Sistema de Restaurantes

**Instruções:**

A avaliação é a construção de um sistema simples de restaurante. Favor se atentar as seguintes tecnologias

**Tecnologia**: Plataforma .NET, linguagem C#

**BackEnd**: WebAPI, REST/JSON; Persistência; EntityFramework Code First - Migration

**FrontEnd**: Angular 2.0/4.0 (Two Way Databinding)

**IMPORTANTE:** Utilizar **Solutions separadas** entre o projeto do **BackEnd**e o projeto do **FrontEnd.**

**OBS1**: Utilizar o Entity Framework no padrão Code First Migration, ou seja, O Code First é uma funcionalidade incluída desde a versão 4.1 do Entity Framework. Ele permite o mapeamento/persistência de classes POCO e geração de base de dados a partir da estrutura dessas classes. NÃO utilizar o mapeamento direto das tabelas do banco de dados.

**OBS2** : Utilizar Angular 2.0 para o FrontEnd (NÃO utilizar o padrão RAZOR), aplicando Two Way Databinding no Angular.

**Para conhecimento da arquitetura, segue abaixo alguns links para estudo:**

[**http://loiane.training/curso/angular-2/**](http://loiane.training/curso/angular-2/)

[**https://www.udemy.com/aprenda-angular-2-com-typescript/**](https://www.udemy.com/aprenda-angular-2-com-typescript/)

[**https://ievangelistblog.wordpress.com/2016/01/13/building-an-angular2-spa-with-asp-net-5-mvc-6-web-api-2-and-typescript-1-7-5/**](https://ievangelistblog.wordpress.com/2016/01/13/building-an-angular2-spa-with-asp-net-5-mvc-6-web-api-2-and-typescript-1-7-5/)

**[https://code.msdn.microsoft.com/ASPNET-Core-Angular-2-EF-8fbfae0c#content](https://code.msdn.microsoft.com/ASPNET-Core-Angular-2-EF-8fbfae0c" \l "content)**

[**https://www.codeproject.com/Articles/1151474/CRUD-in-ASP-NETCore-MVC-with-Angular-and-Web-API**](https://www.codeproject.com/Articles/1151474/CRUD-in-ASP-NETCore-MVC-with-Angular-and-Web-API)

[**https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api**](https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api)

[**https://code.visualstudio.com/**](https://code.visualstudio.com/)

**\*\*Enviar o Código-Fonte pelo GitHub \*\***

O intuito do sistema é manter atualizados os registros de restaurantes e os pratos servidos pelos mesmos. Para isso deverão ser criados dois cadastros que permitirão tais tarefas.

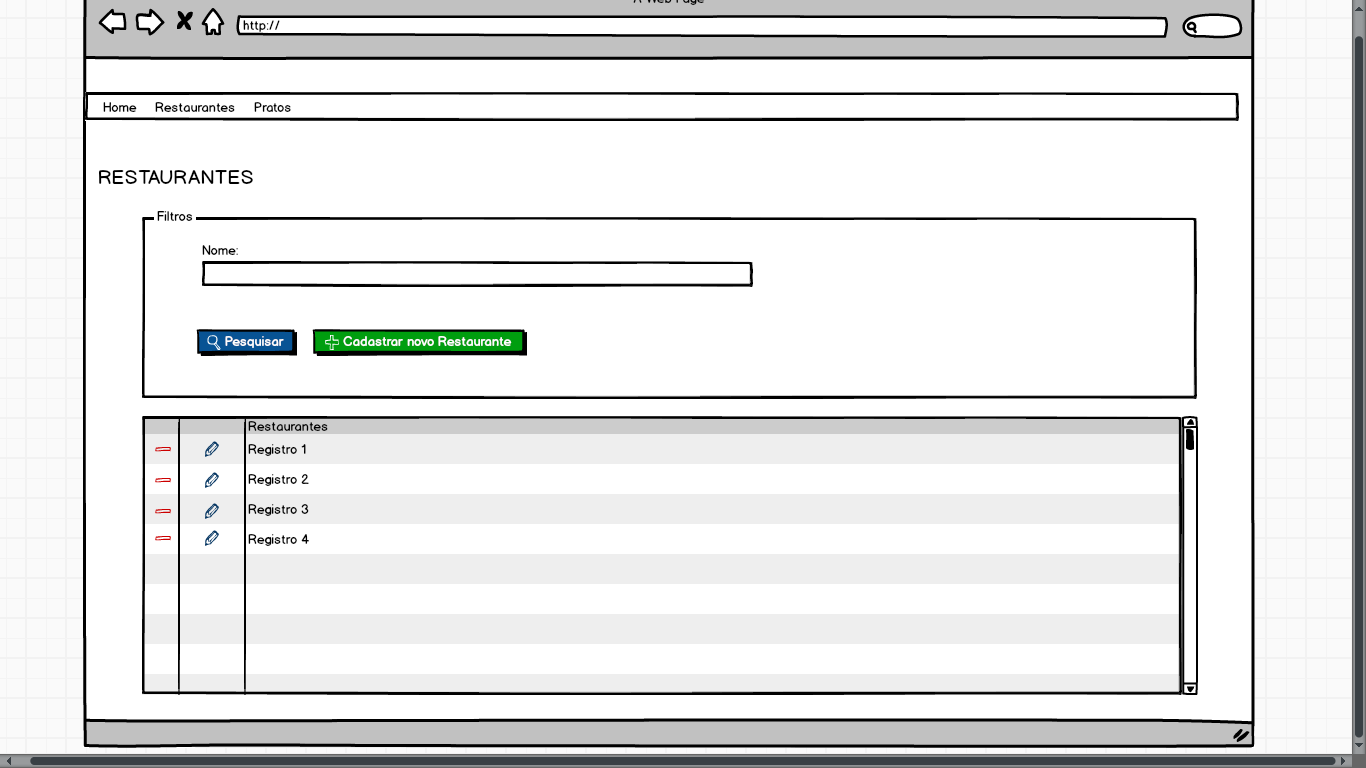
Cadastro de Restaurantes: Esse cadastro irá manter os registros relacionados aos restaurantes.

Cadastro de Pratos: Esse cadastro irá manter os registros relacionados aos pratos.

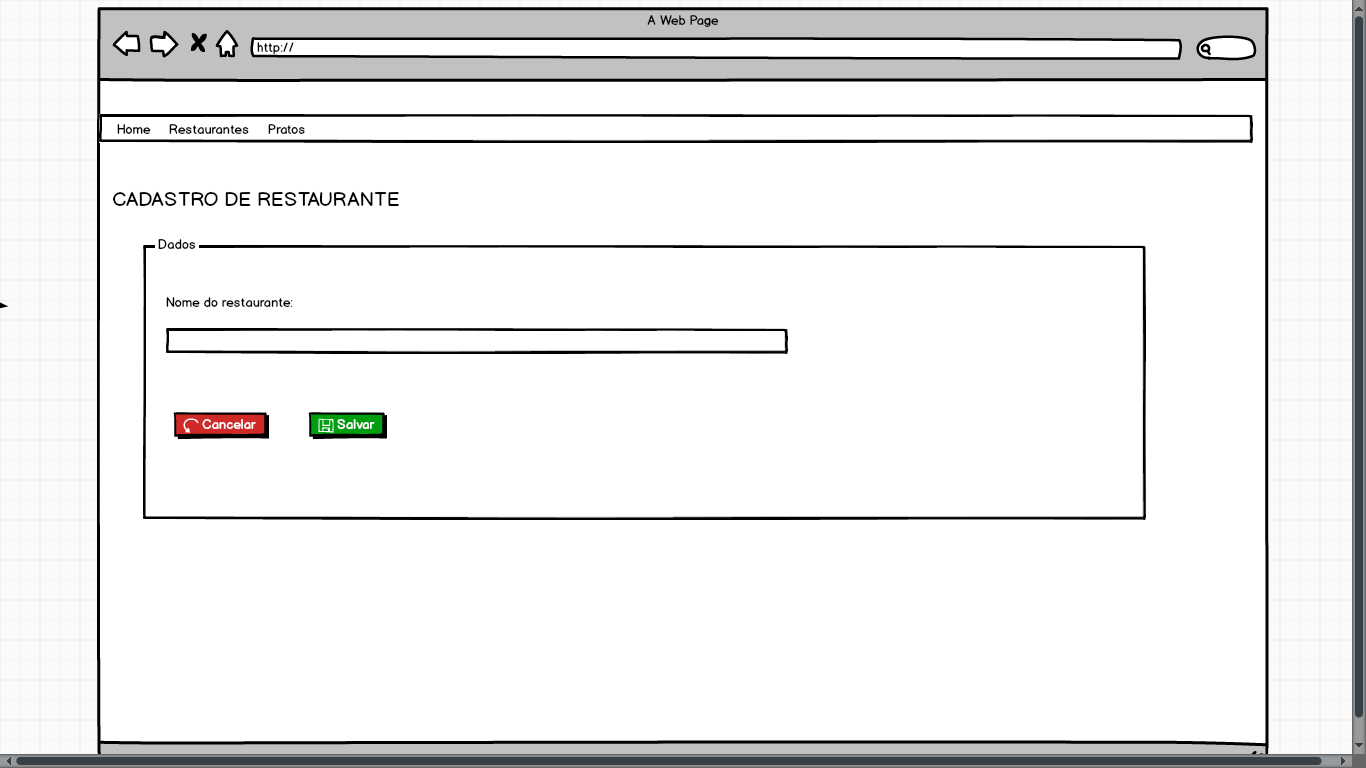
Requisitos funcionais do sistema

|  |  |  |
| --- | --- | --- |
| **Requisito** | **Nome do requisito** | **Descrição** |
| RT010 | Editar Prato | Quando um usuário clicar no botão Editar(apresentado por um lápis azul) na tela de consulta de pratos(Tela3) o sistema irá apresentar a tela de cadastro de pratos(Tela 4) com todos os campos preenchidos possibilitando a edição dos dados pelo usuário. |
| RT011 | Excluir Prato | Quando um usuário clicar no botão Excluir(apresentado por um traço vermelho) na tela de consulta de pratos(Tela 3) o sistema irá excluir um registro de prato. A tela deverá ser atualizada e a linha excluída deverá desaparecer. |
| RT014 | Salvar Prato | Ao clicar no botão salvar (Tela 4) o sistema deverá incluir o registro de um restaurante no banco de dados ou atualizar um registro caso este já exista. Ao finalizar a atualização ou cadastro do registro o sistema exibe a tela de consulta(Tela 3) para o usuário. |

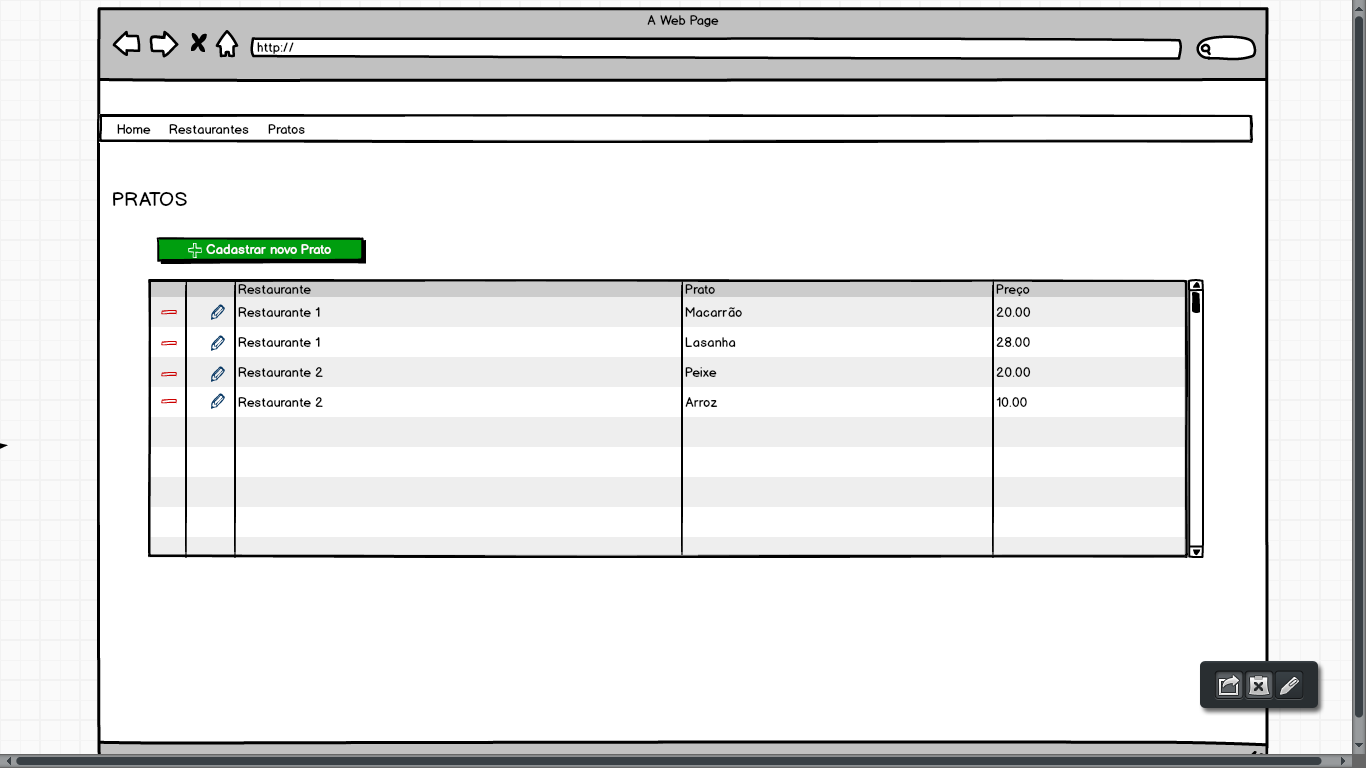
Tela 1 : Consulta de Restaurantes



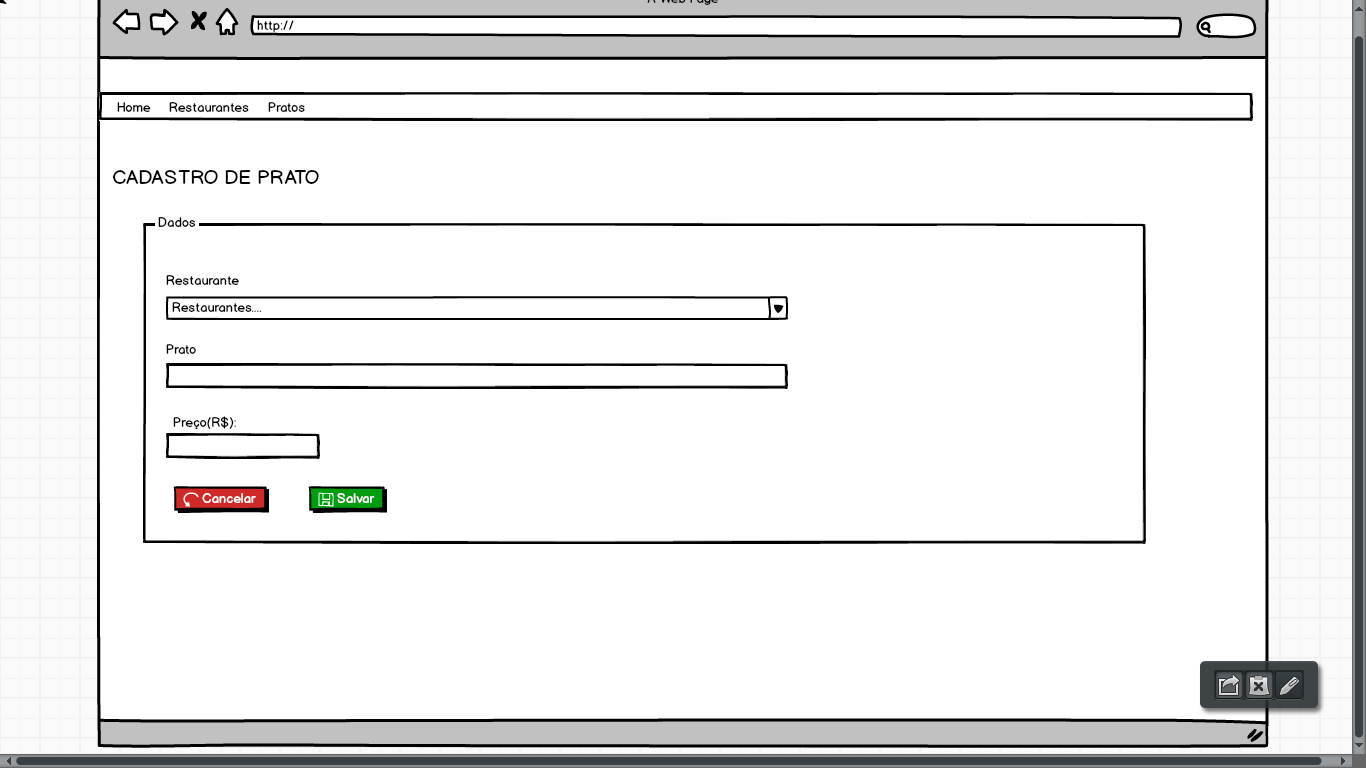
Tela 2: Cadastro de Restaurantes



Tela 3: Consulta de Pratos



Tela 4: Cadastro de pratos



Tela 5: Tela inicial

