

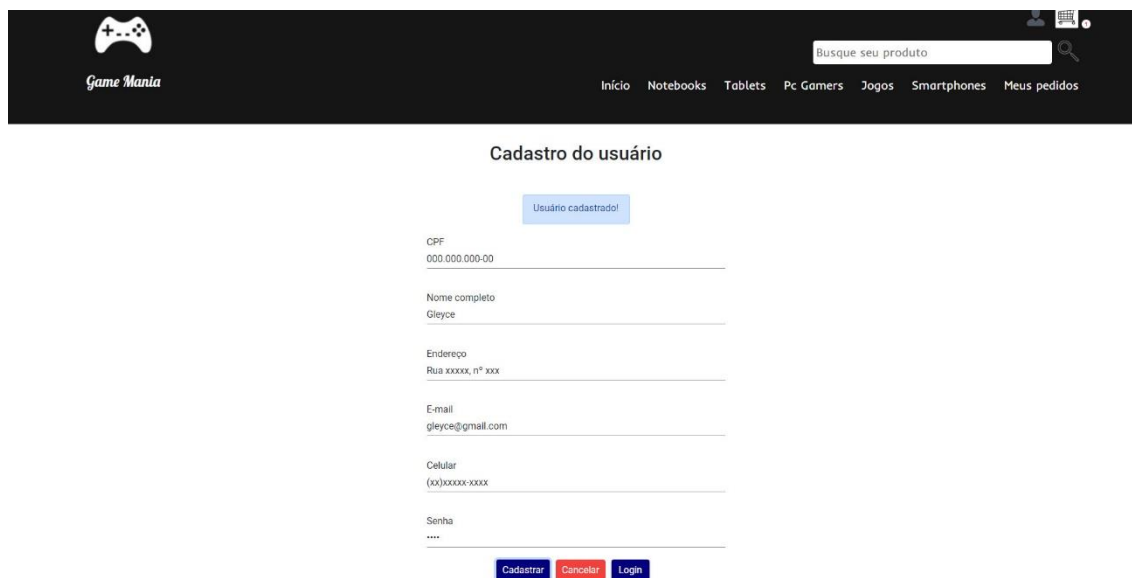
Game Mania – (UC10 – Atividades).

Aluno: Marcelo Carneiro Marques

Parte I - CENÁRIOS POSSÍVEIS.

Cadastro de usuário - Com sucesso.

1- Usuário realiza seu cadastro com sucesso.



Cadastro do usuário

Usuário cadastrado!

CPF
000.000.000-00

Nome completo
Gleyce

Endereço
Rua xxxxx, nº xxx

E-mail
gleyce@gmail.com

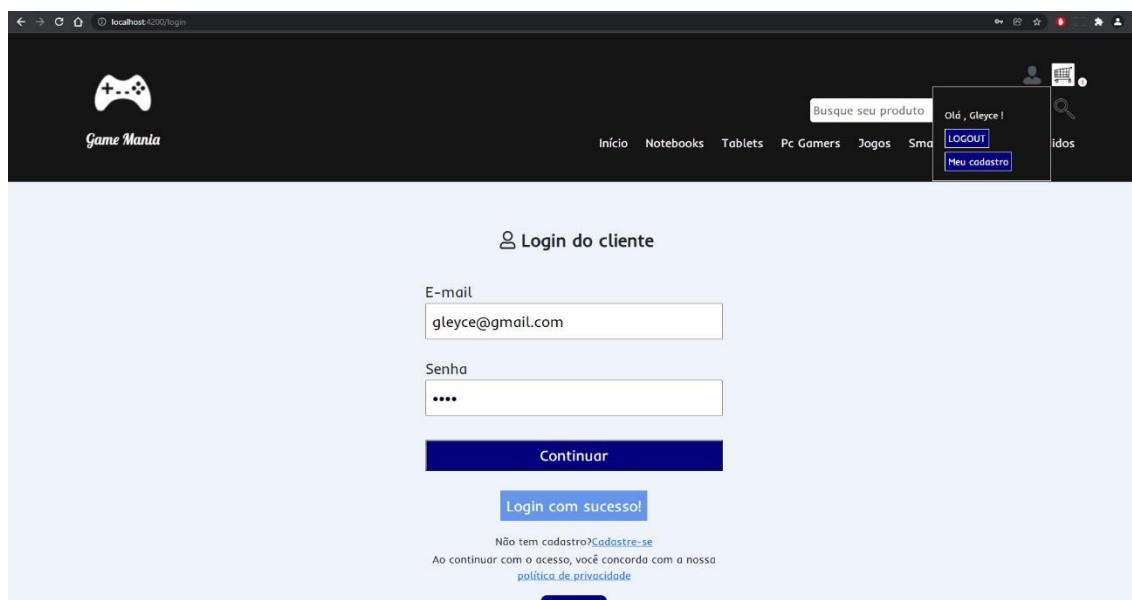
Celular
(xx)xxxx-xxxx

Senha

[Cadastrar](#) [Cancelar](#) [Login](#)

Login de usuário - com sucesso.

2- Usuário realiza o login com sucesso.



Login do cliente

E-mail
gleyce@gmail.com

Senha

[Continuar](#)

[Login com sucesso!](#)

Não tem cadastro? [Cadastre-se](#)

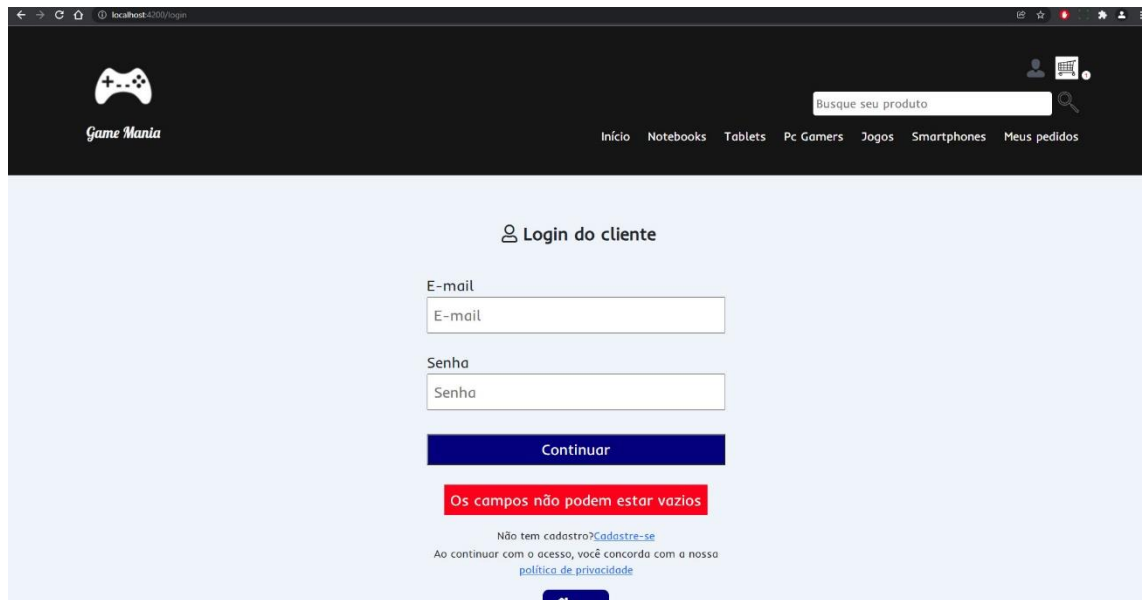
Ao continuar com o acesso, você concorda com a nossa [política de privacidade](#)

Olá, Gleyce!
[LOGOUT](#)
[Meu cadastro](#)

Aluno: Marcelo Carneiro Marques

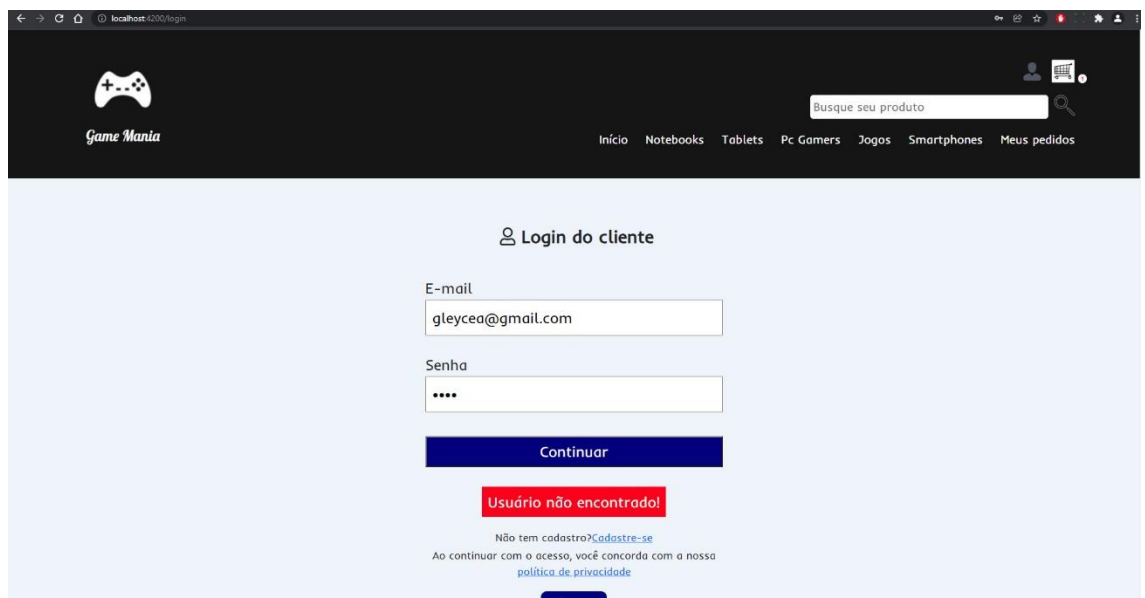
Login de usuário - com erro.

3- Usuário tenta se cadastrar sem preencher os campos de login.

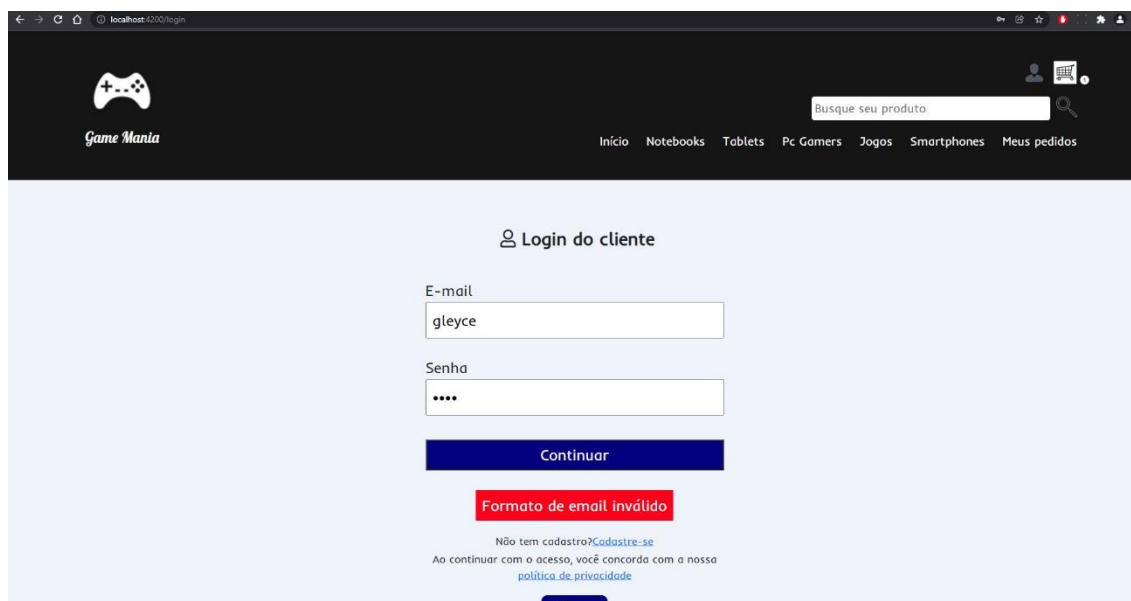


The screenshot shows the 'Game Mania' website's login page. The header includes the site logo, a search bar, and navigation links: Início, Notebooks, Tablets, Pc Gamers, Jogos, Smartphones, and Meus pedidos. The main content area is titled 'Login do cliente' and contains two input fields: 'E-mail' and 'Senha'. Both fields are empty. Below the fields is a blue 'Continuar' button. A red error message box displays the text 'Os campos não podem estar vazios'. At the bottom, there is a link to 'Cadastre-se' and a note about the privacy policy.

4- Usuário tenta fazer o login com usuário inexistente.



The screenshot shows the 'Game Mania' website's login page. The header is identical to the previous one. In the 'Login do cliente' section, the 'E-mail' field is filled with 'gleycea@gmail.com' and the 'Senha' field is filled with four dots. The blue 'Continuar' button is visible. A red error message box displays the text 'Usuário não encontrado!'. At the bottom, there is a link to 'Cadastre-se' and a note about the privacy policy.

5- Usuário insere formato de e-mail inválido.

The screenshot shows the 'Login do cliente' page of the Game Mania website. The email field contains 'gleyce' and the password field contains four dots. A red error message 'Formato de email inválido' is displayed below the 'Continuar' button. The page includes a navigation bar with links to 'Início', 'Notebooks', 'Tablets', 'Pc Gamers', 'Jogos', 'Smartphones', and 'Meus pedidos'. A search bar is also present in the top right corner.

Game Mania

Busque seu produto

Início Notebooks Tablets Pc Gamers Jogos Smartphones Meus pedidos

Login do cliente

E-mail
gleyce

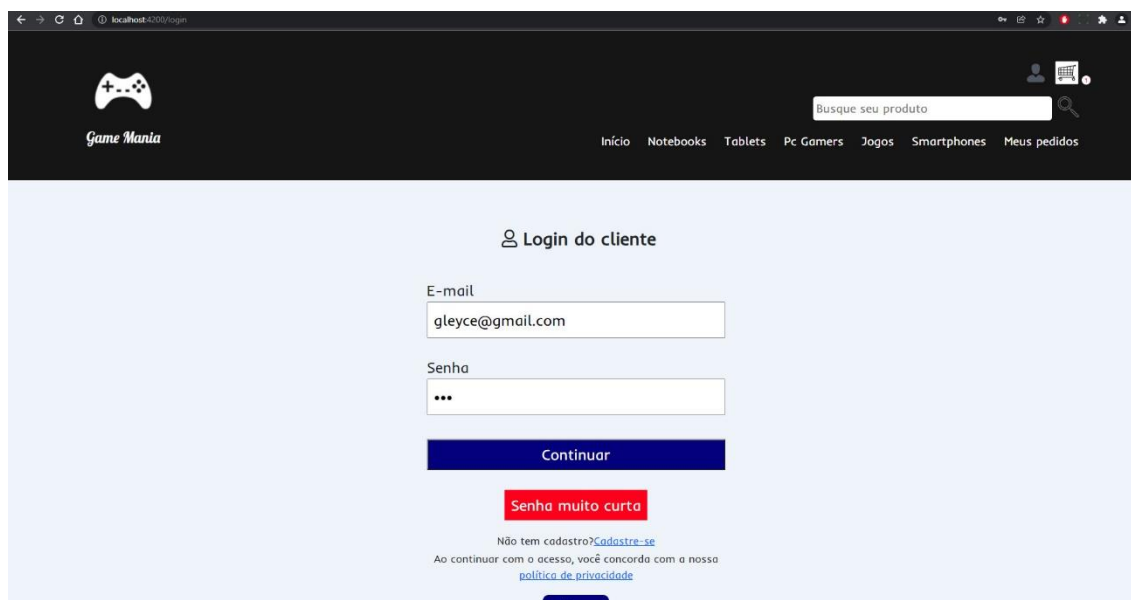
Senha
....

Continuar

Formato de email inválido

Não tem cadastro? [Cadastre-se](#)

Ao continuar com o acesso, você concorda com a nossa [política de privacidade](#)

6- Usuário insere senha muito curta.

The screenshot shows the 'Login do cliente' page of the Game Mania website. The email field contains 'gleyce@gmail.com' and the password field contains three dots. A red error message 'Senha muito curta' is displayed below the 'Continuar' button. The page includes a navigation bar with links to 'Início', 'Notebooks', 'Tablets', 'Pc Gamers', 'Jogos', 'Smartphones', and 'Meus pedidos'. A search bar is also present in the top right corner.

Game Mania

Busque seu produto

Início Notebooks Tablets Pc Gamers Jogos Smartphones Meus pedidos

Login do cliente

E-mail
gleyce@gmail.com

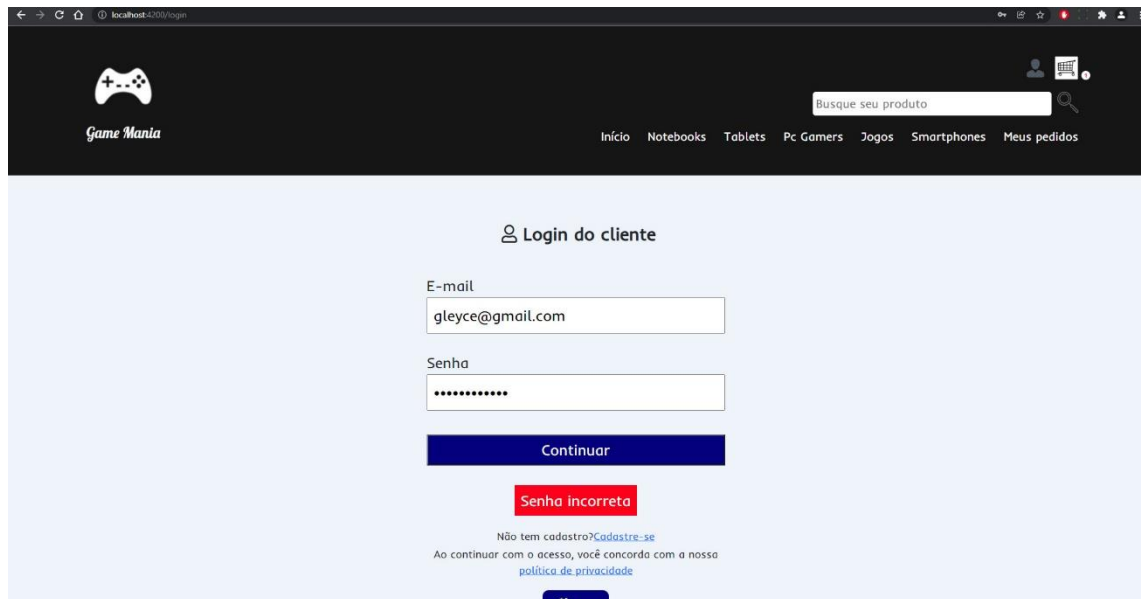
Senha
...

Continuar

Senha muito curta

Não tem cadastro? [Cadastre-se](#)

Ao continuar com o acesso, você concorda com a nossa [política de privacidade](#)

7- Usuário insere senha incorreta.

The screenshot shows the 'Login do cliente' form on the Game Mania website. The E-mail field contains 'gleyce@gmail.com' and the Senha field contains a masked password. A red error message 'Senha incorreta' is displayed below the password field. Below the error message, there is a link to 'Cadastrar-se' and a note about the privacy policy.

Game Mania

Busque seu produto

Início Notebooks Tablets Pc Gamers Jogos Smartphones Meus pedidos

Login do cliente

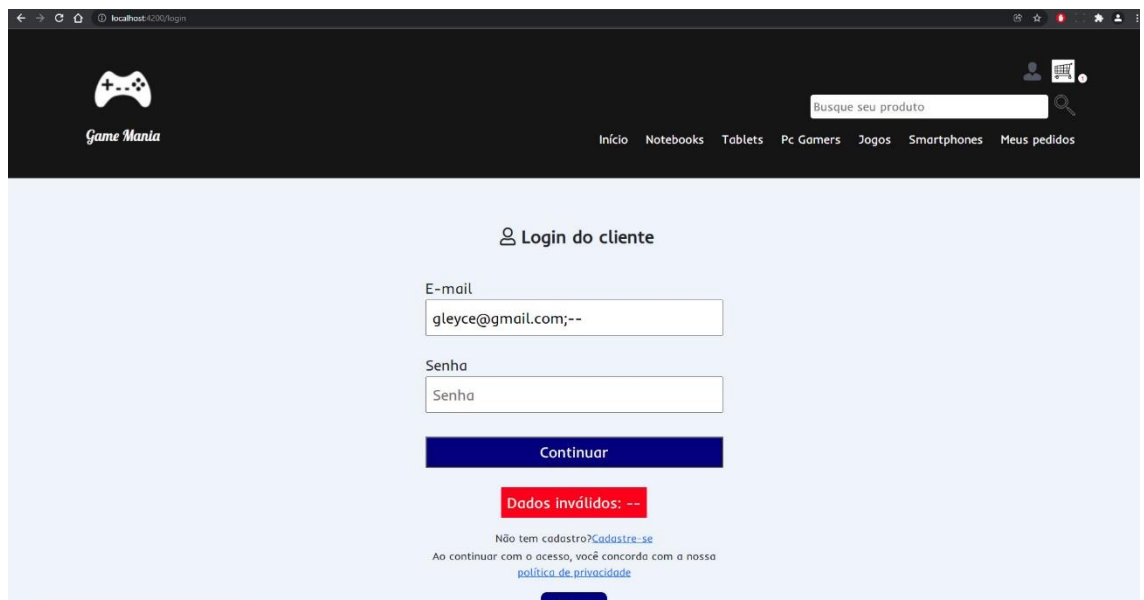
E-mail
gleyce@gmail.com

Senha
.....

Continuar

Senha incorreta

Não tem cadastro? [Cadastrar-se](#)
Ao continuar com o acesso, você concorda com a nossa [política de privacidade](#)

8- Terceiros tentam burlar o sistema para obter dados (SQL Injection).

The screenshot shows the 'Login do cliente' form on the Game Mania website. The E-mail field contains 'gleyce@gmail.com;--' and the Senha field contains 'Senha'. A red error message 'Dados inválidos: --' is displayed below the password field. Below the error message, there is a link to 'Cadastrar-se' and a note about the privacy policy.

Game Mania

Busque seu produto

Início Notebooks Tablets Pc Gamers Jogos Smartphones Meus pedidos

Login do cliente

E-mail
gleyce@gmail.com;--

Senha
Senha

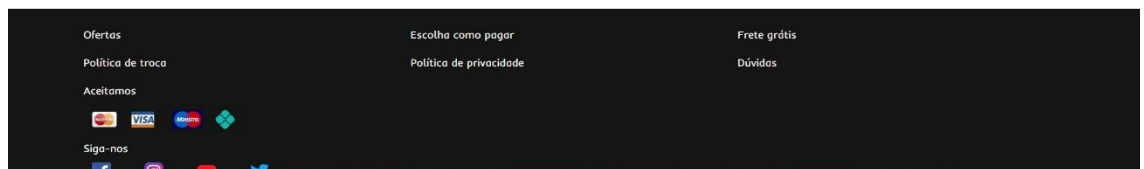
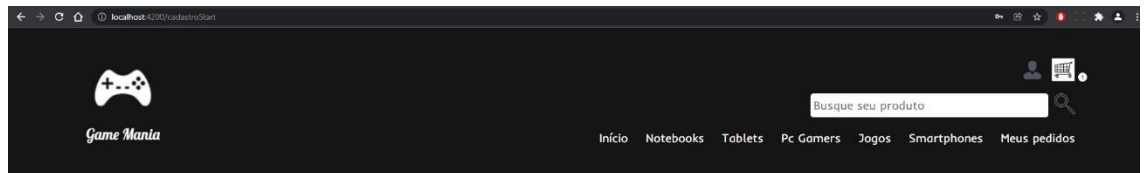
Continuar

Dados inválidos: --

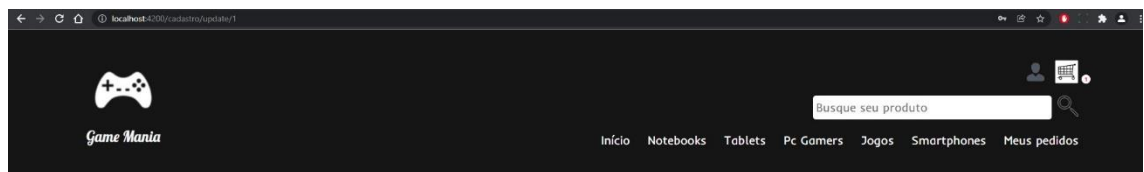
Não tem cadastro? [Cadastrar-se](#)
Ao continuar com o acesso, você concorda com a nossa [política de privacidade](#)

Operações no sistema.

9- Usuário logado pode visualizar seus dados cadastrados.



10- Usuário logado pode atualizar seus dados.



Atualizar dados

Caso atualize os dados, não esqueça de fornecer uma nova senha

Nome
Gleyce

Endereço
Rua xxxxx, nº xxx

E-mail
gleyce@gmail.com

Celular
(xx)xxxxx-xxxx

Senha

[Atualizar](#) [Cancelar](#)

11- Usuário logado pode apagar seu cadastro, encerrando sua conta.

The screenshot shows a web browser window with the URL `localhost:4200/cadastro/delete/1`. The page header features the 'Game Mania' logo and a navigation menu with links: 'Início', 'Notebooks', 'Tablets', 'Pc Gamers', 'Jogos', 'Smartphones', and 'Meus pedidos'. A search bar with the placeholder 'Busque seu produto' is also present. The main content area displays a form titled 'Encerrar conta' with the following fields:

- Nome:** Gleyce
- Endereço:** Rua xxxxx, nº xxx
- Email:** gleyce@gmail.com
- Celular:** (xx)xxxxxx-xxxx
- Senha:** (masked with dots)

At the bottom of the form are two buttons: 'Encerrar conta' (highlighted in red) and 'Cancelar'.

PARTE II - Estrutura do código***login.ts***

```
export class Login {  
  constructor(  
    public email: string = "",  
    public password?: string ,  
    public id: number = 1  
  ){}  
}
```

cadastro.model.ts

```
export interface Cadastro{  
  id?: number,  
  ssn: string,  
  name: string,  
  address: string,  
  email: string,  
  mobile: string,  
  password: string  
}
```

db.json

```
{  
  "users": [  
    {  
      "email": "gleyce@gmail.com",  
      "password": "$2a$10$bzJP1X.Uy/lRIrM0HqNC30P00Kww3VIpaicLgmnEnMrVHnliVXb6a",  
      "ssn": "000.000.000-00",  
      "name": "Gleyce",  
      "mobile": "(xx)xxxxxx-xxxx",  
      "address": "Rua xxxxx, nº xxx",  
      "id": 1  
    }  
  ]  
}
```

Para Login***login.service.ts***

```
import { Observable } from 'rxjs';  
import { HttpClient, HttpHeaders } from '@angular/common/http';  
import { Injectable } from '@angular/core';  
import { Login } from '../models/login';  
  
@Injectable({  
  providedIn: 'root'  
})  
export class LoginService {  
  
  url = "http://localhost:3000/login"  
  
  constructor(private httpClient: HttpClient) { }  
  
  login(user: Login): Observable<any>{  
    return this.httpClient.post(this.url, JSON.stringify(user), {  
      headers: new HttpHeaders({'Content-Type': 'application/json'}),  
      observe: 'response',  
    })  
  }  
}
```

login.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Login } from 'src/app/models/login';
import { LoginService } from 'src/app/services/login.service';
import { LoginStatus } from 'global';
import { StatusService } from 'src/app/services/status.service';
import { Router } from '@angular/router';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
export class LoginComponent implements OnInit {

  loginStatus: LoginStatus = new LoginStatus("", "", false);

  public loginModel = new Login();

  message: string = '';
  responseStatus: string = '';

  constructor(private loginService: LoginService, private statusService: StatusService,
    private router: Router) {}

  ngOnInit(): void {
    this.statusService.currentStatus.subscribe(status => this.loginStatus = status)
  }

  onSubmit(){

    const blacklist: string[] = ["select ", "from ", "drop ", "or ", "having ",
      "group ", "by ", "insert ", "exec ", "\"'", "\"'", "--", "#", "*", ";"];

    for(let i = 0 ; i < blacklist.length; i++){
      if(this.loginModel.email.toLowerCase().includes(blacklist[i])){
        this.message = "Dados inválidos: " + blacklist[i];
        return;
      }
    }

    this.loginService.login(this.loginModel).subscribe((response) => {

      this.message = "Login com sucesso!";
      this.responseStatus = response.status;

      this.loginStatus.email = (JSON.parse(JSON.stringify(response.body.user))).email;
      this.loginStatus.username = (JSON.parse(JSON.stringify(response.body.user))).name;
      this.loginStatus.active = true;

      this.statusService.changeStatus(this.loginStatus);

      setTimeout(() => {
        this.router.navigateByUrl('');
      }, 2000);

    }, err => {

      if(err.error == "Email and password are required"){
        this.message = "Os campos não podem estar vazios";
      }else if(err.error == "Email format is invalid"){
        this.message = "Formato de email inválido";
      }else if(err.error == "Cannot find user"){
        this.message = "Usuário não encontrado!";
      }else if(err.error == "Password is too short"){
        this.message = "Senha muito curta";
      }else if(err.error == "Incorrect password"){
        this.message = "Senha incorreta";
      }
    })
  }
}
```


Aluno: Marcelo Carneiro Marques

Para cadastro

cadastro.service.ts (serviço para realizar o CRUD em relação aos usuários).

```
import { Injectable } from '@angular/core';
import { MatSnackBar } from '@angular/material/snack-bar';
import { HttpClient } from '@angular/common/http';
import { Cadastro } from '../cadastro.model';
import { catchError, map } from 'rxjs/operators';
import { EMPTY, Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class CadastroService {

  baseUrl = "http://localhost:3000/users"

  constructor(private snackBar: MatSnackBar, private http: HttpClient) {}

  showMessage(msg: string, isError: boolean = false): void {
    this.snackBar.open(msg, 'X', {
      duration: 3000,
      verticalPosition: "top",
      panelClass: isError ? ['errorMsg'] : ['successMsg']
    });
  }

  create(cadastro: Cadastro): Observable<Cadastro>{
    return this.http.post<Cadastro>(this.baseUrl, cadastro).pipe(
      map((obj) => obj),
      catchError(e => this.errorMsg(e))
    )
  }

  errorMsg(e: any){
    console.log(e);
    this.showMessage('Erro - E-mail já existe', true);
    return EMPTY;
  }

  read(): Observable<Cadastro[]>{
    return this.http.get<Cadastro[]>(this.baseUrl).pipe(
      map((obj) => obj),
      catchError(e => this.errorMsgLoad(e))
    );
  }

  errorMsgLoad(e: any){
    console.log(e);
    this.showMessage('Erro no carregamento', true);
    return EMPTY;
  }

  readById(id: number): Observable<Cadastro>{
    const url = `${this.baseUrl}/${id}`;
    return this.http.get<Cadastro>(url);
  }

  updateCadastro(cadastro: Cadastro): Observable<Cadastro>{
    const url = `${this.baseUrl}/${cadastro.id}`;
    return this.http.put<Cadastro>(url, cadastro);
  }

  deleteCadastro(cadastro: Cadastro): Observable<Cadastro>{
    const url = `${this.baseUrl}/${cadastro.id}`;
    return this.http.delete<Cadastro>(url);
  }
}
```

cadastro-create.component.ts (criar cadastro)

```
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Cadastro } from '../cadastro.model';
import { CadastroService } from '../cadastro.service';

@Component({
  selector: 'app-cadastro-create',
  templateUrl: './cadastro-create.component.html',
  styleUrls: ['./cadastro-create.component.css']
})
export class CadastroCreateComponent implements OnInit {

  cadastro: Cadastro = {
    ssn: '',
    name: '',
    email: '',
    mobile: '',
    address: '',
    password: ''
  }

  message: string = '';

  constructor( private cadastroService: CadastroService, private router: Router) {

  }

  ngOnInit(): void {
  }

  createCadastro(): void{
    this.cadastroService.create(this.cadastro).subscribe(() => {
      this.message = "Usuário cadastrado!";
      setTimeout(() => {
        this.router.navigateByUrl('login');
      }, 2000);
    });
  }

  cancelarCadastro(): void{
    this.router.navigate([''])
  }

  login(){
    this.router.navigate(['login']);
  }
}
```

cadastro-update.component.ts (atualizar cadastro).

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { Cadastro } from '../cadastro.model';
import { CadastroService } from '../cadastro.service';

@Component({
  selector: 'app-cadastro-update',
  templateUrl: './cadastro-update.component.html',
  styleUrls: ['./cadastro-update.component.css']
})
export class CadastroUpdateComponent implements OnInit {

  newPassword : string = '';

  cadastro = {} as Cadastro;

  constructor(private cadastroService: CadastroService,
              private router: Router,
              private route: ActivatedRoute) { }

  ngOnInit(): void {
    const id = +this.route.snapshot.paramMap.get('id')!;
    this.cadastroService.readById(id).subscribe(cadastro => {
      this.cadastro.id = cadastro.id;
      this.cadastro.ssn = cadastro.ssn;
      this.cadastro.name = cadastro.name;
      this.cadastro.address = cadastro.address;
      this.cadastro.email = cadastro.email;
      this.cadastro.mobile = cadastro.mobile;
    })
  }

  updateCadastro(){
    this.cadastro.password = this.newPassword;
    this.cadastroService.updateCadastro(this.cadastro).subscribe(() => {
      this.cadastroService.showMessage('O usuário foi atualizado');
      this.router.navigateByUrl('/cadastroStart');
    })
  }

  cancelarCadastro(): void{
    this.router.navigateByUrl('/cadastroStart');
  }
}
```

cadastro-delete.component.ts (apagar cadastro).

```
import { StatusService } from '../../../services/status.service';
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { Cadastro } from '../cadastro.model';
import { CadastroService } from '../cadastro.service';

@Component({
  selector: 'app-cadastro-delete',
  templateUrl: './cadastro-delete.component.html',
  styleUrls: ['./cadastro-delete.component.css']
})
export class CadastroDeleteComponent implements OnInit {

  cadastro = {} as Cadastro;

  constructor(private cadastroService : CadastroService,
              private router: Router,
              private route : ActivatedRoute,
              private statusService: StatusService) { }

  ngOnInit(): void {
    const id = +this.route.snapshot.paramMap.get('id')!;
    this.cadastroService.readById(id).subscribe((cadastro) => {
      this.cadastro = cadastro;
    })
  }

  deleteCadastro(): void{
    this.cadastroService.deleteCadastro(this.cadastro).subscribe(() => {
      this.cadastroService.showMessage('Cadastro deletado');
      this.router.navigateByUrl('/cadastroStart');
      this.statusService.changeStatus({ email: '', username: '', active: false })
    })
  }

  cancelarCadastro(): void{
    this.router.navigateByUrl('/cadastroStart');
  }
}
```


cadastro-read.component.ts (ver dados cadastrados).

```
import { Component, OnInit } from '@angular/core';
import { Cadastro } from '../cadastro.model';
import { CadastroService } from '../cadastro.service';
import { StatusService } from 'src/app/services/status.service';

@Component({
  selector: 'app-cadastro-read',
  templateUrl: './cadastro-read.component.html',
  styleUrls: ['./cadastro-read.component.css']
})
export class CadastroReadComponent implements OnInit {

  cadastros : Cadastro[] = [];
  displayedColumns = ["id", "cpf", "nome", "endereço", "email", "cel", "action"];
  activeEmail: string = '';

  constructor(private cadastroService: CadastroService, private statusService: StatusService) {
  }

  ngOnInit(): void {

    this.statusService.currentStatus.subscribe(status => {
      this.activeEmail = status.email
    })

    this.cadastroService.read().subscribe(cadastros => {
      this.cadastros = (cadastros.filter(item => item.email == this.activeEmail));
    })
  }
}
```

cadastro-start.component.ts (componente auxiliar para exibir formulário de cadastro ou dados cadastrados, dependendo se o usuário está logado no sistema ou não).

```
import { StatusService } from 'src/app/services/status.service';
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-cadastro-start',
  templateUrl: './cadastro-start.component.html',
  styleUrls: ['./cadastro-start.component.css']
})
export class CadastroStartComponent implements OnInit {

  active: boolean = false;

  constructor(private statusService: StatusService) {
  }

  ngOnInit(): void {
    this.statusService.currentStatus.subscribe(status => {
      this.active = status.active;
    })
  }
}
```

status.service.ts (serviço auxiliar para monitorar se usuário está logado e notificar componentes interessados fornecendo-lhes os dados pertinentes).

```
import { Injectable } from '@angular/core';
import { LoginStatus } from 'global';
import { BehaviorSubject } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class StatusService {

  private subject = new BehaviorSubject<LoginStatus>({ email: '', username: '', active: false });

  currentStatus = this.subject.asObservable();

  constructor() { }

  changeStatus(status: LoginStatus) {
    this.subject.next(status);
  }
}
```

Criação de classe global para verificação do status do usuário (ativo ou não).

global.ts

```
export class LoginStatus {

  public email: string = '';
  public active: boolean = false;
  public username : string = '';

  constructor(email: string, username: string, active: boolean){
    this.email = email;
    this.username = username;
    this.active = active;
  }
}
```

Aluno: Marcelo Carneiro Marques

Rotas para chamar os componentes de acordo com o endereço URL.

```
import { NotFoundComponent } from './views/not-found/not-found.component';
import { CadastroStartComponent } from './componentes/cadastro/cadastro-start/cadastro-start.component';
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { CadastroCreateComponent } from './componentes/cadastro/cadastro-create/cadastro-create.component';
import { CadastroDeleteComponent } from './componentes/cadastro/cadastro-delete/cadastro-delete.component';
import { CadastroReadComponent } from './componentes/cadastro/cadastro-read/cadastro-read.component';
import { CadastroUpdateComponent } from './componentes/cadastro/cadastro-update/cadastro-update.component';
import { CarrinhoComponent } from './views/carrinho/carrinho.component';
import { CategoriasComponent } from './views/categorias/categorias.component';
import { FaleconoscoComponent } from './views/faleconosco/faleconosco.component';
import { HomeComponent } from './views/home/home.component';
import { LoginComponent } from './views/login/login.component';
import { PedidosComponent } from './views/pedidos/pedidos.component';
import { ProdutosComponent } from './views/produtos/produtos.component';

const routes: Routes = [
  { path: "", component: HomeComponent },
  { path: "login", component: LoginComponent },
  { path: "carrinho", component: CarrinhoComponent },
  { path: "pedidos", component: PedidosComponent },
  { path: "categorias", component: CategoriasComponent },
  { path: "produtos", component: ProdutosComponent },
  { path: "faleconosco", component: FaleconoscoComponent },
  { path: "cadastroStart", component: CadastroStartComponent },
  { path: "cadastro", component: CadastroCreateComponent },
  { path: "cadastroRead", component: CadastroReadComponent },
  { path: "cadastro/update/:id", component: CadastroUpdateComponent },
  { path: "cadastro/delete/:id", component: CadastroDeleteComponent },
  { path: "404", component: NotFoundComponent },
  { path: "**", redirectTo: "404" }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```