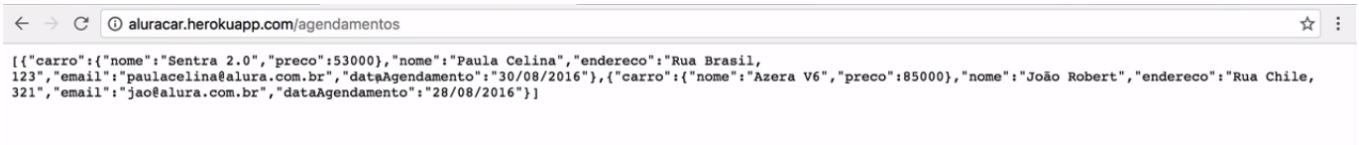
02

# Integrando com o back-end para popular o app

### Transcrição

Agora que temos a "casca" do Front-End da nossa aplicação, podemos começar a colocar inteligencia nela. Pegaremos os dados do servidor e vamos preencher as abas "Agendamentos" e "Fornecedores". Para isto, o nosso cliente nos forneceu a url `aluracar.herokuapp.com/agendamentos` e o `aluracar.herokuapp.com/fornecedores` .

Temos os dados dos agendamentos, com nomes, endereços, e-mail, data do agendamento, modelo do carro e preço.




Nós precisamos destes dados na aplicação.

A seguir, iremos desenvolver a outra app. Até aqui, o arquivo `agendamentos.html` deste projeto estará assim:

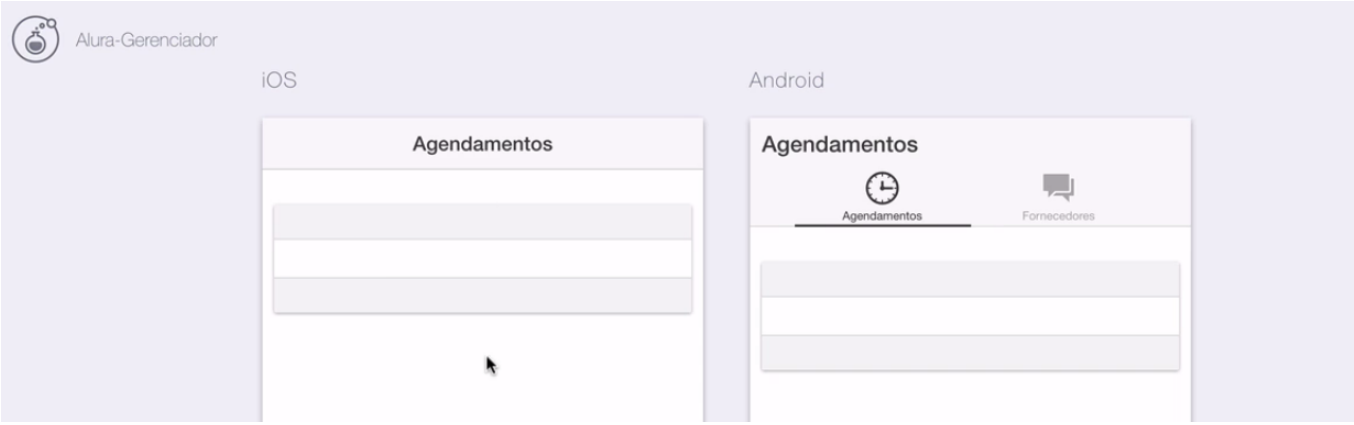
```
<ion-view title="Agendamentos" id="page2">
    <ion-content padding="true" class="has-header">

    </ion-content>
</ion-view>
```

Temos a estrutura básica e podemos adicionar o conteúdo que quisermos. Adicionaremos o componente `card` . Começaremos pelas `<div>` s e depois, colocaremos o CSS ( `class` ). Tanto no cabeçalho como no rodapé teremos `item item-divider` .

```
<ion-view title="Agendamentos" id="page2">
    <ion-content padding="true" class="has-header">
        <div class="card">
          <div class="item item-divider"></div>
          <div class="item item-text-wrap"></div>
          <div class="item item-divider"></div>
        </div>
    </ion-content>
</ion-view>
```

Vamos no browser ver como ficou o componente.




Agora precisamos popular o cartão e repeti-lo de acordo com os itens que temos no servidor. Criamos a View, podemos ir para o `controllers.js` :

Ele criou um arquivo um pouco diferente. Em vez criar a partir um modulo principal como fizemos anteriormente, ele construiu um módulo para cada funcionalidade. Nós vamos seguir o padrão que temos usado:

```
angular.module('app.controllers', [])

.controller('AgendamentoController', function(){

})
```

No segundo parâmetro, usamos uma `function` . Também adicionamos um `controller` , `AgendamentoController()` .

```
.controller('FornecedorController', function(){

})
```

Como fizemos as alterações, teremos que modificar o arquivo de rotas. No `routes.js` , adicionaremos primeiro o `AgendamentoController` :

```
.state('tabsController.agendamentos', {
url: '/agendamentos',
views: {
  'tab1': {
    templateUrl: 'templates/agendamentos.html',
    controller: 'AgendamentoController'
    }
  }
})
```

Depois, adicionaremos `FornecedorController` :

```
.state('tabsController.fornecedores', {
  url: '/fornecedores',
  views: {
    'tab2': {
      templateUrl: 'templates/fornecedores.html',
      controller: 'FornecedorController'
```

```
    }
  }
})
```

Vamos voltar para o `controllers.js`, adicionaremos uma variável que receba um *array* na View. Vamos usar a variável `$scope.agendamentos`

```
angular.module('app.controllers', [])

.controller('AgendamentoController', function($scope){
    $scope.agendamentos = [];


})
```

Precisaremos de um `Service` para fazer a requisição GET e obter os dados. Faremos isto no `services.js` e também modificaremos o arquivo que foi criado:

```
angular.module('app.services', [])
.service('GerenciadorService', function($http){

    var url = "http://aluracar.herokuapp.com";

    return {
        obterAgendamentos : function(){
            return $http.get(url + "/agendamentos").then(function(response){
                return response.data;
            })
        },
```

Alteramos o nome de `BlankService` para `GerenciadorService`. Depois começamos a fazer a customização pegando a `url`. Nós retornaremos uma chamada HTTP e por isso, injetamos o `$http` na `function`. No `return` fizemos uma chamada `get` e passamos a `url` concatenando com o `/agendamentos`. E no fim, retornamos o `response.data`.

Agora, já podemos usar o `GerenciadorService` no arquivo `controller.js`:

```
angular.module('app.controllers', [])

.controller('AgendamentoController', function($scope, GerenciadorService){

    $scope.agendamentos = [];

    GerenciadorService.obterAgendamentos().then(function(dados){
        $scope.agendamentos = dados;
    })
```

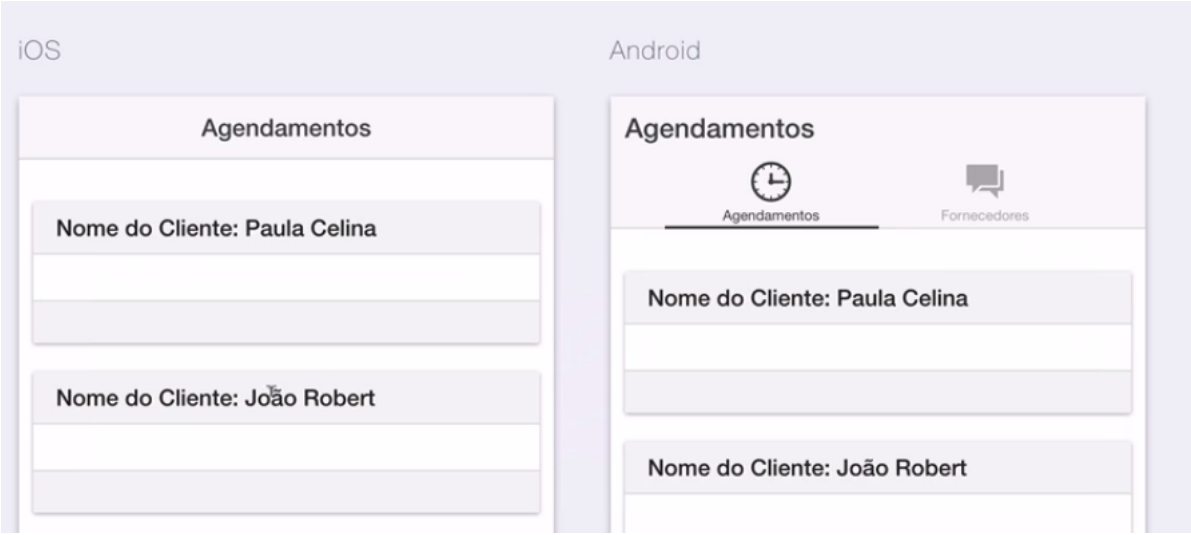
Pegamos a variável recém criada `agendamentos`, que receberá os `dados`.

Precisamos agora apresentar os dados. Em `agendamentos.html`, temos a lista de agendamento. Vamos agora percorrê-la. E nós repetiremos o cartão em cada uma das vezes.

```
<ion-view title="Agendamentos" id="page2">
    <ion-content padding="true" class="has-header">
        <div class="card" ng-repeat="agendamento in agendamentos" >
            <div class="item item-divider">Nome do Cliente: {{agendamento.nome}} </div>
            <div class="item item-text-wrap"></div>
            <div class="item item-divider"></div>

        </div>
    </ion-content>
</ion-view>
```

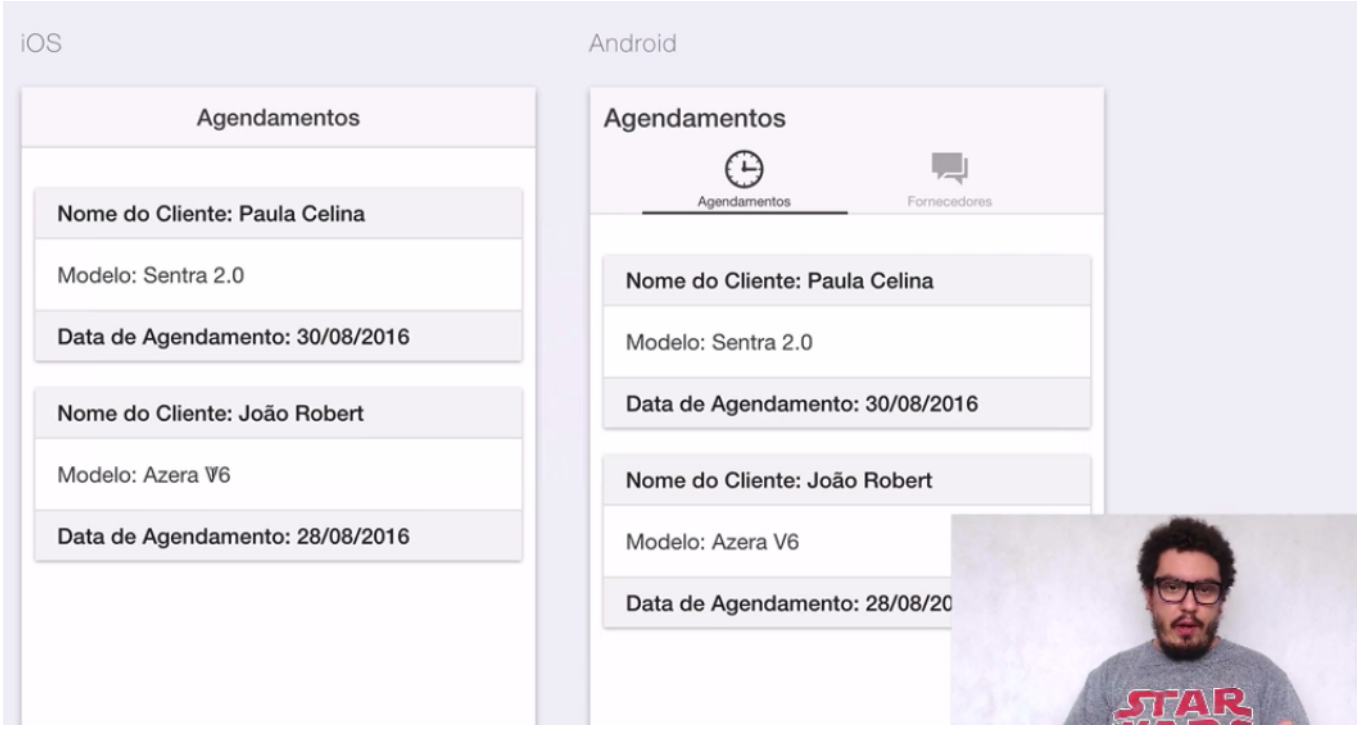
Vamos testar e ver como ficou:




Em seguida, adicionaremos o modelo e no rodapé, adicionaremos a data.

```
<ion-view title="Agendamentos" id="page2">
    <ion-content padding="true" class="has-header">
        <div class="card" ng-repeat="agendamento in agendamentos" >
            <div class="item item-divider">Nome do Cliente: {{agendamento.nome}} </div>
            <div class="item item-text-wrap">
            Modelo: {{agendamento.carro.nome}}
        </div>
            <div class="item item-divider">Data de Agendamento: {{agendamento.dataAgendamento}}</div
        </div>
    </ion-content>
</ion-view>
```

Após salvarmos as alterações, vamos testar:




Perfeito, construímos toda a inteligência do conteúdo!