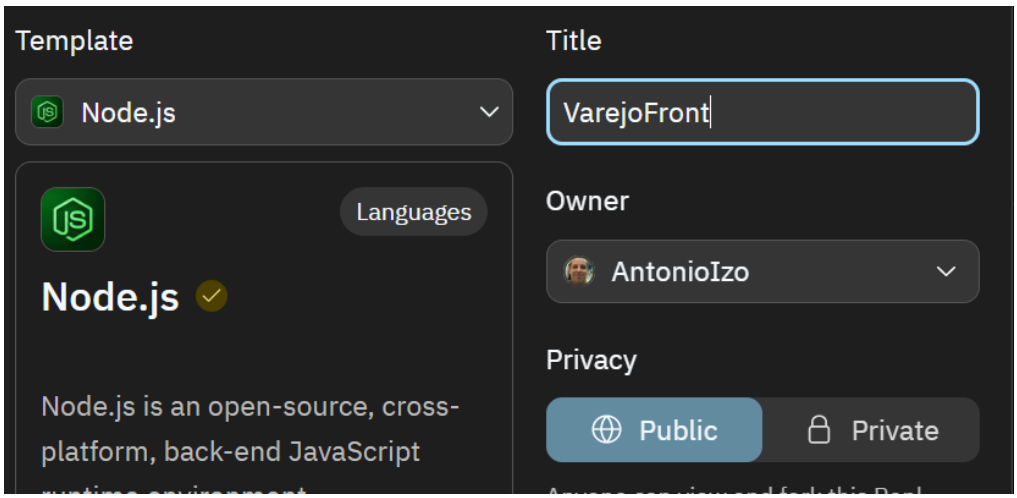
Autenticação com JWT

JSON WEB TOKEN

Criando o FrontEnd



Vamos instalar as bibliotecas que usaremos.

```
npm install express
```

```
~/VarejoFront$ npm install express
```

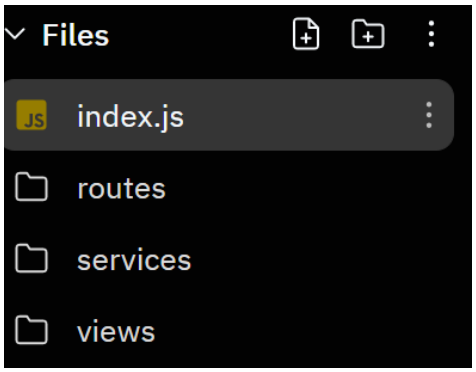
```
npm install express-handlebars
```

```
~/VarejoFront$ npm install express-handlebars
```

```
npm install axios
```

```
~/VarejoFront$ npm install axios
```

Vamos criar as seguintes pastas para nossa aplicação de varejo.



Vamos codificar cada uma delas.

Na pasta views, vamos criar a pasta layouts e dentro dela o arquivo main.handlebars. O código do main.handlebars é mostrado abaixo.

```
<html lang="pt-br">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
     {{! BOOSTRAP 5 }}
     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJO
Z" crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
ENjdO4Dr2bkBIFxQpeoTz1HIcje39Wm4jDKdf19U8gI4ddQ3GYNS7NTKfAdVQSZe"
crossorigin="anonymous"></script>
{{!-- ARQUIVOS LOCAL --}}
    <link rel="stylesheet" href="css/styles.css" />
    <title>Sistema de Lista de Varejo</title>
  </head>
  <body id="body-mobile">
    <nav class="navbar text-bg-dark">
      <div class="container">
        <span class="navbar-brand text-white">Sistema de Varejo virtual</span>
        <ul class="navbar nav">
          <li class="nav-item">
```

```
        </li>
      </ul>
    </div>
  </nav>
  <div class="container">
    {{{body}}}
  </div>
 </body>
</html>
```

Dentro da pasta views vamos criar uma pasta usuários e um arquivo chamado login.handlebars.

Vamos criar um formulário simples com login e senha de acesso para testar se o usuário consegue logar.

```
<h1 class="text-center my-3">Login</h1>
<form action="/login" method="post">
 <div class="row w-50 d-block m-auto g-3">
   <div class="col-12">
     <label class="form-label" for="email">email:</label>
     <input
       class="form-control"
       type="text"
       name="email"
       id="email"
       placeholder="Digite o seu email cadastrado"
     />
   </div>
   <div class="col-12">
     <label class="form-label" for="senha">Senha:</label>
     <input
       class="form-control"
       type="password"
       name="senha"
       id="senha"
```

```
        placeholder="Digite a sua senha"
      />
    </div>
    <div>
      <input class="btn btn-primary my-3" type="submit" value="Acessar" />
    </div>
  </div>
</form>
```

Crie dentro da pasta views, um arquivo chamado mensagem.handlebars com o código abaixo:

```
<h4>{{mensagem}}</h4>
```

Crie na raiz da pasta views, um arquivo chamado logado.handlebars com o código abaixo.

Se nosso usuário for logado, ele será enviado para a página.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.13.0/css/all.min.css">
  <title>Seja bem vindo ao sistema de Compras Virtual</title>
</head>
<body>
  <h1>Seja bem vindo!</h1><br>

  <a href="/">voltar</a>

</body>
</html>
```

Vamos criar o arquivo **services.js** dentro da pasta services e o arquivo **routes.js** dentro da pasta routes.

Agora no arquivo index.js, vamos criar a estrutura que utilizaremos.

```
const express = require(“express”);
const app = express();
const hand = require(“express-handlebars”);
```

```
const Services = require(“./services/services”);
const routes = require(“./routes/routes”);

app.engine(“handlebars”, hand.engine());
app.set(“view engine”, “handlebars”);

app.use(express.urlencoded({extended: true,}));
app.use(express.json());

app.use(“/”, routes);

app.listen(3000);
```

Vamos agora criar nosso arquivo de **services.js**.

Nosso arquivo services.js terá por enquanto a codificação abaixo.

```
const axios = require("axios");

module.exports = class Services{
//VERIFICAR USUÁRIO
  static async UsuarioLogin(req,res){
    let valores = req.body;
    const options = {
      url: ‘COLOCAR O ENDEREÇO DO SUA API/login',
      method: 'POST',
      data: valores
    };
    axios(options).then((usuario) => {
    if(usuario != undefined){
      return res.render('logado');
    }
    })
  }
}
```

Criamos uma função para verificar se os dados passados pelo formulário de login e repassamos para a url da API, depois renderizamos para a página logado o usuário.

ATENÇÃO: É importante que você coloque o endereço da sua API na variável url.

Vamos agora para nosso arquivo de **routes.js**.

```
const express = require("express");
const Services = require("../services/services");
const router = express.Router();

router.get("/",(req, res) =>{
    res.send("Seja bem Vindo ao nosso sistema de Varejo Virtual.");
});

router.post("/login",Services.UsuarioLogin);
router.get("/login",(req, res) =>{
  res.render("usuarios/login");
})
router.post("/login", Services.UsuarioLogin)

module.exports=router;
```

Repare que tanto o arquivo services,js e routes.js são bem parecidos com os arquivos que já fizemos durante nosso módulo.

Vamos criar uma view dentro da pasta usuários chamada Cadastrar.handlebars.

```
<h1 class="text-center my-3">Sistema de Cadastro de Clientes</h1>
<form action="/add_usuario" method="post">
  <div class="row w-50 d-block m-auto g-3">
    <div class="col-12">
      <label class="form-label" for="nome">Nome:</label>
      <input
        class="form-control"
        type="text"
        name="nome"
        id="nome"
        placeholder="Digite o seu Nome"
      />
    </div>
    <div class="col-12">
      <label class="form-label" for="email">Email:</label>
      <input
        class="form-control"
        type="text"
        name="email"
        id="email"
        placeholder="Digite o seu Email"
      />
    </div>
    <div class="col-12">
      <label class="form-label" for="senha">Senha:</label>
      <input
        class="form-control"
```

```
        type="password"
        name="senha"
        id="senha"
        placeholder="Digite a sua senha"
      />
    </div>
    <div>
      <input class="btn btn-primary my-3" type="submit" value="Cadastrar Usuário" />
    </div>
  </div>
</form>
```

Vamos criar no arquivo services.js nossa função create.

```
//Create usuário
  static async UsuarioCreate(req,res){
    let valores = req.body;
    const options = {
      url: 'https://apivarejo.antonioizo.repl.co/add_usuario',
      method: 'POST',
      data: valores
    };
    axios(options);
    const mensagem = "Cadastro realizado com sucesso!";
    res.render("mensagem",{mensagem});
  }
```

Agora vamos criar nossas rotas no routes.js.

```
router.get("/usuarios/Cadastrar",(req, res) =>{
  res.render("usuarios/Cadastrar");
})

router.post("/usuarios/Cadastrar",Services.UsuarioCreate);
```

Uma rota para renderizar para nosso formulário e outra para chamar a função de cadastro.

Agora dentro da view login.handlebars, vamos incluir um link caso o usuário não seja cadastrado ainda para realizar o cadastro.

```
<div class="col-12">
   <a href="/usuarios/Cadastrar">Caso não seja cadastrado clique aqui<i class="bi bi-pencil-square"></i></a></div>
```

Nossa aplicação está ficando bacana demais. Já estamos logando, cadastrando e nossas rotas estão protegidas com JWT.

Um abraço e até a próxima aula.