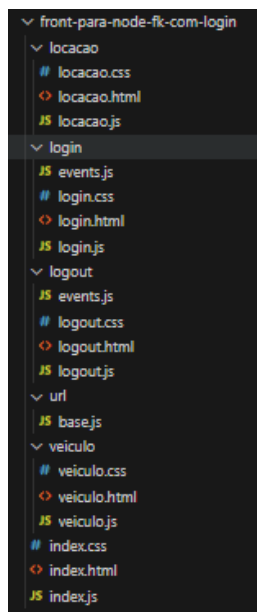
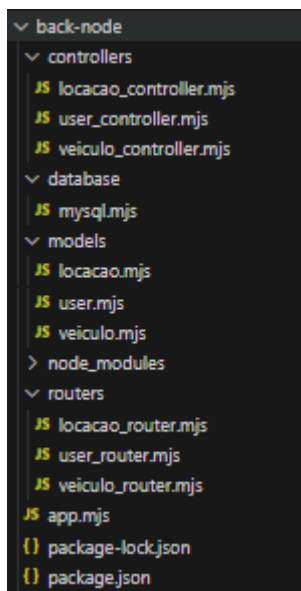


ATIVIDADE 07

A atividade 07 é uma junção de tela de login e foreign key no banco de dados, neste caso, foi criado o usuário pelo thunder cliente, somente com o email samyra@gmail.com e uma senha de 4 dígitos, como demonstrarão as capturas de tela a seguir.

Anteriormente, estava fazendo de uma loja, mas, diante de problemas com conexão, foi somente adicionado um atributo a mais no já realizado em sala. A chave estrangeira está presente na parte de locações dos carros, onde puxa os dados do veículo que será alocado pelo cliente.

Será enviado prints somente das partes de login e foreign key, pois as demais funções permanecem iguais aos exercícios anteriores.



pastas e arquivos usados



A locação controller define um controlador de locações que permite criar, buscar, listar, editar e remover registros de locações no banco de dados, utilizando um modelo Locacao.

```

1 import User from "../models/user.mjs";
2 import bcrypt from "bcrypt";
3
4 const saltRounds = 10;
5
6 const UserController = Object.create(Object.prototype);
7
8 UserController.new = async (req, res) => {
9   const hash = bcrypt.hashSync(req.body.senha, saltRounds);
10   const created = await User.create({
11     email: req.body.email,
12     senha: hash,
13   });
14   res.json({ email: created.email });
15 };
16
17 UserController.login = async (req, res) => {
18   const v = await User.findOne({
19     where: { email: req.body.email },
20   });
21   if (!v) {
22     if (await bcrypt.compare(req.body.senha, v.senha)) {
23       req.session.logged = true;
24       req.session.email = v.email;
25       console.log("Usuário após login:", req.session);
26       res.json({ logged: req.session.logged });
27     } else {
28       console.log("Senha incorreta");
29       res.json({ logged: false });
30     }
31   } else {
32     console.log("Usuário não encontrado");
33     res.json({ logged: false });
34   }
35 };
36
37 UserController.logout = async (req, res) => {
38   req.session.logged = false;
39   req.session.email = null;
40   res.json({ logged: req.session.logged });
41 };
42
43 UserController.logged = async (req, res) => {
44   console.log("Verificando se usuário está logado:", req.session);
45   res.json({ logged: req.session.logged ? true : false });
46 };
47
48 export default UserController;
49

```

O user controller gerencia a criação, autenticação, e controle de sessão de usuários. Ele permite criar novos usuários com senhas criptografadas, realizar login validando as credenciais, encerrar sessões de login, e verificar o status de login do usuário.

```

1 import Veiculo from "../models/veiculo.mjs";
2
3 const VeiculoController = {
4   new: async (req, res) => {
5     const created = await Veiculo.create({
6       fabricante: req.body.fabricante,
7       modelo: req.body.modelo,
8       cor: req.body.cor,
9       listagem: req.body.listagem,
10     });
11     res.send(created);
12   },
13   one: async (req, res) => {
14     const v = await Veiculo.findOne({
15       where: { id: req.params.id },
16     });
17     res.json(v);
18   },
19   all: async (req, res) => {
20     res.json(await Veiculo.findAll());
21   },
22   edit: async (req, res) => {
23     const v = await Veiculo.findOne({
24       where: { id: req.body.id },
25     });
26     v.fabricante = req.body.fabricante;
27     v.modelo = req.body.modelo;
28     v.cor = req.body.cor;
29     v.listagem = req.body.listagem;
30     await v.save();
31     res.json(v);
32   },
33   remove: async (req, res) => {
34     const v = await Veiculo.findOne({
35       where: { id: req.body.id },
36     });
37     await v.destroy();
38     res.json(v);
39   },
40 };
41
42 export default VeiculoController;
43

```

Veículo controller gerencia veículos, permitindo criar, buscar, listar, editar e remover registros de veículos no banco de dados. As operações são baseadas em dados fornecidos nas requisições e os resultados são retornados em formato JSON.

```

1 import sequelize from "../database/mysql.mjs";
2
3 import { DataTypes } from "sequelize";
4
5 import Veiculo from "../veiculo.mjs";
6
7 const Locacao = sequelize.define("Locacao", {
8   cliente: DataTypes.STRING,
9   inicio: DataTypes.DATEONLY,
10   fim: DataTypes.DATEONLY,
11 });
12
13 Locacao.belongsTo(Veiculo);
14
15 export default Locacao;
16

```

Locacao define um modelo de locação que inclui os campos cliente, início e fim, e está relacionado a um veículo através de uma associação.

```

1 import sequelize from "../database/mysql.mjs";
2 import { DataTypes } from "sequelize";
3
4 const User = sequelize.define("User", {
5   email: DataTypes.STRING,
6   senha: DataTypes.STRING,
7 });
8
9 export default User;

```

User define um modelo de usuário com os campos e-mail e senha, utilizando o Sequelize para mapeamento do banco de dados MySQL.

```

1 import sequelize from "../database/mysql.mjs";
2 import { DataTypes } from "sequelize";
3
4 const Veiculo = sequelize.define("Veiculo", {
5   fabricante: DataTypes.STRING,
6   modelo: DataTypes.STRING,
7   cor: DataTypes.STRING,
8   listagem: DataTypes.STRING,
9 });
10
11
12 timestamps: false; // Desativa createdAt e updatedAt
13
14 export default Veiculo;

```

Veiculo define um modelo de veículo com os campos fabricante, modelo, cor e listagem. O modelo tem timestamps desativados, ou seja, não inclui os campos `createdAt` e `updatedAt`.

```

1 import { Router } from "express";
2
3 import LocacaoController from "../controllers/locacao_controller.mjs";
4
5 const locacaoRouter = Router();
6
7 locacaoRouter.get("/", LocacaoController.all);
8
9 locacaoRouter.get("/:id", LocacaoController.one);
10
11 locacaoRouter.post("/", LocacaoController.new);
12
13 locacaoRouter.put("/", LocacaoController.edit);
14
15 locacaoRouter.delete("/", LocacaoController.remove);
16
17 export default locacaoRouter;

```

locação Router define as rotas para gerenciar locações, utilizando o LocacaoController para lidar com as operações de listar todas as locações, buscar uma locação por ID, criar uma nova locação, atualizar uma locação existente e remover uma locação.

```

1 import { Router } from "express";
2
3 import UserController from "../controllers/user_controller.mjs";
4
5 const userRouter = Router();
6
7 userRouter.post("/", UserController.new);
8
9 userRouter.post("/login", UserController.login);
10
11 userRouter.get("/logged", UserController.logged);
12
13 userRouter.get("/logout", UserController.logout);
14
15 export default userRouter;

```

userRouter define as rotas para gerenciar usuários, utilizando o UserController para lidar com as operações de criar um novo usuário, realizar login, verificar o status de login e encerrar a sessão.

```

1 import express from "express";
2 import cors from "cors";
3 import session from "express-session";
4 import sequelize from "../database/mysql.mjs";
5 import CSS from "connect-session-sequelize";
6
7 import userRouter from "../routers/user_router.mjs";
8 import veiculoRouter from "../routers/veiculo_router.mjs";
9 import locacaoRouter from "../routers/locacao_router.mjs";
10
11 const app = express();
12 const port = 3000;
13
14 const SequelizeStore = CSS(session.Store);
15
16 app.use(
17   session({
18     secret: "#7UIERU933E00LERI##32734586",
19     store: new SequelizeStore({
20       db: sequelize,
21     }),
22   })
23 );
24
25 app.use(
26   cors({
27     origin: ["http://localhost:5500", "http://127.0.0.1:5500"],
28     credentials: true,
29   })
30 );
31
32 app.use(express.json());
33 app.use(express.urlencoded());
34
35 app.use("/user", userRouter);
36 app.use("/veiculos", veiculoRouter);
37 app.use("/locacoes", locacaoRouter);
38
39 app.listen(port, () => {
40   console.log(`Example app listening on port ${port}`);
41 });

```

O app mjs configura o servidor Express para gerenciar rotas e sessões, utilizando sequelize para persistência de sessões com connect-session-sequelize. Ele define as rotas para usuários, veículos e locações, e configura CORS para permitir requisições de certos domínios. O servidor escuta na porta 3000.

Agora será mostrado o front-end, não será enviado o arquivo de css, pois é somente a “estética das telas”.

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Locações</title>
8   <link rel="stylesheet" href="locacao.css">
9   <link rel="icon" href="favicon.ico" type="image/x-icon">
10 </head>
11 <body>
12   <header>
13     <h1>Locações</h1>
14     <a href="index.html" id="Inicio">Inicio</a>
15     <a href="login/logout.html" id="Logout">Logout</a><br><br>
16   </header>
17   <main>
18     <div id="tag">
19       <button id="Novo">Novo</button>
20     <table border="1">
21       <thead>
22         <tr>
23           <th>#</th>
24           <th>CLIENTE</th>
25           <th>VEICULO</th>
26           <th>INICIO</th>
27           <th>FIM</th>
28           <th>AÇÕES</th>
29         </tr>
30       </thead>
31       <tbody>
32         <tr>
33           <td></td>
34           <td></td>
35           <td></td>
36           <td></td>
37           <td></td>
38         </tr>
39       </tbody>
40     </table>
41   </main>
42   <div id="modal">
43     <form action="">
44       <input type="hidden" name="id" id="id"><br>
45       <label for="cliente">Cliente</label><br>
46       <input type="text" name="cliente" id="cliente"><br>
47       <label for="veiculo">Veiculo</label><br>
48       <select name="veiculo" id="veiculo"></select><br>
49       <label for="inicio">Inicio</label><br>
50       <input type="date" name="inicio" id="inicio"><br>
51       <label for="fim">Fim</label><br>
52       <input type="date" name="fim" id="fim"><br>
53       <button id="Salvar">Salvar</button>
54     </form>
55   </div>
56   <script type="module" src="locacao.js"></script>
57 </html>

```

locação html cria uma página para gerenciar locações, incluindo uma tabela para listar locações e um formulário em um modal para adicionar ou editar locações. A página também inclui links para voltar à página inicial e para logout.

locação js gerencia a exibição e interação com dados de locações, incluindo criação, edição, exclusão e visualização de locações e veículos. Ele também controla a visibilidade do modal e a validação de login. (próxima página)

```

1 import { checkLogin } from "../login/login.js";
2 import { urlBaseAPI, urlBaseFront } from "../url/base.js";
3
4 let dados = [];
5
6 function desenhaTabela() {
7     const tbody = document.querySelector("tbody");
8     tbody.innerHTML = "";
9     for (let i = 0; i < dados.length; i++) {
10         const tr = document.createElement("tr");
11         const btEx = document.createElement("button");
12         const btEd = document.createElement("button");
13
14         const td1 = document.createElement("td");
15         const td2 = document.createElement("td");
16         const td3 = document.createElement("td");
17         const td4 = document.createElement("td");
18         const td5 = document.createElement("td");
19         const td6 = document.createElement("td");
20
21         btEx.innerText = "-";
22         btEx.setAttribute("data-id", dados[i].id);
23         btEx.addEventListener("click", (e) => {
24             const id = e.target.getAttribute("data-id");
25             alternaModal();
26             enviaDadosParaDelecao(id);
27         });
28
29         btEd.innerText = "+";
30         btEd.setAttribute("data-index", i);
31         btEd.addEventListener("click", (e) => {
32             const index = e.target.getAttribute("data-index");
33             alternaModal();
34             preencheFormParaEdicao(index);
35         });
36
37         td1.innerText = dados[i].id;
38         td2.innerText = dados[i].cliente;
39         td3.innerText = dados[i].veiculo.fabricante +
40             " + " +
41             dados[i].veiculo.modelo +
42             " + " +
43             dados[i].veiculo.cor +
44             " + " +
45             dados[i].veiculo.litragem;
46         td4.innerText = dados[i].inicio;
47         td5.innerText = dados[i].fim;
48         td6.appendChild(btEx, btEd);
49
50         tr.appendChild(td1, td2, td3, td4, td5, td6);
51         tbody.appendChild(tr);
52     }
53 }
54
55 function carregaVeiculos() {
56     const opcoes = {
57         method: "get",
58         credentials: "include",
59     };
60     fetch(`${urlBaseAPI}/veiculos`, opcoes)
61         .then((res) => {
62             //console.log(res);
63             return res.json();
64         })
65         .then((json) => {
66             //console.log(json);
67             //alert(json);
68             const select = document.getElementById("veiculoId");
69             select.innerHTML = "-";
70             for (let i = 0; i < json.length; i++) {
71                 const option = document.createElement("option");
72                 option.innerText =
73                     json[i].fabricante +
74                     " + " +
75                     json[i].modelo +
76                     " + " +
77                     json[i].cor +
78                     " + " +
79                     json[i].litragem;
80                 option.value = json[i].id;
81                 select.appendChild(option);
82             }
83         });
84 }
85
86 function carregaDados() {
87     const opcoes = {
88         method: "get",
89         credentials: "include",
90     };
91     fetch(`${urlBaseAPI}/locaes`, opcoes)
92         .then((res) => {
93             //console.log(res);
94             return res.json();
95         })
96         .then((json) => {
97             //console.log(json);
98             //alert(json);
99             dados = json;
100             desenhaTabela();
101         });
102 }
103
104 function enviaDadosParaCadastro() {
105     const dados = new FormData(document.querySelector("form"));
106     const opcoes = {
107         method: "post",
108         credentials: "include",
109         body: new URLSearchParams(dados),
110     };
111     fetch(`${urlBaseAPI}/locaes`, opcoes)
112         .then((res) => {
113             //console.log(res);
114             return res.json();
115         })
116         .then((json) => {
117             //console.log(json);
118             alert("Local de cadastro!");
119             carregaDados();
120         });
121     alternaModal();
122 }
123
124 function enviaDadosParaDelecao(id) {
125     const dados = new FormData();
126     dados.append("id", id);
127     const opcoes = {
128         method: "delete",
129         credentials: "include",
130         body: new URLSearchParams(dados),
131     };
132     fetch(`${urlBaseAPI}/locaes`, opcoes)
133         .then((res) => {
134             //console.log(res);
135             return res.json();
136         })
137         .then((json) => {
138             //console.log(json);
139             alert("Local de deleção!");
140             carregaDados();
141         });
142     alternaModal();
143 }
144
145 function preencheFormParaEdicao(index) {
146     document.querySelector("#id").value = dados[index].id;
147     document.querySelector("#cliente").value = dados[index].cliente;
148     document.querySelector("#veiculoId").value = dados[index].veiculoId;
149     document.querySelector("#inicio").value = dados[index].inicio;
150     document.querySelector("#fim").value = dados[index].fim;
151 }
152
153 function enviaDadosParaEdicao() {
154     const dados = new FormData(document.querySelector("form"));
155     const opcoes = {
156         method: "put",
157         credentials: "include",
158         body: new URLSearchParams(dados),
159     };
160     fetch(`${urlBaseAPI}/locaes`, opcoes)
161         .then((res) => {
162             //console.log(res);
163             return res.json();
164         })
165         .then((json) => {
166             //console.log(json);
167             alert("Local de alteração!");
168             carregaDados();
169         });
170     alternaModal();
171 }
172
173 function alternaModal() {
174     document.querySelector("#modal").classList.toggle("mostrarModal");
175 }
176
177 document.querySelector("form button").addEventListener("click", (e) => {
178     e.preventDefault();
179     if (document.querySelector("#id").value) {
180         enviaDadosParaEdicao();
181     } else {
182         enviaDadosParaCadastro();
183     }
184     document.querySelector("#id").value = "";
185     e.target.parentNode.reset();
186 });
187
188 document.querySelector("#btnNovo").addEventListener("click", alternaModal);
189 window.addEventListener("load", () => {
190     checkLogin(); then((res) => {
191         if (res) {
192             carregaVeiculos();
193             carregaDados();
194         } else {
195             window.location = `${urlBaseFront}/login/login.html`;
196         }
197     });
198 });

```

```

1 import { realizaLogin } from './login.js';
2
3 const btLogin = document.querySelector('form button');
4 btLogin.addEventListener('click', (e) => {
5   e.preventDefault();
6   realizaLogin();
7 });

```

events.js do login adiciona um evento ao botão de login que, ao ser clicado, previne o comportamento padrão e chama a função realizaLogin para processar o login do usuário.

```

1 <!DOCTYPE html>
2 <html lang="pt">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <meta http-equiv="X-UA-Compatible" content="IE=edge">
8   <link rel="stylesheet" href="login.css">
9   <title>login</title>
10 </head>
11
12 <body>
13   <main>
14     <form>
15       <label for="email">E-mail</label><br>
16       <input type="text" name="email" id="email"><br>
17       <label for="senha">Senha</label><br>
18       <input type="password" name="senha" id="senha"><br>
19       <button type="button">login</button>
20     </form>
21   </main>
22 </body>
23 <script type="module" src="events.js"></script>
24
25 </html>

```

login.html é uma página de login com um formulário contendo campos para e-mail e senha, e um botão para enviar as informações. O arquivo events.js é carregado para adicionar funcionalidade ao botão de login.

```

1 import { urlBaseAPI, urlBaseFront } from '../url/base.js';
2
3 function realizaLogin() {
4   const data = new FormData(document.forms[0]);
5   const opcoes = {
6     method: 'post',
7     credentials: 'include',
8     body: new URLSearchParams(data),
9   };
10   fetch(`${urlBaseAPI}/user/login`, opcoes)
11     .then((res) => {
12       return res.json();
13     })
14     .then((json) => {
15       if (json.logged) {
16         alert('Autenticado.');

```

O login.js lida com a autenticação de usuários, enviando uma solicitação de login e redirecionando para a página inicial se a autenticação for bem-sucedida. Também inclui uma função para verificar o status de login do usuário.

```

1 import { realizaLogout } from './logout.js';
2
3 window.addEventListener('load', (e) => {
4   realizaLogout();
5 });

```

O events do logout é responsável por realizar o logout do usuário automaticamente quando a página é carregada.

```

1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link rel="stylesheet" href="login.css">
8   <title>Login</title>
9 </head>
10
11 <body>
12   <main>
13     <p>Desconectado</p>
14     <p>
15       <a href="../index.html">Inicio</a><br>
16       <a href="../login/login.html">Login</a><br>
17     </p>
18   </main>
19 </body>
20 </html>
21 <script type="module" src="events.js"></script>
22
23 </html>

```

O logout html cria uma página de desconexão com uma mensagem e links para voltar à página inicial ou à página de login. O script events.js é carregado para gerenciar a lógica de desconexão.

```

1 import { urlBaseAPI, urlBaseFront } from "../url/base.js";
2
3 function realizaLogout() {
4   const opcoes = {
5     method: "get",
6     credentials: "include",
7   };
8   fetch(`${urlBaseAPI}/user/logout`, opcoes)
9     .then((res) => {
10       return res.json();
11     })
12     .then((json) => {
13       window.location = `${urlBaseFront}/index.html`;
14     });
15 }
16
17 export { realizaLogout };
18

```

O logout.js envia uma solicitação para deslogar o usuário e redireciona para a página inicial após a desconexão ser confirmada.

```

1 const urlBaseAPI = "http://localhost:3000";
2 const urlBaseFront = "http://127.0.0.1:5500/front-para-node-fk-com-login";
3
4 export { urlBaseAPI, urlBaseFront };
5

```

A base.js define e exporta as URLs base para a API e o front-end, usadas para fazer solicitações e redirecionar no projeto.


```

1 <!DOCTYPE html>
2 <html lang="pt">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <meta http-equiv="X-UA-Compatible" content="IE=edge">
8
9   <title>Teste</title>
10   <link rel="stylesheet" href="veiculo.css">
11 </head>
12
13 <body>
14   <header>
15     <h1>LISTAGEM DE VEÍCULOS</h1>
16     <a href="#">Inicio</a>
17     <a href="#">Logout</a>
18   </header>
19   <main>
20     <h2>Listagem</h2>
21     <button id="btnovo">Novo</button>
22     <table border="1">
23       <thead>
24         <tr>
25           <th>ID</th>
26           <th>FABRICANTE</th>
27           <th>MODELO</th>
28           <th>COR</th>
29           <th>LISTAGEM</th>
30           <th>AÇÕES</th>
31         </tr>
32       </thead>
33     </table>
34   </body>
35
36   <div id="modal">
37     <form action="#">
38       <input type="hidden" name="id" id="id">
39       <input type="text" name="fabricante" id="fabricante">
40       <input type="text" name="modelo" id="modelo">
41       <input type="text" name="cor" id="cor">
42       <input type="text" name="litragem" id="litragem">
43       <button id="salvar">Salvar</button>
44     </form>
45   </div>
46
47   <script type="module" src="veiculo.js"></script>
48 </html>

```

O veiculo.html cria uma página para listar veículos com uma tabela, um botão para adicionar novos veículos e um modal para edição.

O veiculo.js gerencia a listagem e manipulação de veículos na página. Ele carrega, exibe, adiciona, edita e remove veículos usando chamadas à API, e também controla a exibição do modal de edição. (próxima página)

```

1 import { checklogin } from "../login/login.js";
2 import { urlBaseAPI, urlBaseFront } from "../url/base.js";
3
4 let dados = [];
5
6 function desenhaTabela() {
7   const tbody = document.querySelector("tbody");
8   tbody.innerHTML = "";
9   for (let i = 0; i < dados.length; i++) {
10     const tr = document.createElement("tr");
11     const btEx = document.createElement("button");
12     const btEd = document.createElement("button");
13
14     const td1 = document.createElement("td");
15     const td2 = document.createElement("td");
16     const td3 = document.createElement("td");
17     const td4 = document.createElement("td");
18     const td5 = document.createElement("td");
19     const td6 = document.createElement("td");
20
21     btEx.innerText = "-";
22     btEx.setAttribute("data-id", dados[i].id);
23     btEx.addEventListener("click", (e) => {
24       const id = e.target.getAttribute("data-id");
25       alternaModal();
26       enviaDadosParaDelecao(id);
27     });
28
29     btEd.innerText = "+";
30     btEd.setAttribute("data-index", i);
31     btEd.addEventListener("click", (e) => {
32       const index = e.target.getAttribute("data-index");
33       alternaModal();
34       preencheFormParaEdicao(index);
35     });
36
37     td1.innerText = dados[i].id;
38     td2.innerText = dados[i].fabricante;
39     td3.innerText = dados[i].modelo;
40     td4.innerText = dados[i].cor;
41     td5.innerText = dados[i].litragem;
42     td6.appendChild(btEd, btEx);
43
44     tr.appendChild(td1, td2, td3, td4, td5, td6);
45     tbody.appendChild(tr);
46   }
47 }
48
49 function carregaDados() {
50   const opcoes = {
51     method: "get",
52     credentials: "include",
53   };
54   fetch(`${urlBaseAPI}/veiculos`, opcoes)
55     .then((res) => {
56       //console.log(res);
57       return res.json();
58     })
59     .then((json) => {
60       //console.log(json);
61       //alert(json);
62       dados = json;
63       desenhaTabela();
64     });
65 }
66
67 function enviaDadosParaCadastro() {
68   const dados = new FormData(document.querySelector("form"));
69   const opcoes = {
70     method: "post",
71     credentials: "include",
72     body: new URLSearchParams(dados),
73   };
74   fetch(`${urlBaseAPI}/veiculos`, opcoes)
75     .then((res) => {
76       //console.log(res);
77       return res.json();
78     })
79     .then((json) => {
80       //console.log(json);
81       alert("Veículo " + json.modelo + " cadastrado!");
82       carregaDados();
83     });
84   alternaModal();
85 }
86
87 function enviaDadosParaDelecao(id) {
88   const dados = new FormData();
89   dados.append("id", id);
90   const opcoes = {
91     method: "delete",
92     credentials: "include",
93     body: new URLSearchParams(dados),
94   };
95   fetch(`${urlBaseAPI}/veiculos`, opcoes)
96     .then((res) => {
97       return res.json();
98     })
99     .then((json) => {
100       //console.log(json);
101       alert("Veículo " + json.modelo + " deletado!");
102       carregaDados();
103     });
104   alternaModal();
105 }
106
107 function preencheFormParaEdicao(index) {
108   document.querySelector("#id").value = dados[index].id;
109   document.querySelector("#fabricante").value = dados[index].fabricante;
110   document.querySelector("#modelo").value = dados[index].modelo;
111   document.querySelector("#cor").value = dados[index].cor;
112   document.querySelector("#litragem").value = dados[index].litragem;
113 }
114
115 function enviaDadosParaEdicao() {
116   const dados = new FormData(document.querySelector("form"));
117   const opcoes = {
118     method: "put",
119     credentials: "include",
120     body: new URLSearchParams(dados),
121   };
122   fetch(`${urlBaseAPI}/veiculos`, opcoes)
123     .then((res) => {
124       //console.log(res);
125       return res.json();
126     })
127     .then((json) => {
128       //console.log(json);
129       alert("Veículo " + json.modelo + " alterado!");
130       carregaDados();
131     });
132   alternaModal();
133 }
134
135 function alternaModal() {
136   document.querySelector("#modal").classList.toggle("mostrarModal");
137 }
138
139 document.querySelector("form button").addEventListener("click", (e) => {
140   e.preventDefault();
141   if (document.querySelector("#id").value) {
142     enviaDadosParaEdicao();
143   } else {
144     enviaDadosParaCadastro();
145   }
146   document.querySelector("#id").value = "";
147   e.target.parentNode.reset();
148 });
149
150 document.querySelector("#btnNovo").addEventListener("click", alternaModal);
151 window.addEventListener("load", () => {
152   checklogin().then((res) => {
153     if (res) {
154       carregaDados();
155     } else {
156       window.location = `${urlBaseFront}/login/login.html`;
157     }
158   });
159 });
160

```

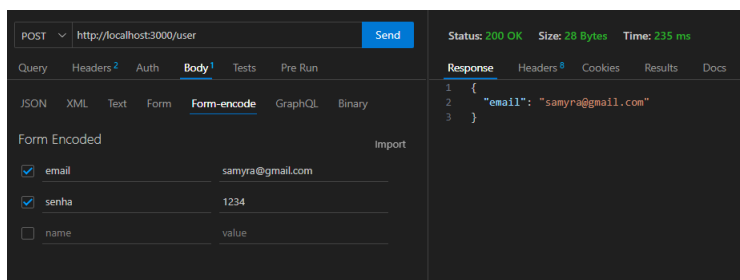
```
1 <!DOCTYPE html>
2 <html lang="pt">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <link rel="stylesheet" href="index.css">
8
9   <title>Prática Node</title>
10 </head>
11 <body>
12   <main>
13     <h1>Prática Node</h1>
14     <a href="./veiculo/veiculo.html">Veículos</a>
15     <a href="./locacao/locacao.html">Locações</a>
16     <a id="link_login" href="./login/login.html">Login</a>
17   </main>
18   <script type="module" src="index.js"></script>
19 </body>
20 </html>
21
```

O index.html é a página principal que fornece links para as páginas de veículos, locações e login.

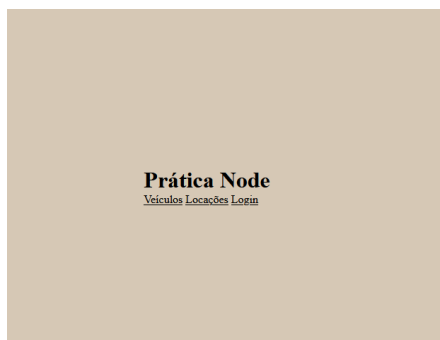
```
1 import { checkLogin } from "../login/login.js";
2
3 const linkLogin = document.getElementById("link_login");
4
5 if (await checkLogin()) {
6   linkLogin.href = "../logout/logout.html";
7   linkLogin.innerText = "Logout";
8 } else {
9   linkLogin.href = "../login/login.html";
10  linkLogin.innerText = "Login";
11 }
12
```

O index.js atualiza o link de login/logout na página principal com base no status de autenticação do usuário, utilizando a função checkLogin para verificar se o usuário está autenticado.

SEGUIE AS TELAS COM AMBOS LIGADOS E EM FUNCIONAMENTO.



criando login no thunder client



tela inicial

E-mail

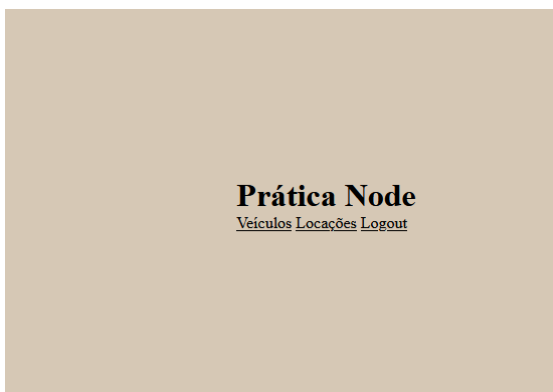
samyra@gmail.com

Senha

....

Login





para acessar foi feito o login



tendo acesso a pagina, desta vez com veículos e locações

[Início](#) [Logout](#)

Listagem
Novo

#	FABRICANTE	MODELO	COR	LISTAGEM	AÇÕES
1	Volkswagen	Audi	preto	01	 
2	Volkswagen AG	Bentley	Prata	02	 

adicionei veículos ao banco

Cliente

Wrias

Veículo

Volkswagen AG Bentley Prata 02 ▾

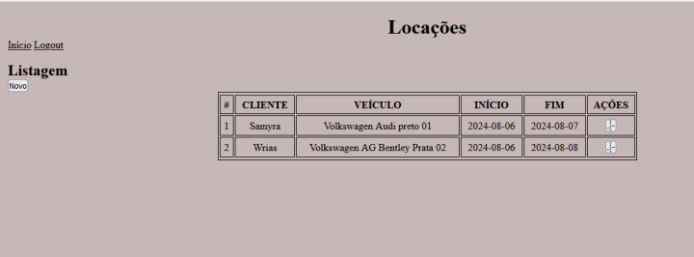
Volkswagen Audi preto 01

Volkswagen AG Bentley Prata 02

dd / mm / aaaa ☐

Salvar

demonstração da foreign key já em locações, podendo escolher o veículo que será alugado

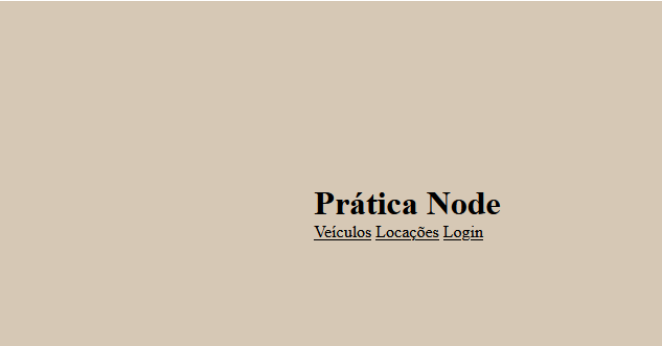


Wrias escolheu o ultimo veículo

cadastrado



cliquei em LOGOUT



voltamos a tela inicial e terá que fazer

login novamente para acessar.