEX</title>
  </head>
  <body>
    <div align="center">
      <button id="mostra_maior">Mostrar Maior</button>
      <br /><br />
      <button id="inverte_palavra">Inverte Palavra</button>
    </div>
  </body>
  <script>
    <!-- Aqui JavaScript -->
  </script>
</html>

```

Na pasta “**views/serviços**”

**mostra\_maior.ejs**

```

<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mostra Maior</title>
  </head>
  <body>
    <h3>Maior número</h3>
    <p>Digite primeiro número: <input type="text" id= "num1" size="5"></p>
    <p>Digite segundo número: <input type="text" id= "num2" size="5"></p>
    <p>Digite terceiro número: <input type="text" id= "num3" size="5"></p>
    <p>Maior Número: <input type="text" id= "resultado" disabled></p>
    <p><button id="processar">Processar</button>
    <button id="voltar">Voltar</button> </p>
  </body>
  <script>
    <!-- Aqui JavaScript -->
  </script>
</html>

```

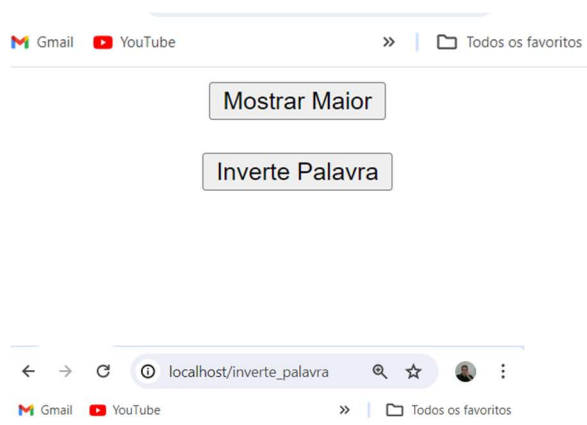
“**inverte\_palavra.ejs**”

```

<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inverter</title>
  </head>
  <body>
    <h3>Inverte Palavra</h3>
    <p>Digite Frase: <input type="text" id= "texto" ></p>
    <p>Invertida: <input type="text" id= "resultado" disabled></p>
    <p><button id="processar">Processar</button>
    <button id="voltar">Voltar</button> </p>
  </body>
  <script>
    <!-- Aqui JavaScript -->
  </script>
</html>

```

Agora crie o arquivo index.js com as devidas configurações, usando “express”, “ejs” para que rode como as imagens abaixo. **PORTA 80**



## Inverte Palavra

Digite Frase:

Invertida:

Passando variáveis do **nodeJs** para o arquivo **ejs**.

Vamos abri nossa pasta/projeto anterior no meus caso “evandro\_express\_ejs\_355”

“index.js”

```
const express = require('express');
app = express();

app.set('view engine','ejs');

app.get("/",(req,res)=>{
  res.render('index')
})

app.get("/usuario",(req,res)=>{
  var id = "01"
  var nome ="Evandro Jose Vieira"
  var linguagem = "JavaScript"
  res.render('principal/perfil',{
    id: id,
    nome : nome,
    linguagem: linguagem
  })
})

app.listen(80,()=>{
  console.log('SERVIDOR RODANDO')
})
```



## “perfil.ejs”

```
<html lang="pt-br">
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Usuário</title>
</head>
<body>
  <h2>Aqui o arquivo do perfil do usuário</h2>
  <p>Id: <%= id %></p>
  <p>Nome: <%= nome %></p>
  <p>Linguagem: <%= linguagem %></p>
</body>
</html>
```

Usando estrutura de condição dentro do “ejs”

```
const express = require('express');
app = express();

app.set('view engine', 'ejs');

app.get("/", (req, res) => {
  res.render('index')
})

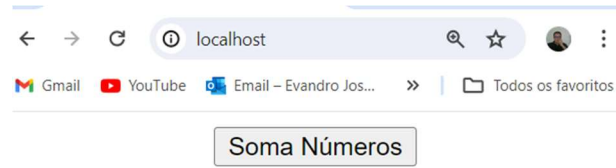
app.get("/usuario", (req, res) => {
  var id = "01";
  var nome = "Evandro Jose Vieira";
  var linguagem = "JavaScript";
  var exibirDados = false;
  res.render('principal/perfil', {
    id: id,
    nome: nome,
    linguagem: linguagem,
    dados: exibirDados,
  })
})

app.listen(80, () => {
  console.log('SERVIDOR RODANDO')
})
```

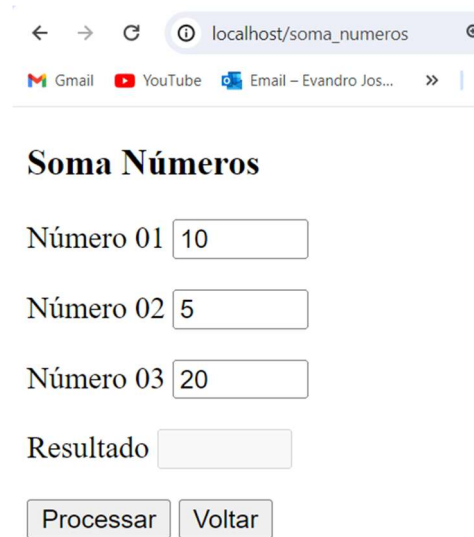
```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Usuário</title>
</head>
<body>
  <h2>Aqui o arquivo do perfil do usuário</h2>
  <% if(dados){ %>
  <p>Id: <%= id %></p>
  <p>Nome: <%= nome %></p>
  <p>Linguagem: <%= linguagem %></p>
  <% } else { %>
  <p>Esta retornando Falso</p>
  <% } %>
</body>
</html>
```

## EXERCICIO

Crie a pasta exercicio\_”express\_ejs\_02” crie o index para ficar como a imagem abaixo



Ao clicar no botão deverá ser aberta a página “soma\_números”, os valores já serão preenchidos automaticamente, ao clicar em “Processar” será feita a soma dos números, outros números claro também poderão ser digitados.



Você deve configurar o arquivo index.js com as devidas configurações.