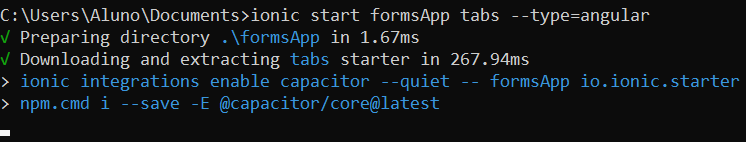
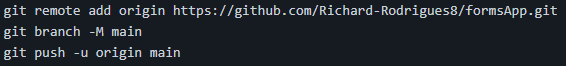
**Nome:** Richard Rodrigues da Silva **Turma:** B

**Documentação FormsApp - P.A.M**

**Link do Projeto em minha página do GitHub:** <https://github.com/souorichard/formsApp>

 Como primeiro passo, iremos criar o nosso projeto chamado **formsApp**, com o seguinte comando no seu terminal (da sua preferência): <**ionic start formsApp tabs –type=angular**>.

 Entraremos na pasta do projeto com o seguinte comando <**cd formsApp**>. Após a criação do projeto, subiremos ele no **GitHub**, mas primeiro precisamos criar um repositório. Logo que criarmos o repositório, usaremos os seguintes comandos para dar o **Initial Commit**:

**\*Obs:** no lugar do nome do branch MAIN, usaremos **MASTER**.

Logo em seguida, dê o seguinte comando para abrir o editor de códigos (VS Code): <**code .**>.

Após abrir o editor de códigos vamos ter a seguinte estrutura, onde entraremos na pasta <**src/app**> para ter acesso à toda nossa aplicação.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

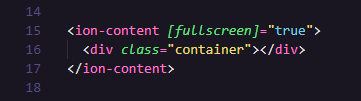
Texto

Descrição gerada automaticamente Agora com tudo pronto, bora codar, primeiramente criaremos a páginas de **LOGIN** e **REGISTRO**. Com o seguinte comando no terminal de sua preferência, execute o comando <**ionic generate page ( nome da página )**>.

Texto

Descrição gerada automaticamente Logo após de ter criado as duas páginas do ionic, abra a pasta **TAB1** para que possa fazer os botões linkados a essas páginas.

A TAG <**ion-buttons**> engloba os botões que iremos usar e adicionaremos um propriedade dentro dessa TAG <**slot = “end”**> para que os botões fiquem no **FINAL** da nossa <**ion-toolbar**>. Com isso, coloque dois <**ion-button**> dentro da TAG pai e dentro de cada botão um <**ion-icon**> como a propriedade <**name = “(nome do ícone)”**>. Agora com os botões criados iremos dar a funcionalidade a um deles para que consiga ser redirecionado para nossa página de **LOGIN**. No botão escolhido coloque a propriedade <**routerLink = “/login”**> com isso assim que clicar no botão vai para a página escolhida.

 Se for de sua escolha coloque a logo da sua impresa com a seguinte TAG <**img**> e coloque o caminho da imagem dentro da propriedade <**src = “( caminho )”**>.

Adicione a seguinte TAG <**div**> com a propriedade <**class = “( nome da classe )”**>.

**- LOGIN**

Texto

Descrição gerada automaticamente Logo após vá para a pasta de **LOGIN** e depois vamos a página de REGISTRO para fazer toda a estrutura **HTML** e depois para a personalização geral.

Aqui será o mesmo processo que fizemos acima na <**ion-toolbar**> da **TAB1**, porém o ícone vai ser uma seta apontada para trás com intuição de voltar a página anterior.

**\*Obs:** ao invés da propriedade <**routerLink**> ter algo após a barra, deixaremos sem nada, pois ela representa o caminho anterior.

Texto

Descrição gerada automaticamente

Dentro do nosso <**ion-content**> iniciaremos uma TAG <**section**> com a seguinte classe chamada também <**class = “container”**>. Logo a seguir uma TAG <**div**> com a classe <**class = “card”**>, ela será o cartão que vai englobar todo o conteúdo de formulário. Dentro, colocaremos outras TAG <**div**> com classes de diferentes nomes:

**\*logo -- \*inputs -- \*btn -- \*others-options**

Na TAG <**div**> com a classe **\*logo** vamos colocar um TAG <**img**> com a logo de sua preferência e o caminho que ela está.

Na TAG <**div**> com a classe **\*inputs** colocaremos o seguinte componente do que iremos pegar no site [Ionic Framework](https://ionicframework.com/).

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Usaremos duas vezes esse componente, uma para EMAIL e o outro para SENHA.

Na TAG <**div**> com a classe **\*btn** vamos udar o componente <**ion-button**> também do Ionic e colocaremos a seguinte classe <**class = “btn”**>.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média

Na TAG <**div**> com a classe **\*others-options** usaremos a TAG <**p**> duas vezes, porém com funcionalidades diferentes.

Na primeira TAG, colocaremos também uma TAG <**a**>que redicionará para a página de **REGISTRO**, com a propriedade <**routerLink = “/register”**>.

Na segunda TAG, **abaixo** na primeira colocaremos uma TAG <**a**>com a mesma propriedade, mas vazia ou com uma **#** que não servirá para nada. Entretanto pode servir para projetos mais avançados com esse protótipo.

**- REGISTRO**

Com a página de **LOGIN** concluída, vamos desenvolver a página de **REGISTRO**. Vá para a pasta e acesse o arquivo **HTML** para fazer toda a estrutura.

Texto

Descrição gerada automaticamente

Aqui iremos basicamente fazer a mesma coisa que a página de **LOGIN**, porém com mais componentes de inputs, e com uma opção a menos na TAG <**div**>com a classe **\*others-options**. Dentro da TAG <**div**> com classe **\*inputs** adicionaremos 5 <**ion-tem**>igual o que usamos na página de **LOGIN**, mas aqui teremos a seguinte ordem:

**\*nome -- \*email -- \*cpf -- \*senha -- \*confirmação**

Como feito na imagem acima, também iremos fazer uma alteção na TAG <**div**>com a classe **\*others-options** ao invéz de duas TAGS <**p**> usaremos somente um com uma TAG <**a**> para redirecionar para a página de **LOGIN** com a propriedade <**routerLink = “/login”**>. Agora com toda a estrutura em **HTML** feita, vamos para a estilização no arquivo **GLOBAL.SCSS** no nivél da pasta **APP**.

**- ESTILIZAÇÃO**

Texto

Descrição gerada automaticamente Agora com toda a estrutura em **HTML** pronta, iremos para a estilização no arquivo **GLOBAL.SCSS**.

Texto

Descrição gerada automaticamente Primeiramente, caso seja de sua prefência, importe uma fonte de seu gosto através do <**@import url()**> disponível apenas nos arquivos de estilização. Logo em seguida definiremos alguns estilos para todas as TAGS disponíveis no projeto através do <**\***>, dentro definiremos o seguinte: <**margin**: 0> <**box-sizing:** border-box> <**font-family:** ‘Poppins’> como feito na imagem acima. Na TAG <**a**> vamos tirar a linha do link que aparece abaixo da escrita com a seguinte propriedade <**text-decoration:** none>. E para finalizar essa parte mudaremos a cor da <**ion-toolbar**> e <**ion-tab-bar**> com a propriedade do próprio Ionic <**--background**> e definir com a cor desejada.

Nessa parte definiremos algumas propriedades para todas as TAGS que tem a classe **\*container** como uma forma de “economia” de código, essa classe é responsável por englobar todo o conteúdo, entretanto, vai ter as seguintes propriedades para deixar como desejado: <**width:** 100%> <**height:** 100%> <**display:** flex> <**justify-content:** center> <**align-items:** center> <**background:** linear-gradient(**\*var da sombra**, **\*var da sombra**,), url(**\*caminho da imagem de fundo desejada**)> <**background-position:** center> <**background-size:** cover>. Com isso centralizará nosso **CARD** que vem a seguir, e com uma imagem de fundo.

Agora para nossa TAG <**div**> com a classe **\*card** utlizaremos as seguintes propriedades: <**width:** 90%> <**padding:** 25px> <**background-color:** definir uma cor de sua preferência> <**border-radius:** 10px>. Nosso **CARD** terá uma largura de 90% do \***container**, com um espaçamento interno de 25px e uma borda arredondada de 10px.

Texto

Descrição gerada automaticamente Já a TAG <**div**> com as classes **\*logo** e **\*inputs** terão uma largura de 100% <**width:** 100%>, porém com **margin-bottom** diferentes uma com 30px e outra com 20px. Agora dentro da <**div**> com a classe **\*inputs** vamos definir algumas propriedades para a TAG do Ionic <**ion-item**>dentre elas são: <**margin-bottom:** 10px> <**--border-radius:** 10px>. Apenas o **último**, por isso a propriedade <**:last-chield**> resetaremos sua margin de baixo para 0px.

Nessa nossa última parte antes de finalizamos o projeto, definiremos algumas propriedades para nossa TAG <**div**> com a classe **\*btn**, para deixar ele mais bonito com um aspecto de botão. Suas propriedades serão: <**margin-top:** 30px> <**--background:** linear-gradient( **\*para a esquerda**, **\*1º cor de sua preferência**, **\*2º cor de sua preferência** )> <**--border-radius:** 8px> <**transition:** 0.5s>. Depois disso utilizaremos a propriedade <**:hover**> para adicionar uma animação assim que passarmos o mouse por cima do botão com um <**letter-spacing:** 5px> para distanciar as letras.

Agora para finalizar, estilizaremos a TAG < **div**> com a classe **\*others-options** com as propriedades: <**width:** 100%> <**margin-top:** 20px> <**display:** flex> <**justify-content:** space-between> <**align-items:** center>. E dentro, temos uma TAG <**p**> com a seguinte propriedade <**font-size:** 0.6rem> que muda o tamanho da font do projeto.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**Placa de letreiro afixada em fachada de loja com cobertura de entrada de estabelecimento

Descrição gerada automaticamenteSE SEGUIU TODOS OS PASSOS O SEU PROJETO DEVERÁ TER FICADO ASSIM (COM AS CORES E LOGO DE SUA PREFERÊNCIA) :**

**- IONIC STORAGE**

Texto

Descrição gerada automaticamente Na pasta **APP**, criaremos uma pasta com nome **MODELS** e logo em seguida cria-se um arquivo com nome **USER.TS**.

Texto

Descrição gerada automaticamente No **REGISTER.PAGE.TS**, vamos igualar as **variáveis** para **passarmos** as **informações** fornecidas pelo usuário cadastrado.

Em **HOME.PAGE.TS** foifeitaacriaçãoda **lista de usuários**, juntamente com o método responsável por realizar a **busca** dos mesmos e em seguida **exibir** **os** **usuários** cadastrados na tela. Por fim cria-se um método para **excluir** **cadastro** caso haja necessidade.

Texto

Descrição gerada automaticamente

Em **HOME.PAGE.HTML**, utiliza-se um <**ion-item-sliding>** para que seja possível realizar todos os métodos criados anteriormente no **HOME.PAGE.TS**, entre eles a **exibição dos usuários cadastrados** e **exclusão de cadastro.**