Banking) com Angular 6 — na prática e sem complicações parte 3

Danilo Agostinho Follow

Sep 7, 2018 · 7 min read

- Salve pessoal!

Chegamos a parte 3 da nossa série de posts para o **Training Center.**

Caso tenha perdido a primeira e segunda parte, não deixe de acompanhar clicando nos links abaixo:

Parte 1;

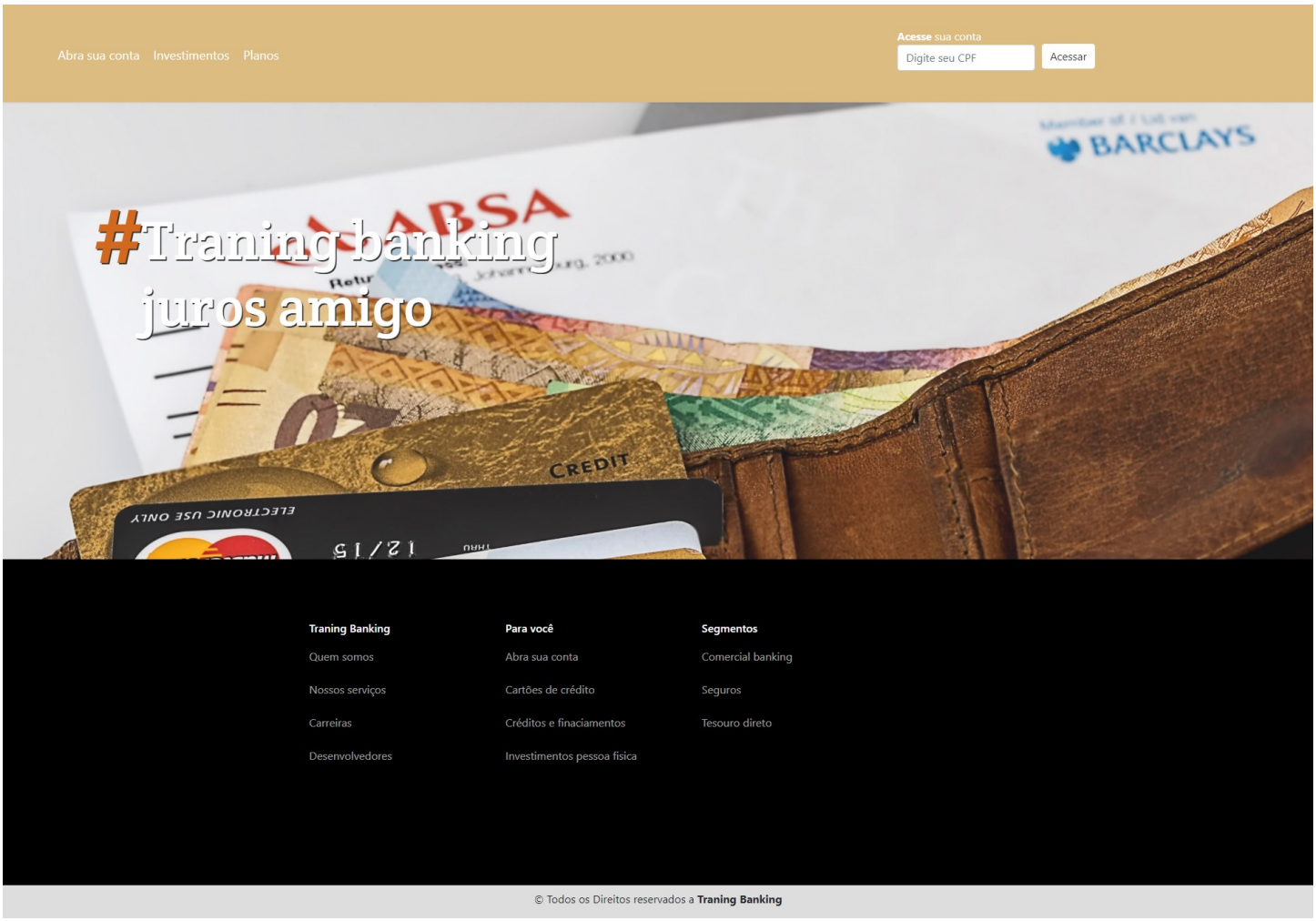
Parte 2.

Você também poderá baixar o projeto que está no Github clicando neste link.

Não poderia deixar também de agradecer o carinho dos leitores e o feedback positivo que tenho recebido. Eles são muito importantes para que eu continue motivado a escrever.

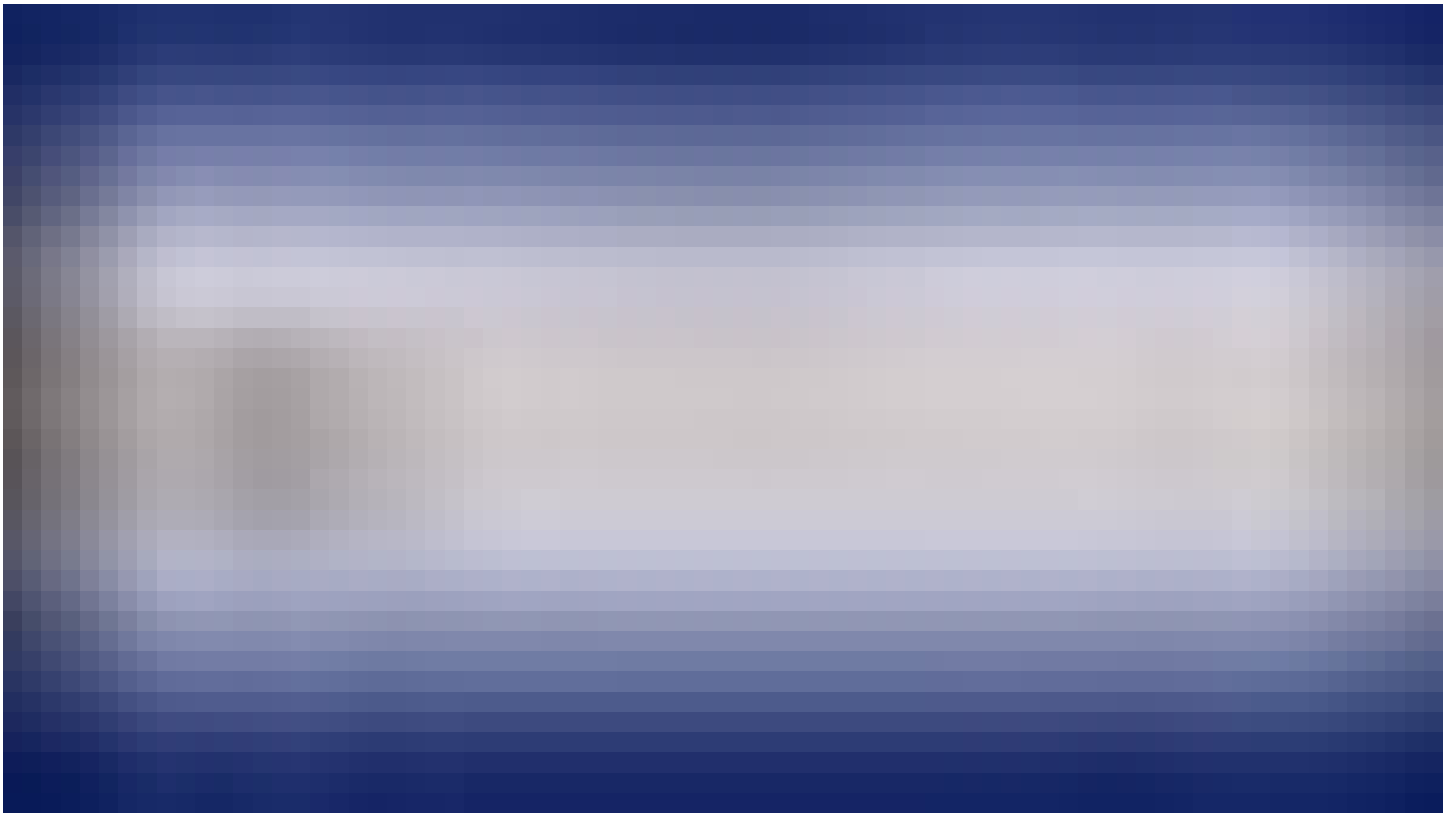
Onde paramos

No exato momento nossa aplicação está assim:



Incrível! Eu tô curtindo muito, e vocês? Nesse tutorial vamos criar o roteamento da aplicação com um módulo do Angular nativo chamado Router. Entenda como roteamento sendo a criação das rotas da aplicação (como se fosse o menu: **home**, **produtos**, **categorias**). Essas rotas, quando são clicadas, chamam os components correspondentes de cada rota.

Exemplo: ao clicar no botão **Crie sua conta** ativamos o component **cadastro-conta**, que é exibido no lugar do component **content.**



Observe que a imagem mostra uma parte chamada **router-outlet,** onde estiver sido declarado essa tag, os components irão aparecer. Portanto, vamos editar nosso **app.component.html** com o código abaixo.

Código do arquivo app.component.html

```
<app-header></app-header>
<router-outlet></router-outlet>
<app-footer></app-footer>
```

Apenas substituímos a tag do component **app-content** pela tag **router-outlet.**

Para usar o router-outlet precisamos declará-lo no app.module.ts e realizar algumas configurações básicas. Então, abra o arquivo **app.module.ts** e siga os passos:

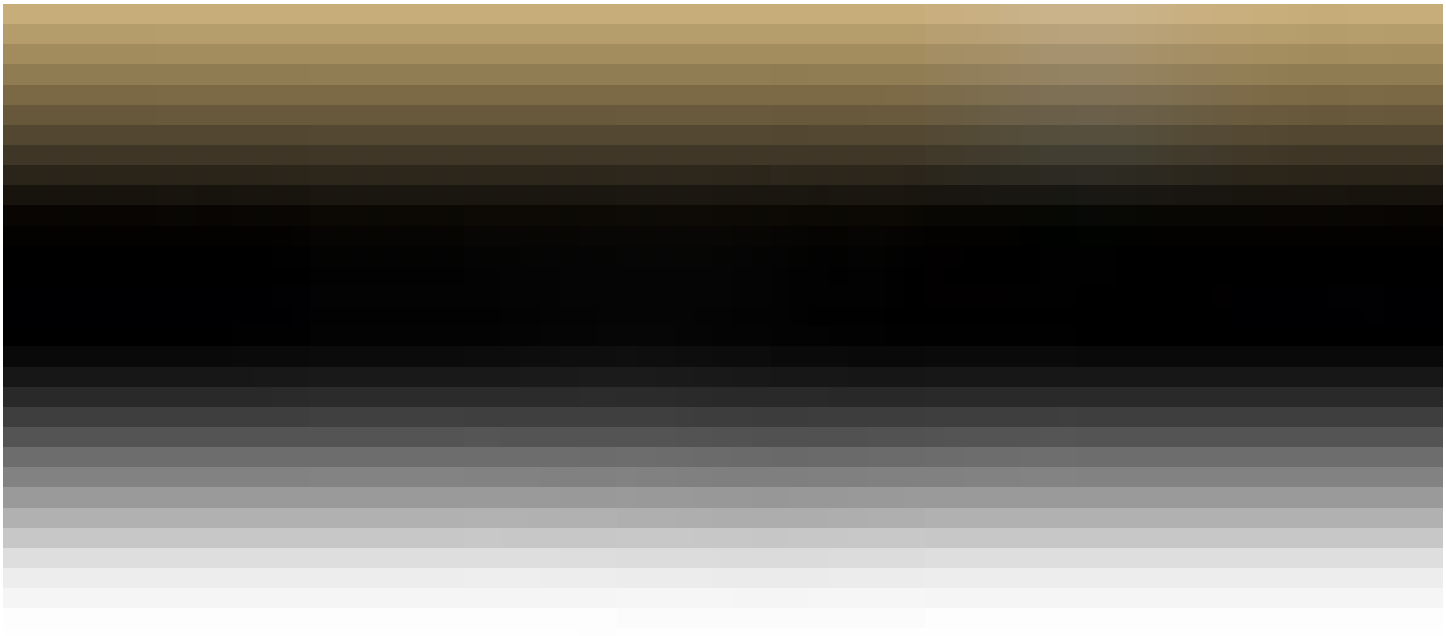
1. Importe o **CUSTOM_ELEMENTS_SCHEMA.**

```
CUSTOM_ELEMENTS_SCHEMA
```

2. Declare em "**@NgModule**" um schema com o valor: "schemas: [CUSTOM_ELEMENTS_SCHEMA]".

```
schemas: [CUSTOM_ELEMENTS_SCHEMA]
```

Veja como ficou nossa aplicação:



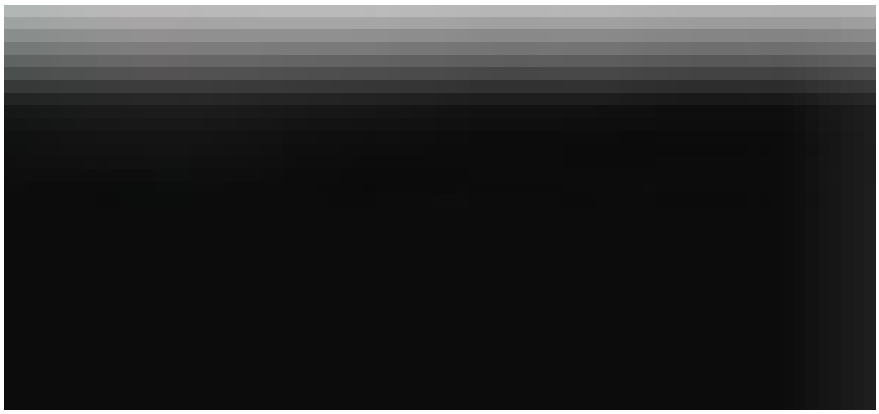
Repare a renderização dos components:

Nosso component content sumiu, mas temos o router-outlet que se encarregará de renderizar quaisquer components que quisermos ali. Sabendo disso, vamos criar a rota default da aplicação.

Criando o App Routing Module

No seu terminal, cole o comando abaixo:

```
ng generate module app-routing — flat — module=app
```



Abra o arquivo **app.module.ts** e importe o routing que acabamos de criar:

import do app-routing.module.ts.

```
import { AppRoutingModule } from ‘./app-routing/app-
routing.module’;
```

declaração do módulo AppRoutingModule no **app.module.ts.**

```
imports: [
...
AppRoutingModule
]
```

Configurando as rotas para navegação

Abra o arquivo **app-routing.module.ts** e atualize com o código abaixo:

import os módulos RouterModule e Routes do pacote
“@angular/router”

```
import { RouterModule, Routes } from ‘@angular/router’;
```

export o RouterModule

```
@NgModule({
..
exports: [RouterModule]
})
```

Criando as rotas da aplicação

Chegou a melhor parte: criar as rotas da nossa aplicação. Primeiro, importe o component que você deseja usar no roteador, em nosso caso o **content.component.ts**

```
import { ContentComponent } from './../content
/content.component';
```

E agora, crie uma contante chamada routes e diga que seu tipo é Routes. Esse cara nada mais é do que um Array de objetos com os principais atributos **“path” (caminho da rota)** e **“component” (component)** que será renderizado quando o path for chamado.

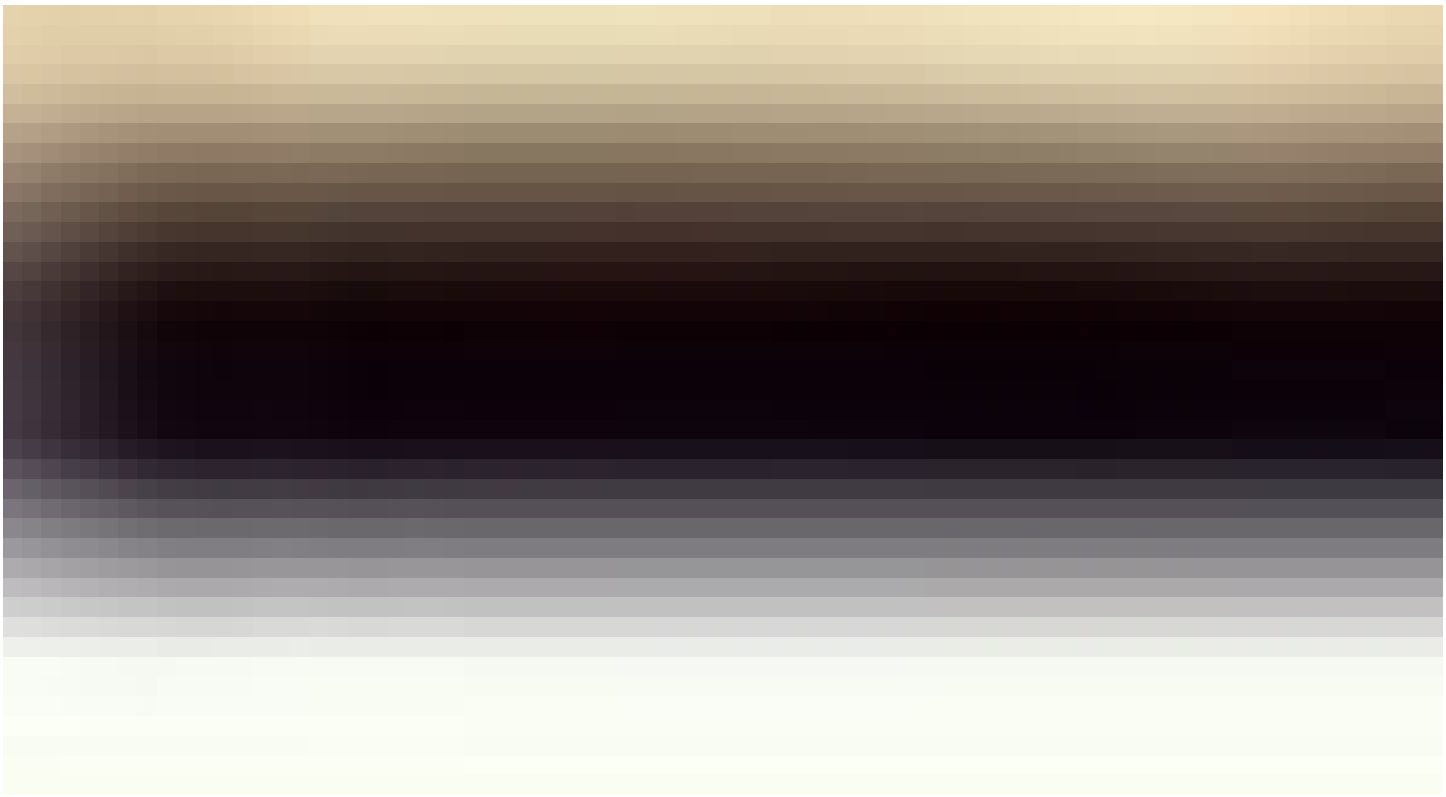
```
const routes: Routes = [
{ path: 'cadastro', component: ContentComponent }
];
```

declare em “@NgModule” o **RouterModule.forRoot(routes)** como abaixo:

```
@NgModule({

imports: [
…
RouterModule.forRoot(routes)
],
});
```

Testando a rota

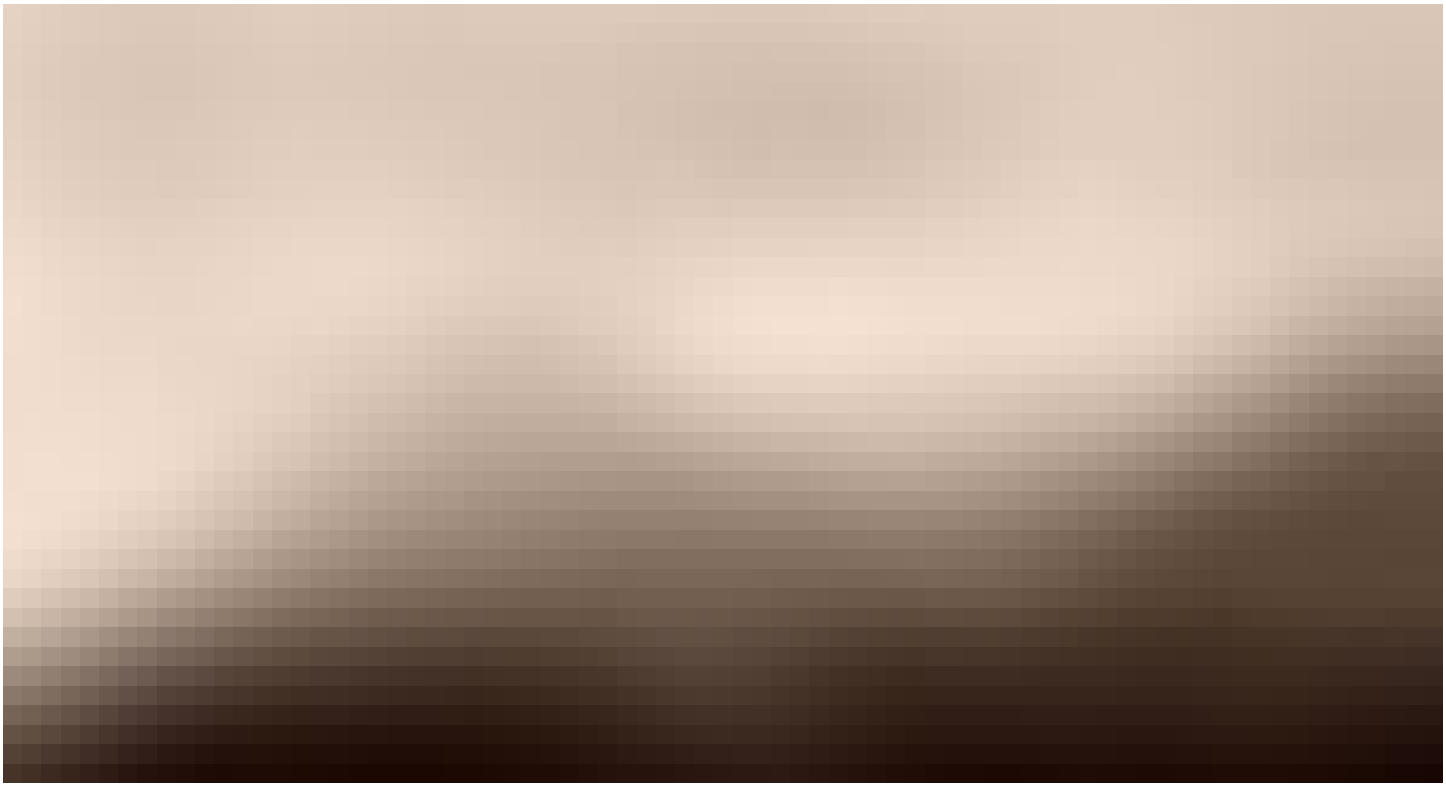
Chegou o momento mais bacana do post. No navegador, acesse o endereço: http://localhost:4300/cadastro e vamos ver no que vai dar.



Sucesso! repare no inspect do navegador que abaixo do router-outlet, temos o o componet **content.** Isso é uma prova de que nosso roteador está funcionando.

Não para por ai..

Vamos voltar para a home da aplicação e veremos no que vai dar:



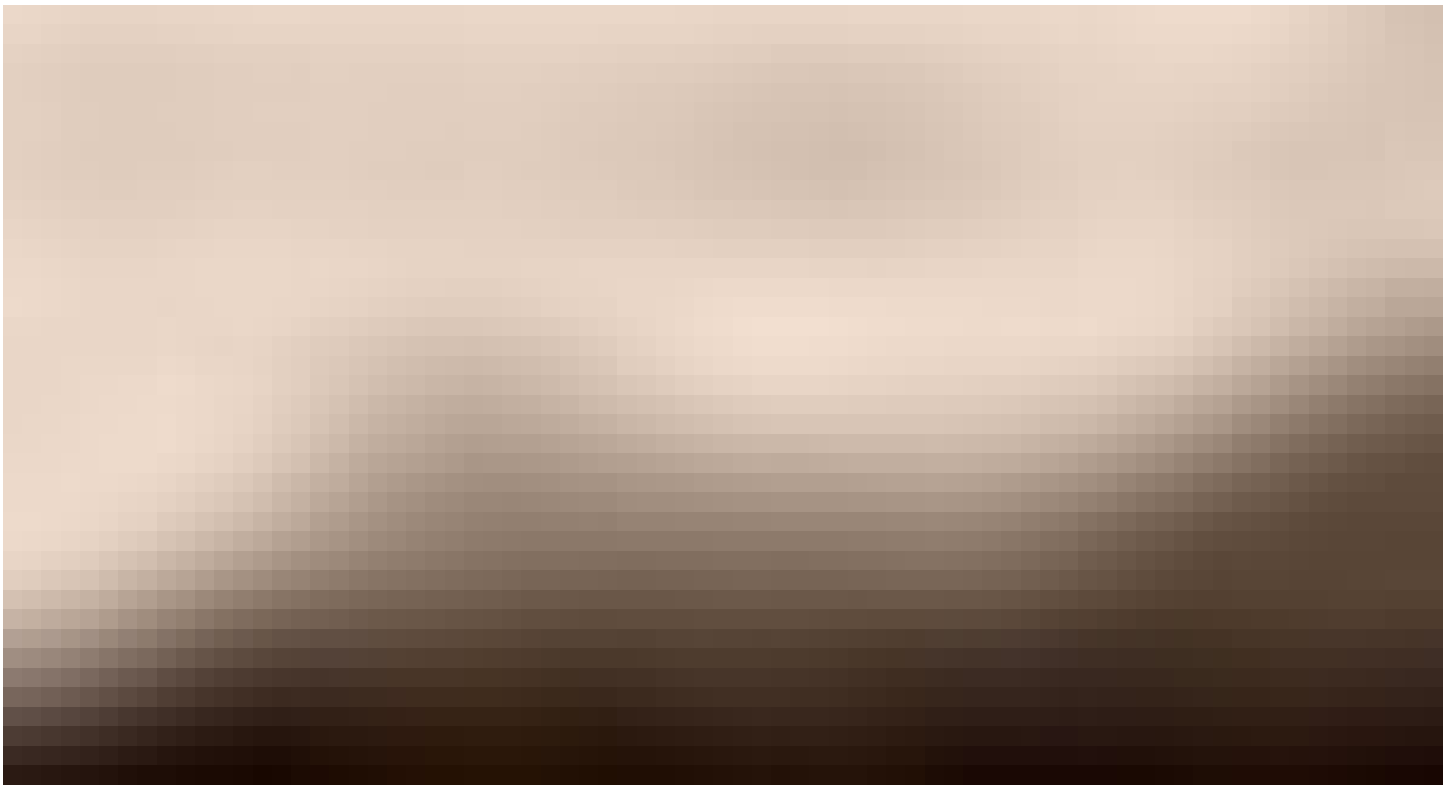
Não se assuste, é simples de entender! Nosso roteador ainda está funcionando mas agora nossa url mudou e está assim: http://localhost:4300. Em nosso arquivo **app-routing.module.ts,** não configuramos nenhum path para essa url.

Criando path para a home

Ainda no arquivo **app-routing.module.ts**, adicione um novo path em branco chamando o component Content. Assim, quando iniciarmos a aplicação, por default já teremos nosso banner.

```
const routes: Routes = [

{ path: '', component: ContentComponent },
{ path: 'cadastro', component: ContentComponent }
];
```

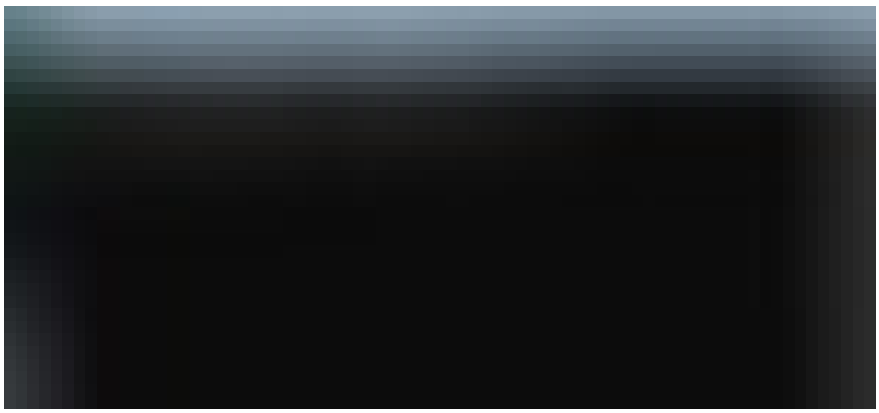
Funcionamento da aplicação:



Cadastro de usuários no Banco

Gere um component para cadastro de novos usuários:

```
ng g c cadastro-clientes
```



Agora, vamos criar a rota de cadastro de clientes. Siga os passos:

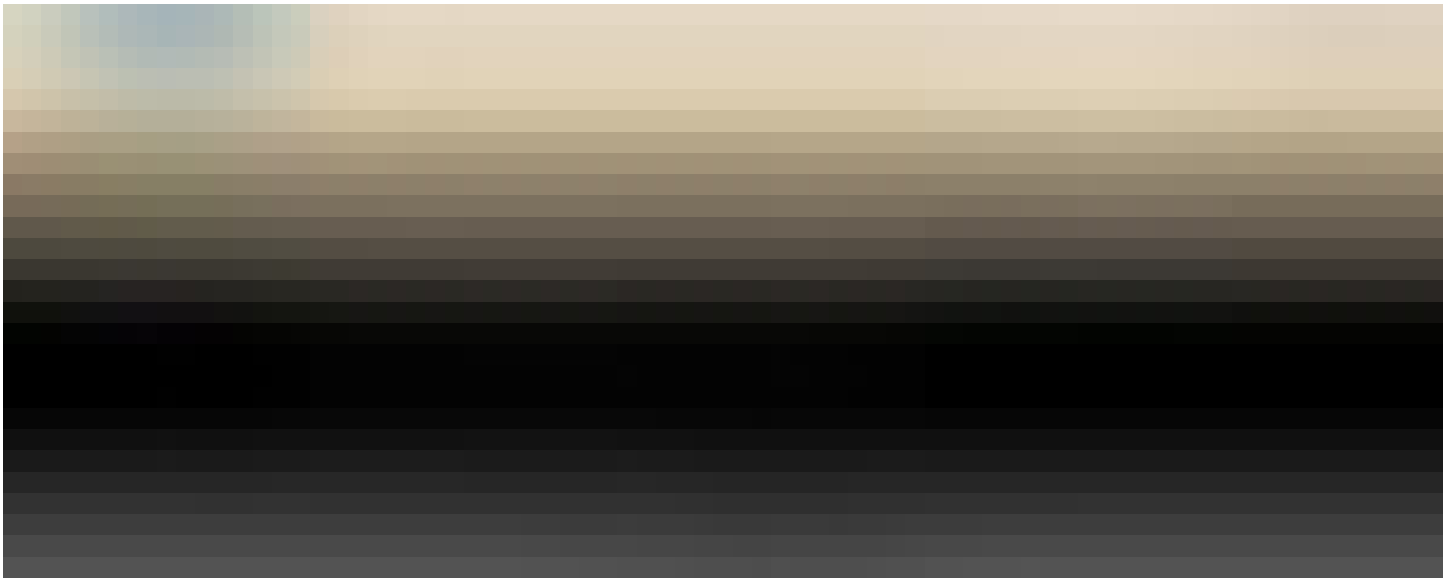
> No arquivo app-rounting.module.ts, importe o component que acabamos de criar

```
import { CadastroClientesComponent } from './../cadastro-
clientes/cadastro-clientes.component';
```

2. Atualize a constante routes com um novo path para o component que acabamos de importar

```
const routes: Routes = [
{ path: '', component: ContentComponent },
{ path: 'cadastro-clientes', component:
CadastroClientesComponent },
];
```

Pouca coisa mudou. Agora, quando acessarmos no nevegador o endereço http://localhost:4300/cadastro-clientes, teremos o resultado:



Atualizando a página de cadastro de clientes

Vamos criar o formulário de cadastro de clientes do banco, como se trata de um projeto de teste. Ignore os dados requeridos na aplicação. rs.

Abra o arquivo **cadastro-clientes.component.html e** atualize com a marcação abaixo:

```
<div class="container">

<div class="row box-cadastro">

<div class="col-12">

<h3 class="text-center">Você está a um passo de criar sua
conta Training Banking</h3>

<form class="mt-5">

<div class="form-group">

<input type="text" class="form-control cadastro"
id="exampleInputEmail1" aria-describedby="emailHelp"
placeholder="Nome">

</div>

<div class="form-group">

<input type="number" class="form-control cadastro"
id="exampleInputPassword1" placeholder="CPF">

</div>

<div class="form-group">

<input type="email" class="form-control cadastro"
id="exampleInputPassword1" placeholder="E-mail">

</div>
```

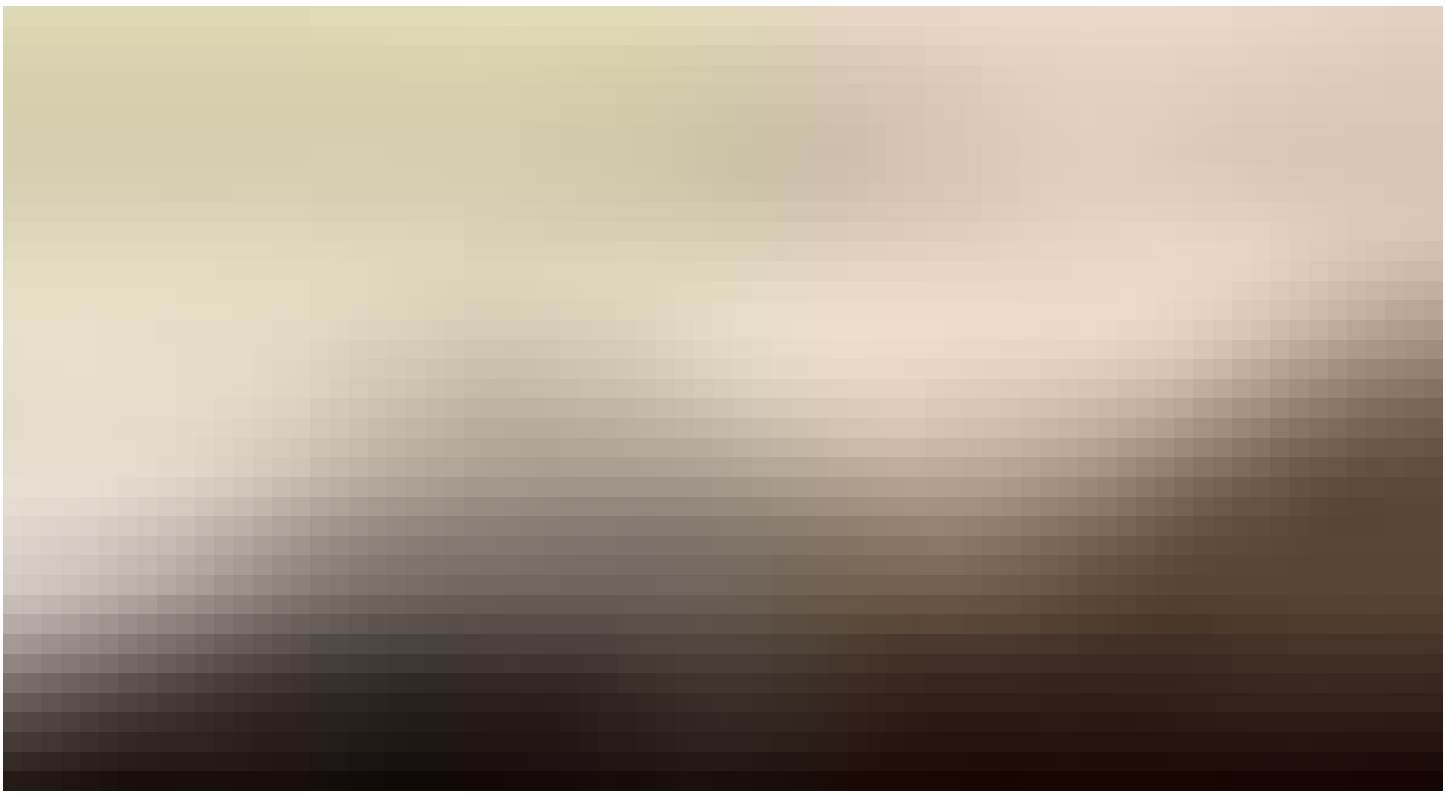
```
<div class="form-group">

<input type="tel" class="form-control cadastro"
id="exampleInputPassword1" placeholder="Telefone">

</div>

<div class="form-group">

<input type="text" class="form-control cadastro"
id="exampleInputPassword1" placeholder="Endereço">

</div>

<button type="submit" class="btn btn-cadastro btn-
lg">Cadastrar</button>

</form>

</div>

</div>

</div>
```

Atualize também o estilo do formulário. Abra o arquivo **cadastro-clientes.component.css**:

```
.box-cadastro {

margin: 30px;

padding: 30px;

}

.cadastro {

padding: 10px;

border: none;
```

```
border-bottom: 2px solid #ddd;

border-radius: 0;

outline: none;

height: 50px;

}

.btn-cadastro {

width: 30%;

display: block;

margin: 0 auto;

padding: 10px;

background-color: #dcbc81;

font-weight: bold;

}
```

Testando a aplicação:



Ótimo, estamos quase finalizando! Agora só precisamos de um botão na home convidando o usuário a criar sua conta. Para isso siga, os passos:

No arquivo **content.component.html**, crie um botão html com um evento de click, disparando um método:

```
<button class="btn btn-home-cadastro-clientes"
(click)="gotoCadastroClientes()">Crie sua conta</button>
```

2. Adicione o estilo do botão de cadastro no arquivo **content.component.css**:

```
.btn-home-cadastro-clientes {

position: absolute !important;

top: 390px;

padding: 20px;
```

```
font-size: 25px;

width: 40%;

background: #000;

color: #fff;

border: 2px solid;

}
```

3. Repare que em nosso botão html no evento de click adicionamos um método chamado **gotoCadastroClientes.** Abra o **content.component.ts** e inicie esse método.

```
gotoCadastroClientes() {
 this.router.navigate(['cadastro-clientes'];
}
```

4. No constructor, declare o router como privado:

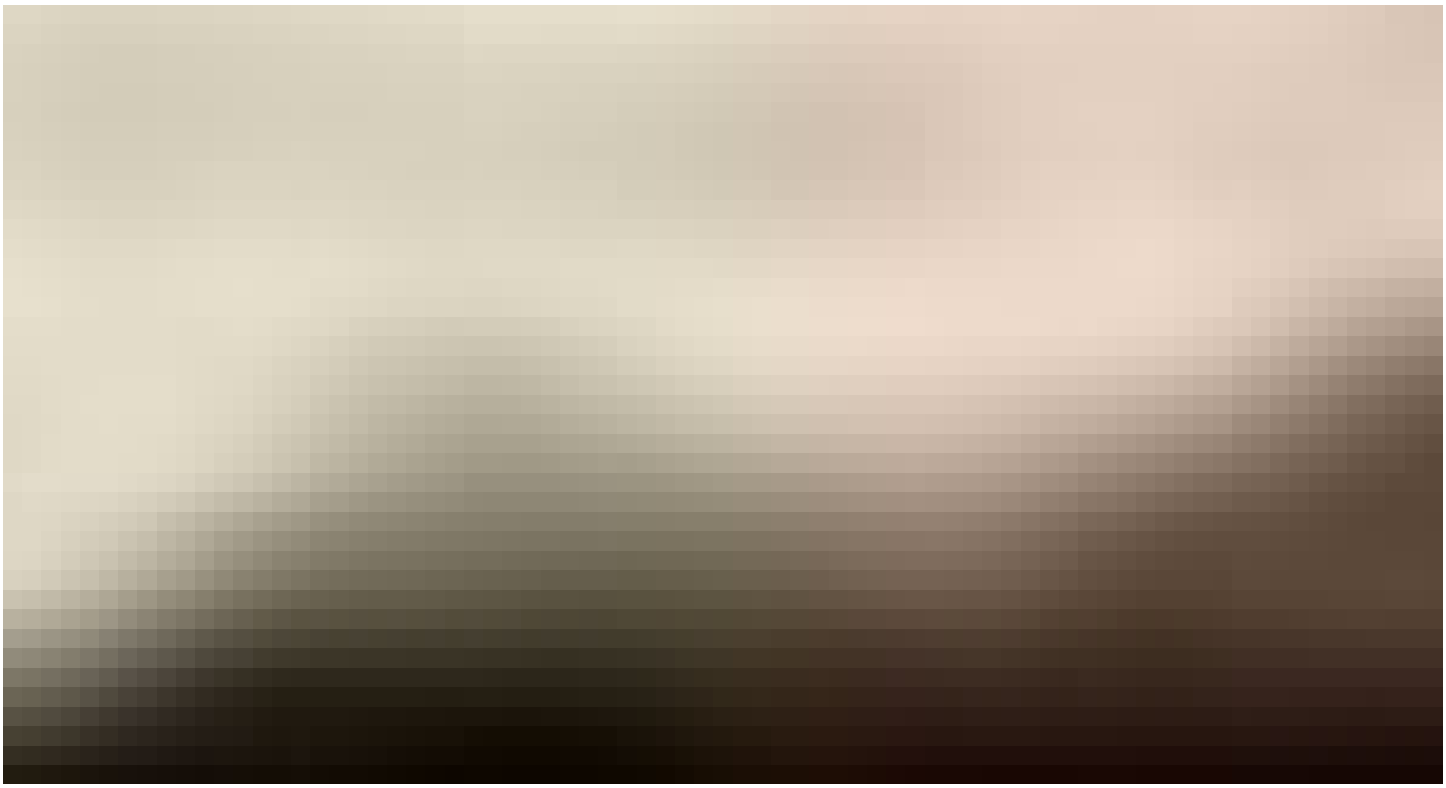
```
constructor(private router: Router) { }
```

5. Importe o Router do pacote “@angular/router”:

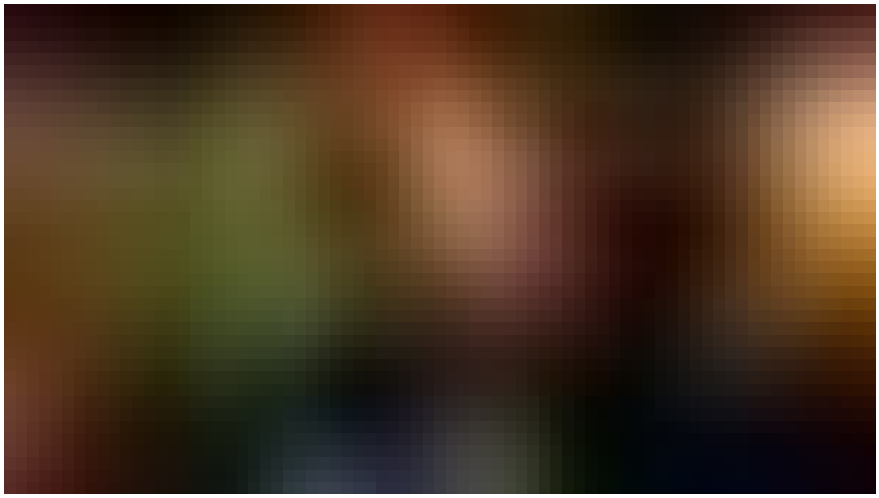
```
import { Router } from '@angular/router';
```

Agora tudo indica que, ao clicarmos no botão “Crie sua conta”, seremos levado à rota **/cadastro-clientes.**

Vamos ver:



Sucesso!



Nosso post vai ficando por aqui. Espero ter ajudado vocês. Nos próximos posts, vamos criar o cadastro dos novos usuários e autenticar a aplicação.

Ficou com alguma dúvida? Deixe nos comentários. Aproveite também pra sugerir features para os próximos posts lá no Twitter. Segue: “@danilodev_silva”.

Grande abraço!