

Projeto e Desenvolvimento de Software II

Prof. Carlos Eduardo de Carvalho Dantas

carloseduardodantas@iftm.edu.br

Parte I – Técnicas e Ferramentas para Desenvolvimento de Sistemas

AGENDA

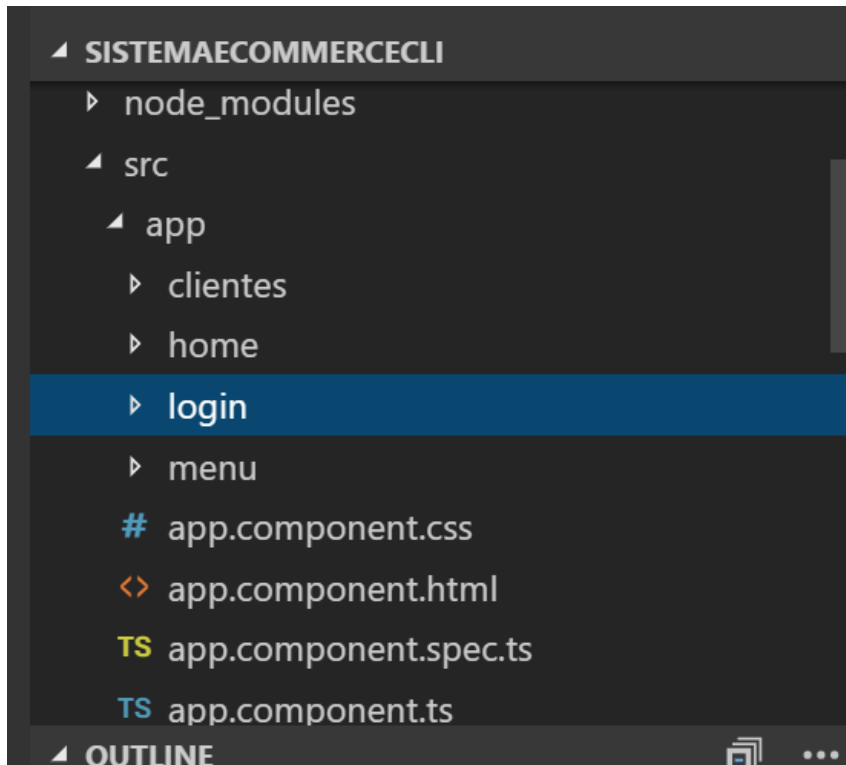
1. Desenvolvimento Front-End

- Histórico
- Aplicações RIA
- Frameworks front-end
- Modelo MVW
- Web Responsiva
- Escalabilidade e websockets

Estudo de Caso: *Angular*

- Criação de login

- Criar o diretório de login. Entrar no diretório e criar o componente de login



```
PS D:\Magistério\IFTM\Aulas\2019\2019-1\Projeto e Desenvolvimento de Software 2\projetos\pds2-2019-1\SistemaEcommerceCLI> cd .\src\  
PS D:\Magistério\IFTM\Aulas\2019\2019-1\Projeto e Desenvolvimento de Software 2\projetos\pds2-2019-1\SistemaEcommerceCLI\src> cd .\app\  
PS D:\Magistério\IFTM\Aulas\2019\2019-1\Projeto e Desenvolvimento de Software 2\projetos\pds2-2019-1\SistemaEcommerceCLI\src\app> cd .\login\  
PS D:\Magistério\IFTM\Aulas\2019\2019-1\Projeto e Desenvolvimento de Software 2\projetos\pds2-2019-1\SistemaEcommerceCLI\src\app\login> ng g c login
```

Estudo de Caso: *Angular*

- Criação de login

- Criar o serviço de login e a interface User

```
-2019-1\SistemaEcommerceCLI\src\app\login> ng g s loginService
```

As a forewarning, we are moving the CLI npm package to "@angular/cli" with the next release, which will only support Node 6.9 and greater. This package will be officially deprecated shortly after.

```
PS D:\Magistério\IFTM\Aulas\2019\2019-1\Projeto e Desenvolvimento de Software 2\projetos\pds
```

```
-2019-1\SistemaEcommerceCLI\src\app\login> ng g i User
```

As a forewarning, we are moving the CLI npm package to "@angular/cli" with the next release, which will only support Node 6.9 and greater. This package will be officially deprecated shortly after.

Estudo de Caso: *Angular*

- Criação de login

- Conteúdo da interface User

```
TS user.ts  x
1  export interface User {
2
3      email: string;
4      password: string;
5  }
6  |
```

Estudo de Caso: *Angular*

- Criação de login

- Conteúdo da classe de Serviço

TS login-service.service.ts x

```
1  import { Injectable, EventEmitter } from '@angular/core';
2  import { Router } from '@angular/router';
3  import { User } from './user';
4
5  @Injectable()
6  export class LoginServiceService {
7
8      public showNavBarEmitter: EventEmitter<boolean> = new EventEmitter<boolean>();
9
10     private authenticated = false;
11
12     constructor(private router: Router) {}
13
14     signIn(user: User) {
15         if ((user.email === 'user@email.com' || user.email === 'usuario@email.com')
16             && user.password === '123456'){
17             this.authenticated = true;
18             this.showNavBar(true);
19             this.router.navigate(['/']);
20         } else {
```

```
20         } else {
21             this.authenticated = false;
22         }
23     }
24
25     logout() {
26         this.authenticated = false;
27         this.showNavBar(false);
28         this.router.navigate(['/signin']);
29     }
30
31     isAuthenticated() {
32         return this.authenticated;
33     }
34
35     private showNavBar(ifShow: boolean) {
36         this.showNavBarEmitter.emit(ifShow);
37     }
38 }
```

Estudo de Caso: *Angular*

- Criação de login

- Conteúdo da classe de componente

TS login.component.ts ✕

```
1  import { Component, OnInit } from '@angular/core';
2  import { FormGroup, FormBuilder, Validators } from '@angular/forms';
3  import { LoginServiceService } from '../login-service.service';
4
5  @Component({
6    selector: 'app-login',
7    templateUrl: './login.component.html',
8    styleUrls: ['./login.component.css']
9  })
10 export class LoginComponent implements OnInit {
11
12   form: FormGroup;
13   error = false;
14   errorMessage = '';
15
16   constructor(
17     private fb: FormBuilder,
18     private authService: LoginServiceService) {}
19
```

```
19
20   onSignIn() {
21     this.authService.signIn(this.form.value);
22   }
23
24   ngOnInit():any {
25     this.form = this.fb.group({
26       email: ['', Validators.required],
27       password: ['', Validators.required],
28     });
29   }
30 }
```


Estudo de Caso: *Angular*

- Criação de login

- Conteúdo do template html

<> login.component.html x

```
1 <h5>Login</h5>
2 <div class="row">
3   <form [formGroup]="form" (ngSubmit)="onSignin()">
4     <div class="row">
5       <div class="form-group">
6         <input id="email" type="email" class="validate"
7           |>
8           formControlName="email"
9           [class.invalid]="form.controls['email'].touched
10            && !form.controls['email'].valid">
11         <label for="email"
12           |>
13           data-error="Invalid email">
14         Email
15       </label>
16     </div>
17   </div>
```

<> login.component.html x

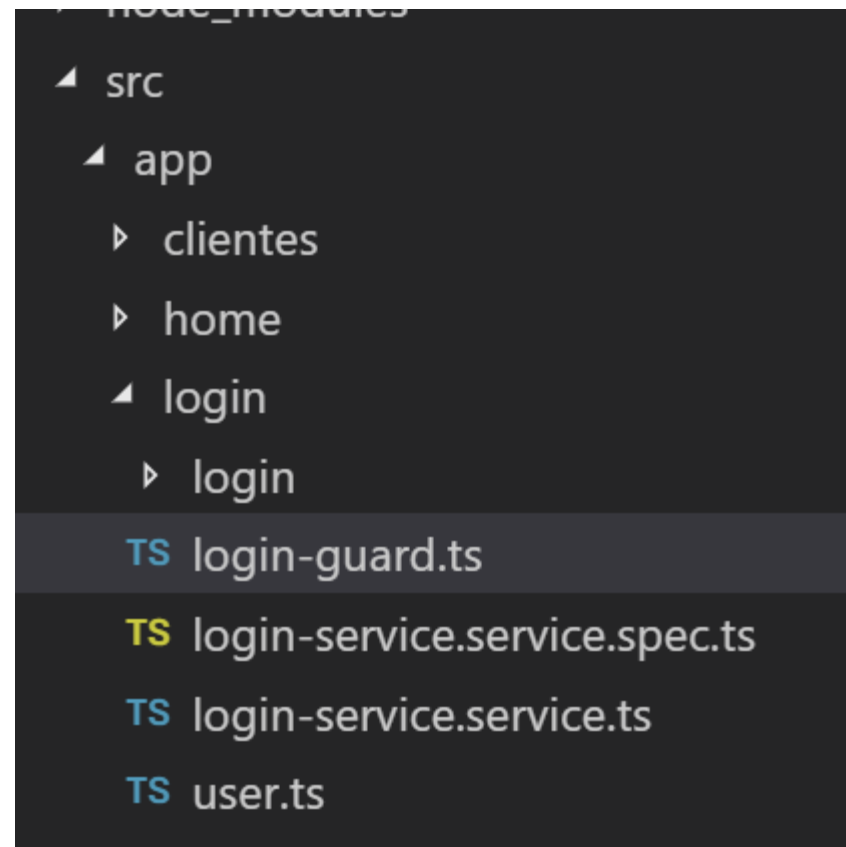
```
16 <div class="row">
17   <div class="form-group">
18     <input id="password" type="password" class="validate"
19       |>
20       formControlName="password"
21       [class.invalid]="form.controls['password'].touched &&
22        !form.controls['password'].valid">
23     <label for="password"
24       |>
25       data-error="Password is mandatory">
26     Password
27   </label>
28 </div>
29 <div class="row">
30   <div class="form-group">
```

```
30 <button class="btn btn-primary" type="submit"
31   |>
32   [disabled]="!form.valid">
33   Entrar
34 </button>
35 </div>
36 </form>
37 </div>
38
```

Estudo de Caso: *Angular*

- Criação de login

- Criar novo arquivo login-guard.ts



Estudo de Caso: *Angular*

- Criação de login

- Criar novo arquivo login-guard.ts

TS login-guard.ts ×

```
1 import { Injectable } from '@angular/core';
2 import { Router, CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot }
3   from '@angular/router';
4 import { Observable } from 'rxjs/Rx';
5
6 import { LoginServiceService } from '../login-service.service';
7
8 @Injectable()
9 export class AuthGuard implements CanActivate {
10
11   constructor(
12     private router: Router,
13     private authService: LoginServiceService) {}
14
```

```
15   canActivate(
16     route: ActivatedRouteSnapshot,
17     state: RouterStateSnapshot): Observable<boolean> | boolean {
18     if (this.authService.isAuthenticated()) {
19       return true;
20     }
21
22     this.router.navigate(['/signin']);
23     return false;
24   }
25 }
```

Estudo de Caso: *Angular*

- Criação de login

- Alterando a rota para não permitir que entre nas telas sem estar logado, e criação da rota de login

```
TS app.router.ts x
11     path: 'nome',
12     component: HomeComponent
13   },
14   {
15     path: 'clientes',
16     loadChildren: 'app/clientes/clientes.module#ClientesModule',
17     canActivate: [AuthGuard]
18   },
19   {
20     path: 'signin',
21     component: LoginComponent
22   }
23 ];
24 export const RoutingModule = RouterModule.forRoot(routes);
```

Estudo de Caso: *Angular*

- Criação de login

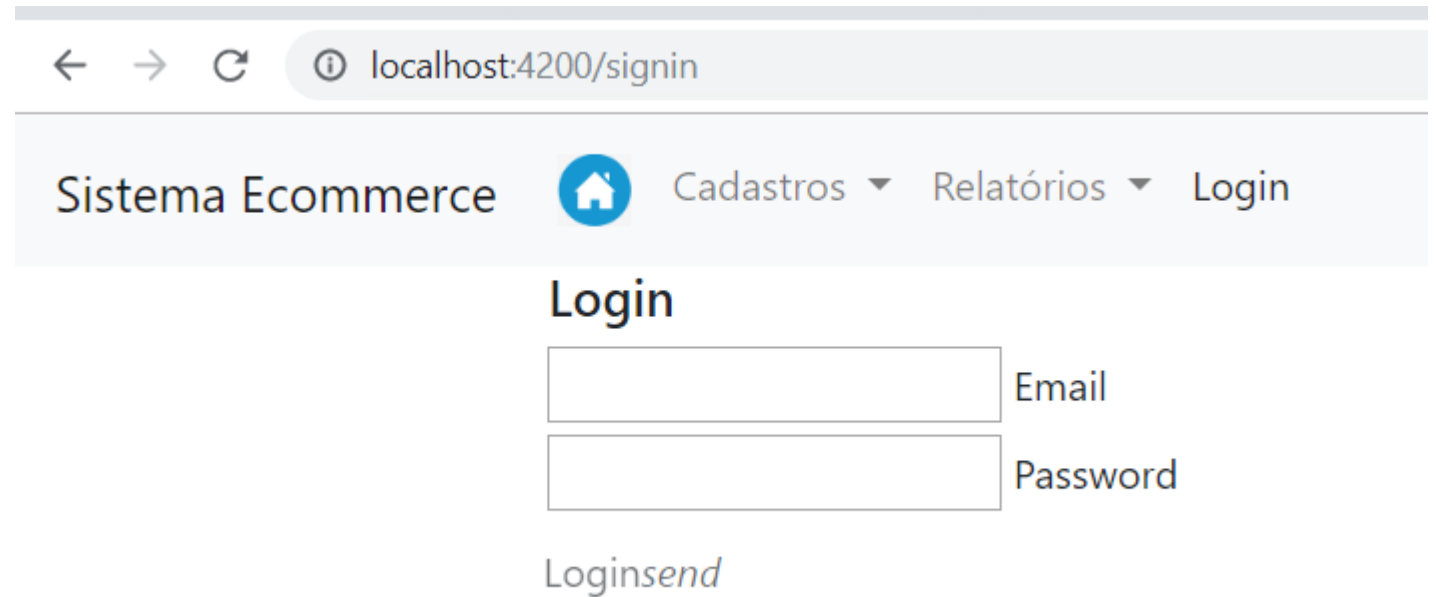
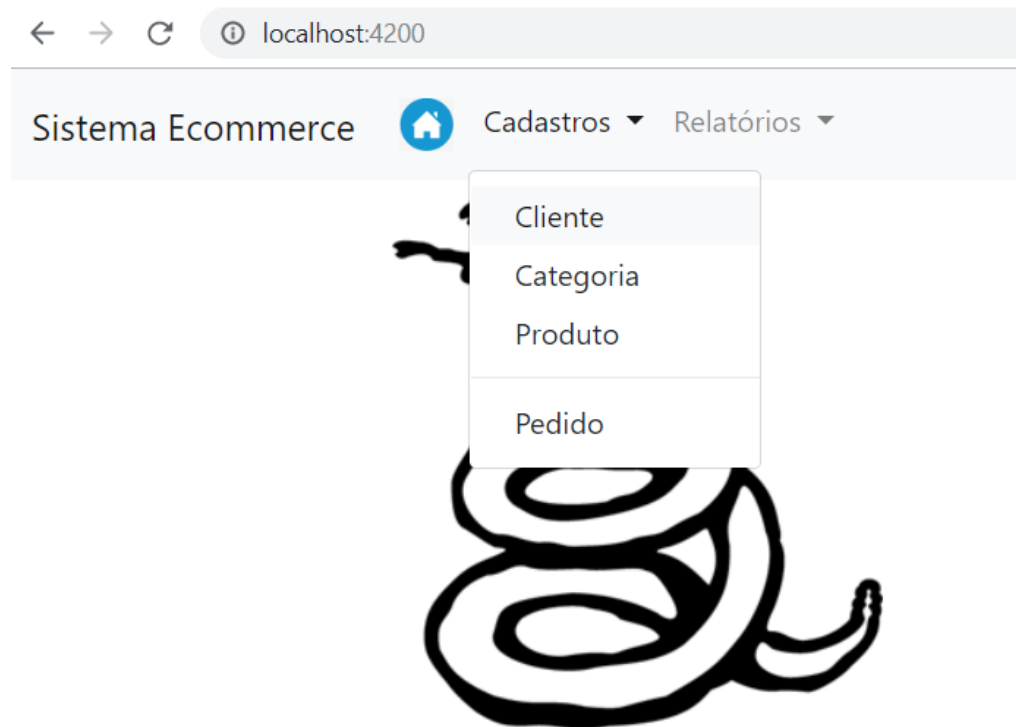
- Importação dos módulos no arquivo app.module.ts

```
TS app.module.ts x
19   LoginComponent
20   ],
21   imports: [
22     BrowserModule,
23     FormsModule,
24     HttpClientModule,
25     RouterModule,
26     ReactiveFormsModule
27   ],
28   providers: [
29     LoginServiceService,
30     AuthGuard
31   ],
32   bootstrap: [AppComponent]
33 })
```

Estudo de Caso: *Angular*

- Criação de login

- Ao testar a tela, não entra mais em cliente sem estar logado.



Estudo de Caso: *Angular*

- Criação de login

- Alteração no menu

```
<> menu.component.html x
32     </a>
33     <div class="dropdown-menu" aria-labelledby="navbarDropdown">
34       <a class="dropdown-item" href="relatorioPedido.html">Pedido</a>
35     </div>
36   </li>
37   <li class="nav-item active" *ngIf="!isAuth()">
38     <a class="nav-link" routerLink="signin">
39       Login
40     </a>
41   </li>
42   <li class="nav-item active" *ngIf="isAuth()">
43     <a class="nav-link" (click)="onLogout()">
44       Logout
45     </a>
46   </li>
47 </ul>
```

Estudo de Caso: *Angular*

- Criação de login
 - Alteração no menu

TS menu.component.ts ✕

```
8  })
9  export class MenuComponent implements OnInit {
10
11      constructor(private authService: LoginServiceService) { }
12
13      private showNavBar: boolean = false;
14
15      ngOnInit(){
16          this.authService.showNavBarEmitter.subscribe(
17              (mode: boolean) => {
18                  if (mode !== null) {
19                      this.showNavBar = mode;
20                  }
21              }
22          );
23      }
24  }
```


```
25  isAuth() {
26      return this.authService.isAuthenticated();
27  }
28
29  onLogout() {
30      this.authService.logout();
31  }
32
33  }
```


Estudo de Caso: *Angular*

- Criação de login

- Existirão os botões login e logout

← → ↻ ⓘ localhost:4200/signin


Sistema Ecommerce  Cadastros ▼ Relatórios ▼ Login

Login

Email

Password

← → ↻ ⓘ localhost:4200

Sistema Ecommerce  Cadastros ▼ Relatórios ▼ Logout



Estudo de Caso: *Angular*

- Criação de login

- Ocultar

```
<> menu.component.html x
14      </li>
15      <li class="nav-item dropdown" *ngIf="showNavBar">
16          <a class="nav-link dropdown-toggle" href="#" id="navbarDropd
17              aria-haspopup="true" aria-expanded="false">
18              Cadastros
19      </li>
```

```
<> menu.component.html x
26      </div>
27      </li>
28      <li class="nav-item dropdown" *ngIf="showNavBar">
29          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
30              aria-haspopup="true" aria-expanded="false">
31              Relatórios
```

Estudo de Caso: *Angular*

- Exercício

- Construir o back-end para login (pode incluir os usuários diretamente no banco, sem necessidade de CRUD completo)

LINKS DOS SOFTWARES UTILIZADOS

- **Visual Studio Code** - <https://code.visualstudio.com/>
- **Node JS** - <https://nodejs.org/en/>
- Link repositório GITHUB: <https://github.com/carloseduardoxp/pds2-2019-1>

REFERÊNCIAS

- GUEDES, Thiago. Crie aplicações com Angular. Casa do Código, 2018.
- GRONER, Loiane. Curso Angular. Disponível em: <https://www.youtube.com/watch?v=tPOMG0D57S0&list=PLGxZ4Rq3BOBoSRcKWEdQACbUCNWLczg2G>. Acesso em 02/04/2019.