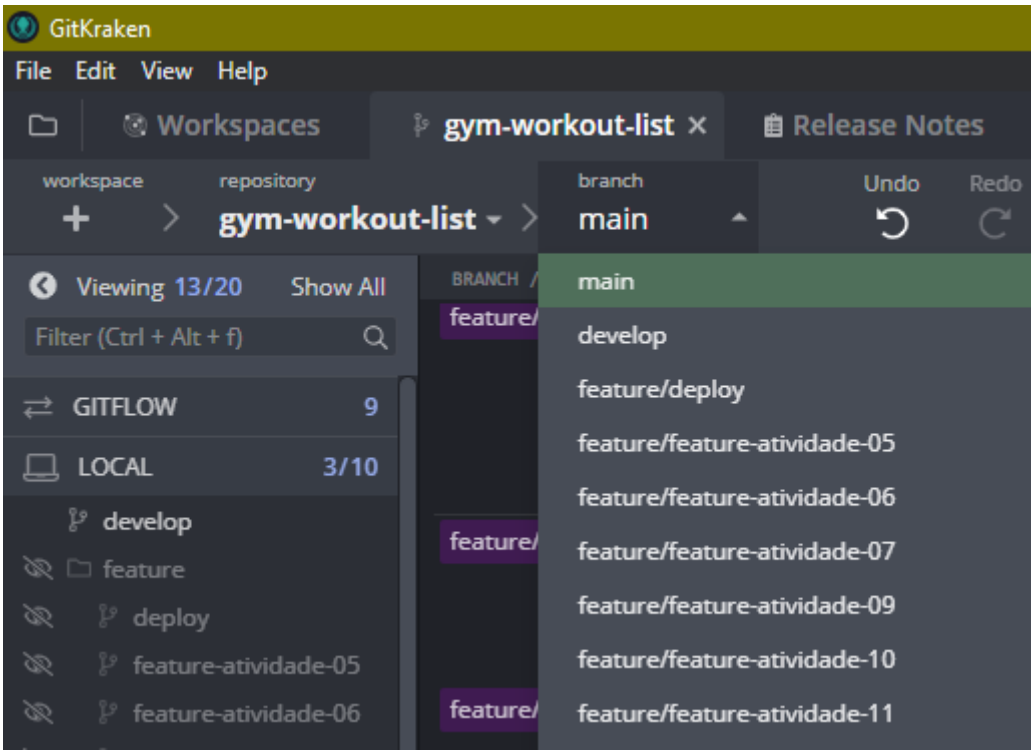
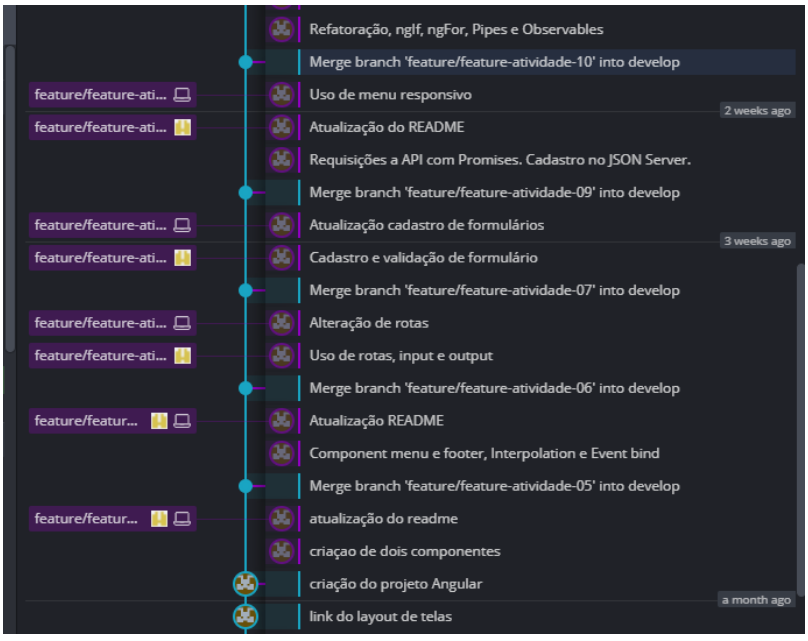
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

RELATÓRIO DE CRIAÇÃO DO PROJETO GYMWORKOUTLIST

1. **Criar o repositório no GitHub com a estrutura do Gitflow, ou seja, branches main e develop.**



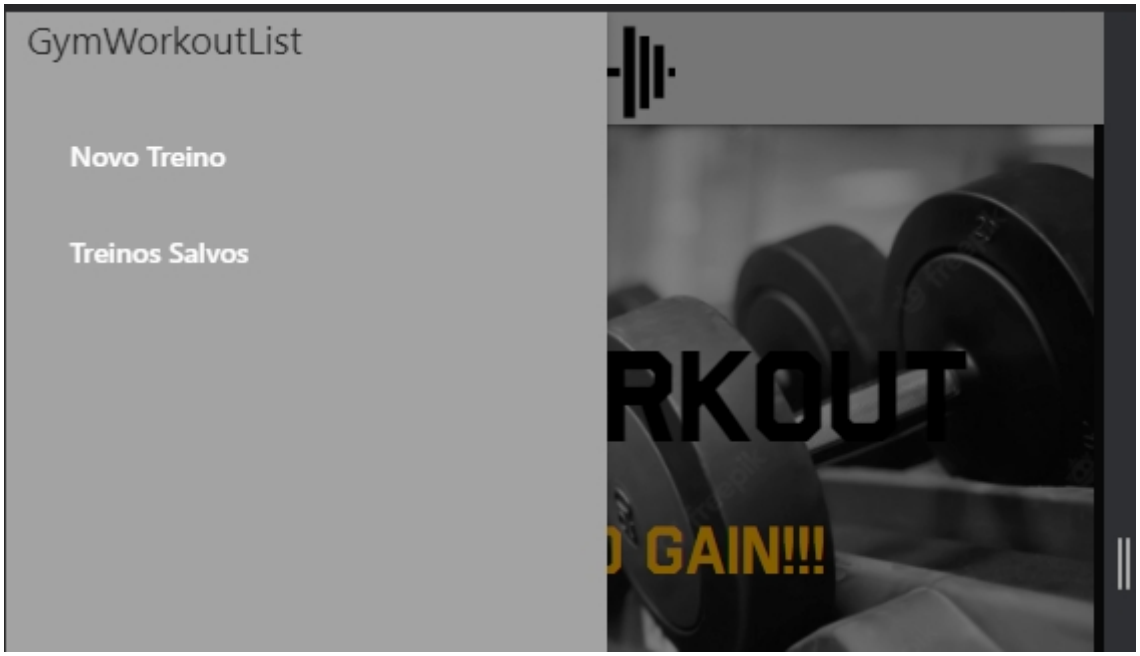
2. **Criar branches feature a partir do branch develop para cada nova funcionalidade a ser implementada.**



3. **Usar algum framework CSS (Bootstrap, Materialize ou outro). Optei pelo Materialize.**
   Dependências no package.json.

```
"dependencies": {
  "@angular/animations": "~12.1.0-",
  "@angular/common": "~12.1.0-",
  "@angular/compiler": "~12.1.0-",
  "@angular/core": "~12.1.0-",
  "@angular/forms": "~12.1.0-",
  "@angular/platform-browser": "~12.1.0-",
  "@angular/platform-browser-dynamic": "~12.1.0-",
  "@angular/router": "~12.1.0-",
  "json-server": "^0.17.0",
  "material-icons": "^1.10.11",
  "materialize-css": "^1.0.0",
```

4. **Apresentar as telas com layout responsivo.**



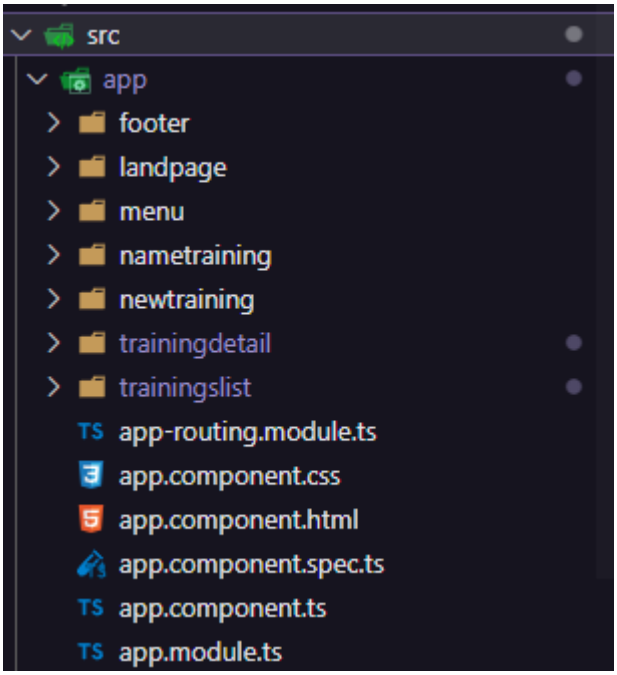
5. **Usar componentes de algum framework CSS (Bootstrap, Materialize ou outro)**

Exemplo de uso de classes do materialize.

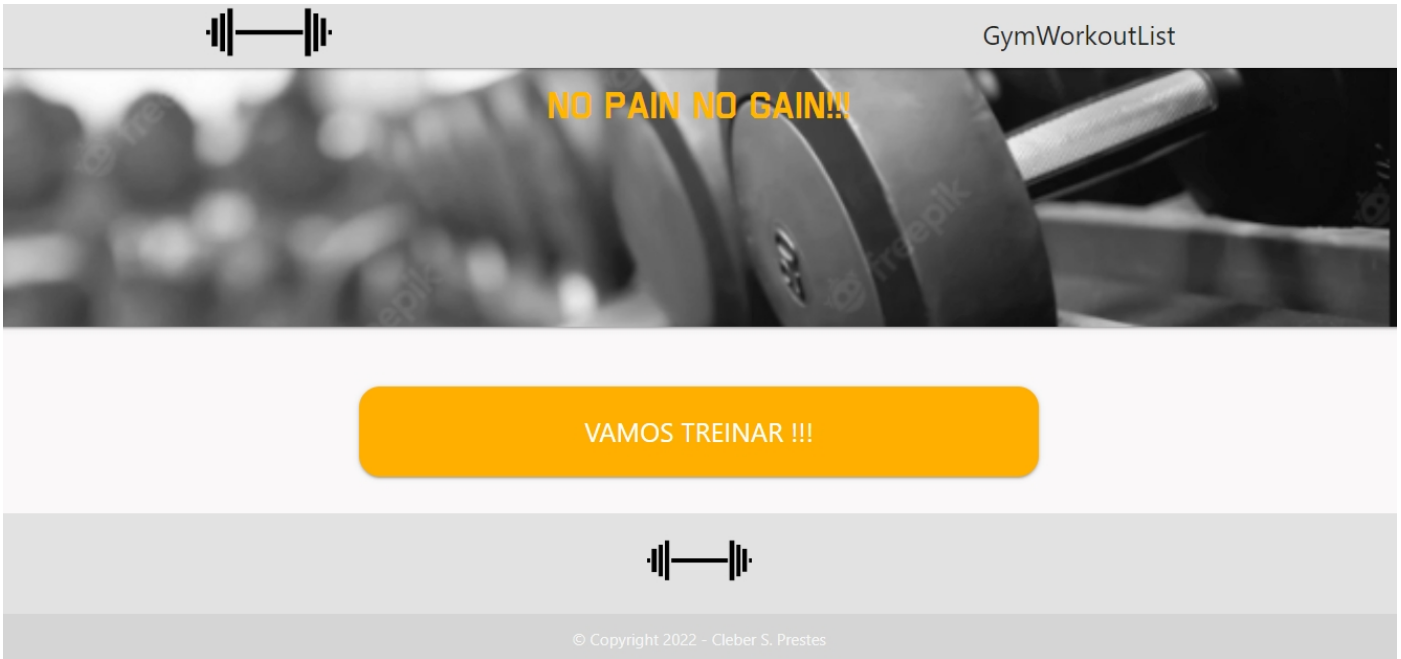
```
<div class="navbar-fixed">
  <nav>
    <div class="nav-wrapper grey lighten-2 accent-4">
      <div class="container">
        <a class="brand-logo" href="index.html">
          <img class="my-logo" src="assets/images/logogym.png" />
        </a>
        <a href="#" data-target="mobile-demo" class="sidenav-trigger"
          ><i class="material-icons">menu</i></a
        >
        <ul class="right hide-on-med-and-down">
          <li>
            <a href="#" class="flow-text grey-text text-darken-4" >GymWorkoutList</a>
          </li>
          </ul>
      </div>
    </div>
  </nav>
</div>
```

6. **Apresentar as telas com layout responsivo usando ou não algum framework CSS.**

7. **Construir páginas web com o conceito de componentes. Componentes criados foram: Menu, Footer, Landpage, nametraining, traininiglist, newtraining, trainingdetail.**



8. **Criar o layout da aplicação com componentes, ou seja, o cabeçalho e rodapé precisam ser componentes.**



9. **Usar pelo menos dois tipos de data-binding (Interpolation, Property Binding, Event Binding e Two Way Data Binding). Interpolation foram utilizados em varias partes, como exemplo no texto que apresenta a origem dos dados, se localStorage ou Json-sever. Two Way Data Binding foi utilizado para inserção de dados no form de cadastro dos treinos.**

Uso de interpolation e Two Way Data Binding

```
<h5>{{dataOrigin}}</h5>

  <div class="app-loading" *ngIf="showLoading">
  <div class="logo"></div>
  <svg class="spinner" viewBox="25 25 50 50">
    <circle
      class="path"
      cx="50"
      cy="50"
      r="20"
      fill="none"
      stroke-width="2"
      stroke-miterlimit="10"
    />
  </svg>
```

```
  <div class="card-panel red accent-4 center my-card-button">
    <input id="training-serie"  name="training-serie" min='0'
    type="number" class="validate"
    [(ngModel)]="training.serie" placeholder="Series"
    pattern="\d+"
    #serie="ngModel"
    required >
    <div class="white-text" [hidden]="serie.valid || serie.untouched">
      Serie deve ser um numero.
    </div>
  </div>
```

10. **Passar dados via hierarquia de componentes, ou seja, usando @Input ou @Output. Foram utilizados para o titulo "Lista de Treinos". Foi criado um componente somente para essa funcionalidade.**

```
@Output() informacaoPai =  'Lista de Treinos'
@Output() listaOutput: Training[]=[]

constructor(private trainingService: TrainingserviceService) {
  this.getTrainings()
}
```

11. **Mapear componentes à rotas no módulo de rotas.**

```
const routes: Routes = [
{path:'',component: LandpageComponent},
{path:'newtraining', component:NewtrainingComponent},
{path:'trainingslist', component:TrainingslistComponent},
{path:'trainingslist/detail/:id', component:TrainingdetailComponent, resolve:{TrainingResolver}}
];

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

12. Criar navegação entre páginas por meio de rotas. São exemplos da navegação entre as paginas a implementação dos botões de voltar, Criar Treinos e Treinos Salvos.

```
<div class="container about"  *ngIf="this.showButtons">
  <div id="novo-treino" class="row">
    <div class="col s12 m6 offset-m3">
      <a routerLink="/newtraining">
        <div
          class="card-panel red accent-4 center my-card-button"
        >
          <span class="white-text flow-text"> NOVO TREINO </span>
        </div>
      </a>
    </div>
  </div>
<div id="treino-salvo" class="row">
  <div class="col s12 m6 offset-m3">
    <a routerLink="/trainingslist">
      <div
        class="card-panel red accent-4 center my-card-button">
        <span class="white-text flow-text" > TREINOS SALVOS </span>
      </div>
    </a>
  </div>
```

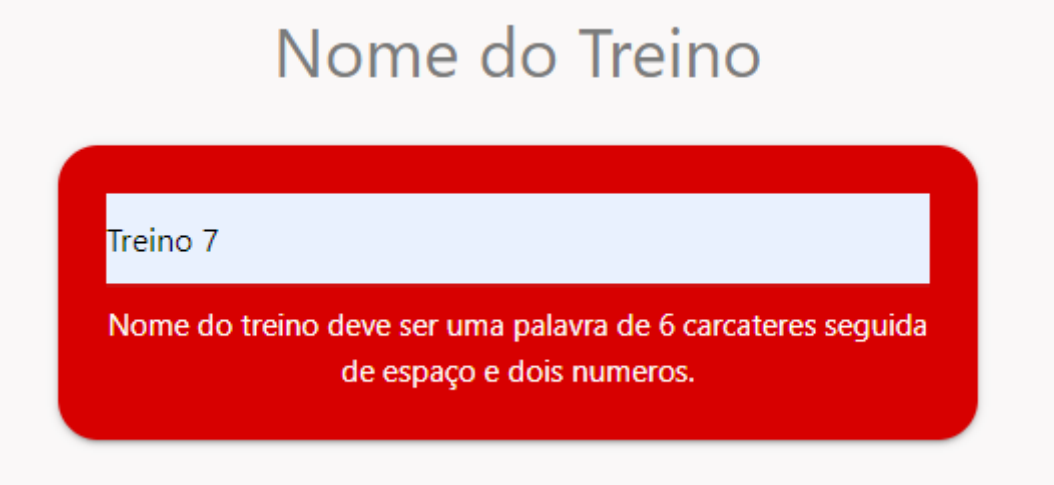
13. Passar dados entre componentes que representam diferentes telas via parâmetros de rotas. Foi utilizado no botão detalhes para passar o id de um treino e exibir os detalhes dele no componente trainingdetail.

```
<td ><button  class="waves-effect waves-light btn-small  grey darken-4" (click)
="testRemoveComService(d)" *ngIf="this.jsonOrigin">Del Server</button></td>
<td><button  class="waves-effect waves-light btn-small  amber "  [routerLink]="['/trainingslist/
detail', d.id]" >Detalhes</button></td>
```

```
ngOnInit(): void {

  let idParam: any = this.route.snapshot.params.id
  let trainingListLocal = JSON.parse(localStorage.getItem('listaTreino')!) as Training[];
  trainingListLocal = trainingListLocal.filter((t) => {
    return t.id === idParam})
    this.localTraining = trainingListLocal[0]

}
```

14. Validar campos do formulário com REGEX e apresentar os erros. Os campos validados foram o Nome do treino e o nome do Exercicio no componente newtraining.

```
  id="entrada-treino"
  class="input-field card-panel red accent-4 center my-card-button">
 <input
 id="training-name"
  name="training-name"
  type="text"
 class="input-field validate white-text "
 placeholder="Nome do treino"
 [(ngModel)]="training.name"
 [(ngModel)]="training.id"
 pattern="[A-ZÀ-Úa-z]{6}\s\d{2}"
 #name="ngModel"
 required
/>
<div class="white-text" [hidden]="name.valid || name.untouched">
  Nome do treino deve ser uma palavra  de 6 carcateres seguida de espaço e dois numeros.
</div>
```



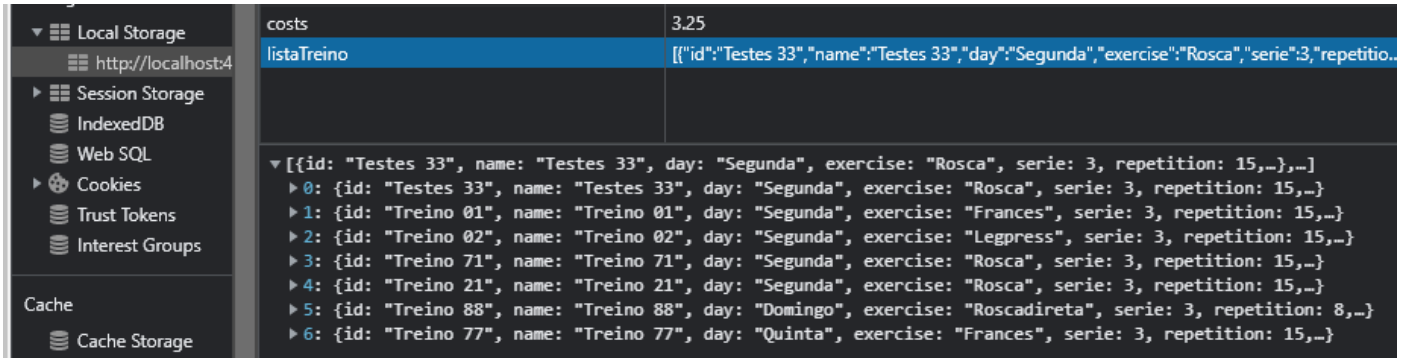
15. **Desabilitar o botão de submit enquanto o formulário está inválido. No componente newtraining o botão de Salvar só é habilitado se o formulario estiver válido.**

```
<button
class=" waves-effect waves-light btn-large card-panel amber accent-4 center my-card-button
white-text flow-text"
type="submit"
*ngIf="this.localOrigin"
[disabled]="!form.valid">SALVAR LOCAL</button>
```

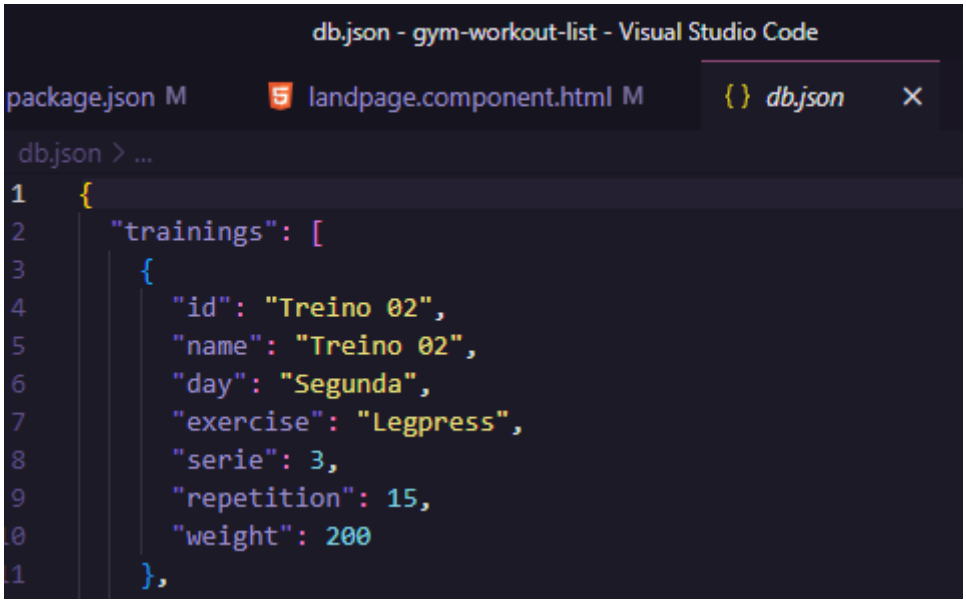
16. **Fazer requisições a API com tratamento da resposta com Promises ou Observables. Foi utilizado o Json-sever como API, o tratamento de resposta utilizado foi para o carregamento de dados do JSon-sever ou do localStorage.**

```
getAll(): Observable<Training[]>{

 return this.http.get<Training[]>(this.apiUrl);
}


addTraining(training: Training){
  return this.http.post<Training>(this.apiUrl, training);
}


getTraininigWithPromise(): Promise<Training[]> {
  return this.http.get<Training[]>(this.apiUrl).toPromise();
}

getTraininigWithObservable():Observable<Training[]>{
  return this.http.get<Training[]>(this.apiUrl)
}
```

17. **Cadastrar uma entidade no Web Storage.**



18. **Cadastrar uma entidade no JSON Server.**



19. **Apresentar uma lista de dados com a diretiva estrutural ngFor.A lista de treinos cadastrados é mostrada utilizando a diretiva ngFor.**

```
<tbody >
  <tr *ngFor="let d of this.trainingListSave" >
    <td >{{d.name}}</td>
    <td ><button  class="waves-effect waves-light btn-small  red darken-4" (click)="removeTraining(d)
    " *ngIf="this.localOrigin" >Del Local</button></td>
    <td ><button  class="waves-effect waves-light btn-small  grey darken-4" (click)
    ="testRemoveComService(d)" *ngIf="this.jsonOrigin">Del Server</button></td>
    <td><button  class="waves-effect waves-light btn-small  amber "  [routerLink]="['/trainingslist/
    detail', d.id]" >Detalhes</button></td>
  </tr>
</tbody>
```

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

20. **Usar a diretiva nglf. A diretiva nglf foi utilizada em varios lugares. Como exemplo no botão treinar que esconde ou mostra os botões de Novo treino e Treino Salvo.**

```
<div class="container about"  *ngIf="this.showButtons">
  <div id="novo-treino" class="row">
    <div class="col s12 m6 offset-m3">
      <a routerLink="/newtraining">
        <div
          class="card-panel red accent-4 center my-card-button"
        >
          <span class="white-text flow-text"> NOVO TREINO </span>
        </div>
      </a>
    </div>
  </div>
```