**UNIVERSIDADE PAULISTA - UNIP**

GABRIEL SANABRIA FERRERI

PEDRO MIURA SUGAYAMA

LEONARDO CARPEJANI PRESTES

ERICK EIJI KANAYAMA NAGANO

**Aplicativo de gerência e gestão de relacionamentos humanos em ambiente Web**

SÃO PAULO – SP

2019

**UNIVERSIDADE PAULISTA - UNIP**

GABRIEL SANABRIA FERRERI

PEDRO MIURA SUGAYAMA

LEONARDO CARPEJANI PRESTES

ERICK EIJI KANAYAMA NAGANO

**Aplicativo de gerência e gestão de relacionamentos humanos em ambiente Web**

Trabalho de conclusão de curso como parte do exigido para obtenção do título de graduação em Ciências da computação apresentado à Universidade Paulista

Orientador: Luiz Carlos Magrini

SÃO PAULO – SP

2019

Resumo

O uso de ferramentas web no mercado vem se tornando cada vez mais comum, qualquer empresa da atualidade possui seu próprio sistema de gestão ou contabilidade. Algumas dessas ferramentas estão desenvolvidas em desktop, fazendo com que o aplicativo careça de interoperabilidade ou manutenibilidade. Com o intuito de atingir esses aspectos de suma importância, foi desenvolvido uma ferramenta web para gestão de pessoas. Para alcançar o que foi proposto, foram utilizadas ferramentas para o desenvolvimento web implementando a arquitetura MVC (Model View Controller).

Palavras Chave: Interoperabilidade, manutenibilidade

**Lista de Figuras**

[Figura 1 - Modelo MVC 13](#_Toc24228806)

[Figura 2 - Modelo de Data Annotation 14](#_Toc24228807)

[Figura 3 - Modelo de requisição de uma API 16](#_Toc24228808)

[Figura 4 - Diagrama de banco de dados 19](#_Toc24228809)

[Figura 5 - Exemplo divisão em 3 camadas 21](#_Toc24228810)

[Figura 6 - Exemplo de Query à base 21](#_Toc24228811)

[Figura 7 - Método de criptografia utilizado 23](#_Toc24228812)

[Figura 8 - Exemplificação divisão em camadas 24](#_Toc24228813)

[Figura 9 - Diagrama Base de dados do sistema 25](#_Toc24228814)

[Figura 10 - Comparação base de dados com entidade 26](#_Toc24228815)

[Figura 11 - Exemplo de uso do Dapper 27](#_Toc24228816)

[Figura 12 - Exemplo de validação no Login 28](#_Toc24228817)

[Figura 13 - Uso de data binding Model View 29](#_Toc24228818)

[Figura 14 - Tela de Login do sistema 30](#_Toc24228819)

[Figura 15 - Home page 31](#_Toc24228820)

[Figura 16 - Tela Inicial Avaliação 32](#_Toc24228821)

[Figura 17 - Tela de avaliação com campos preenchidos 32](#_Toc24228822)

[Figura 18 - Fim do fluxo avaliação 33](#_Toc24228823)

[Figura 19 - Início fluxo de desligamento 33](#_Toc24228824)

[Figura 20 - Código método de envio de e-mail 34](#_Toc24228825)

[Figura 21 - Tela de edição 34](#_Toc24228826)

[Figura 22 - Validação cadastro novo usuário 35](#_Toc24228827)

[Figura 23 - Preenchimento automático pela API 35](#_Toc24228828)

[Figura 24 - Cadastro Finalizado 36](#_Toc24228829)

[Figura 25 - Tela de relatório 36](#_Toc24228830)

[Figura 26 - Casos de uso 42](#_Toc24228831)

[Figura 27 - Diagrama de classes 43](#_Toc24228832)

**Sumário**

[**1.** **INTRODUÇÃO** 12](#_Toc24244148)

[1.1. OBJETIVOS 13](#_Toc24244149)

[**1.1.1.** **OBJETIVOS ESPECÍFICOS** 13](#_Toc24244150)

[1.1.2. METODOLOGIA 13](#_Toc24244151)

[**2.** **FUNDAMENTAÇÃO TEÓRICA** 14](#_Toc24244152)

[2.1. MVC (MODEL, VIEW, CONTROLLER) 14](#_Toc24244153)

[2.1.1. MODEL 15](#_Toc24244154)

[2.1.2. VIEW 15](#_Toc24244155)

[2.1.3. CONTROLLER 16](#_Toc24244156)

[**2.2.** **API** 17](#_Toc24244157)

[**2.3.** **BASE DE DADOS** 18](#_Toc24244158)

[**2.4.** **Divisão em 3 camadas** 20](#_Toc24244159)

[**2.5.** **CRIPTOGRAFIA** 22](#_Toc24244160)

[2.5.1. HASH 22](#_Toc24244161)

[2.5.2. UTILIZAÇÂO 23](#_Toc24244162)

[**3.** **DESENVOLVIMENTO** 24](#_Toc24244163)

[**3.1.** **CRIAÇÃO DO PROJETO** 24](#_Toc24244164)

[3.1.1. PROJETO EM CAMADAS 25](#_Toc24244165)

[3.1.2. DIAGRAMA DA BASE DE DADOS 26](#_Toc24244166)

[3.1.3. CONSTRUÇÃO ENTIDADE 27](#_Toc24244167)

[3.1.4. CAMADA DE ACESSO A DADOS 28](#_Toc24244168)

[3.1.5. CAMADA DE NEGÓCIOS 29](#_Toc24244169)

[3.1.6. CONSTRUÇÃO DO WEB MVC 30](#_Toc24244170)

[**3.2.** **FLUXO WEB** 31](#_Toc24244171)

[3.2.1. FLUXO AVALIAÇÃO 33](#_Toc24244172)

[3.2.2. FLUXO DESLIGAMENTO 34](#_Toc24244173)

[3.2.3. FLUXO DE EDIÇÃO DE COLABORADOR 35](#_Toc24244174)

[3.2.4. FLUXO DE CADASTRO 36](#_Toc24244175)

[3.2.5. FLUXO DE RELATORIO 37](#_Toc24244176)

[**4.** **ELICITAÇÃO FICTÍCIA DO SISTEMA** 38](#_Toc24244177)

[**4.1.** **ELICITAÇÃO** 38](#_Toc24244178)

[**4.2.** **ESPECIFICAÇÃO DO SOFTWARE** 40](#_Toc24244179)

[4.2.1. INTERFACE DO APLICATIVO 40](#_Toc24244180)

[4.2.2. FUNÇÕES DO SISTEMA 41](#_Toc24244181)

[4.2.3. RESTRIÇÕES DE AMBIENTE 42](#_Toc24244182)

[4.2.4. DESCRIÇÃO DE USUÁRIOS 42](#_Toc24244183)

[**4.3.** **MODELAGEM DO SISTEMA** 43](#_Toc24244184)

[4.4. PLANOS DE TESTE 44](#_Toc24244185)

[**5.** **DISCUSSÃO** 48](#_Toc24244186)

[5.1. FINALIDADES E VANTAGES 48](#_Toc24244187)

[5.1.1. FACIL ACESSO 48](#_Toc24244188)

[5.1.2. FÁCIL IMPLEMENTAÇÃO 49](#_Toc24244189)

[5.1.3. SEPARAÇÃO EM CAMADAS 49](#_Toc24244190)

[6. CONCLUSÃO 50](#_Toc24244191)

[**7.** **REFERÊNCIAS BIBLIOGRÁFICAS** 51](#_Toc24244192)

[8. APÊNDICE – CÓDIGO DO PROGRAMA 54](#_Toc24244193)

[8.1. Entidades 54](#_Toc24244194)

[8.2. Data Access Object 57](#_Toc24244195)

[8.3. Business 61](#_Toc24244196)

[8.4. Util 69](#_Toc24244197)

[8.5. Projeto WEB 70](#_Toc24244198)

[9. APÊNDICE – BASE DE DADOS 109](#_Toc24244199)

# **INTRODUÇÃO**

Com o advento de novas tecnologias, tornou-se cada vez mais difícil manter sistemas de nível legado em dia. As dificuldades recorrentes de se implementar novas funcionalidades, assim como desenvolver novas características ao programa vinham se tornando uma tarefa cada vez mais árdua e refinada. Tornou-se habitual enfrentar problemas graças à implementação de um novo componente desenvolvido. Temos como ênfase o fato de que no desenvolvimento web “os funcionários pode passar menos tempo efetuando manutenções e mais tempo se dedicando a tarefas como marketing, teste[...]” (BARRON, 2018)

Para suprir essa necessidade de atualização, de interoperabilidade e manutenibilidade, sistemas Web se destoaram como a chave e resposta para diversos problemas. Com o sistema web, é possível manter constantes atualizações, em conjunto com a capacidade de ter acesso em qualquer localidade. O maior desafio então tornou-se a forma de desenvolvimento desses ambientes. Para solucionar esse desafio, foi necessária a implementação de novos padrões de arquitetura, o que transformou o desenvolvimento desses sistemas para algo mais facilitado e segmentado. Um exemplo clássico disso é o modelo **MVC** (Model, View,Controller), que segmenta os projetos em 3 camadas para maior facilidade. Também é tido como destaque, o modelo **MVVM**(Model-View-ViewModel). Ademais, pode ser citada a utilização de outros recursos como APIs, que podem funcionar como uma camada extra de busca de dados em qualquer projeto, uma vez que as APIs consistem em efetuar consultas, atualizações, inserções e deleções dentro de um determinado sistema. A utilização de arquiteturas como as supracitadas, torna o sistema mais robusto, com maior capacidade de atualização e manutenção ou desenvolvimento de seus novos componentes.

Visando demonstrar da capacidade desse tipo de aplicativo e desses conceitos, foi desenvolvido um aplicativo que uniria todas as características e facilidades de um ambiente portátil. Um aplicativo cujo intuito é a resolução de problemas com a gestão de pessoas dentro de uma determinada empresa.

# OBJETIVOS

O trabalho apresentado tem como função demonstrar a elaboração de um sistema baseado em ambiente WEB para controle de fluxo de pessoas de uma microempresa, tendo como principal usuário apenas gestores ou funcionários de cargo superior, de uma específica área, além de oferecer um básico *overview* do desempenho da área e de seus colaboradores, em conjunto com gráficos e avaliações cedidas pelo gestor do mesmo dentro da plataforma.

# **OBJETIVOS ESPECÍFICOS**

* Desenvolver um sistema de gerência completo de uma área microempresarial
* Realizar a análise da arquitetura MVC contida no projeto
* Demonstrar a facilidade da utilização de frameworks para acesso à camada de dados
* Efetuar a implementação do sistema em um ambiente de Produção

# METODOLOGIA

Como a natureza do problema estudado é a falta de interoperabilidade e manutenibilidade, seja de sistemas legados ou sistemas desktop, foi efetuado um estudo que teve como base a leitura de código de ambas as ferramentas desenvolvidas para esse ambiente.

Dado o conhecimento adquirido, foi estudado o conceito de MVC(Model View Controller) e então inserido em um ambiente WEB, utilizando a ferramenta Visual Studio.

Como foi necessário dinamismo dos dados, foi estudada e planejada a construção de uma base de dados relacional SQL, e para seu manuseio foi utilizado o Microsoft Sql Server Management Studio. As consultas em base não foram feitas de forma completamente segura para demonstração apenas

Enfim, todas as devidas partes foram agrupadas para desenvolvimento do protótipo do sistema e durante seu desenvolvimento, a separação de camadas foi efetuada, tendo como camadas: DAO (Objeto de acesso a dados), Business (Regras de negócio), Entity (Representando as entidades da base), Util (Utilitários para o programa, contendo códigos para uso genérico, sendo criptografia ou qualquer outro algoritmo que fosse utilizado por todo o projeto), e o próprio projeto Web com template MVC.

# **FUNDAMENTAÇÃO TEÓRICA**

Neste capítulo serão demonstrados todos os fundamentos no desenvolvimento do projeto, relacionados a Model, View, Controller, API, e conexão com base de dados, criptografia para os dados salvos na base de dados e divisão em 3 camadas.

# MVC (MODEL, VIEW, CONTROLLER)

Padrão de arquitetura de software, separa as camadas de uma aplicação em diferentes níveis. Define a divisão entre três componentes: Model, View e Controller. Cada um destes componentes tem um objetivo individual e específico, estando conectados entre si. O objetivo é separar a arquitetura do software para facilitar a compreensão e a manutenção, ou seja, o objetivo em do MVC consiste em “separar as regras e lógicas do negócio da apresentação em si, permitindo um maior controle sobre a aplicação” (ZACCANINI, 2010)

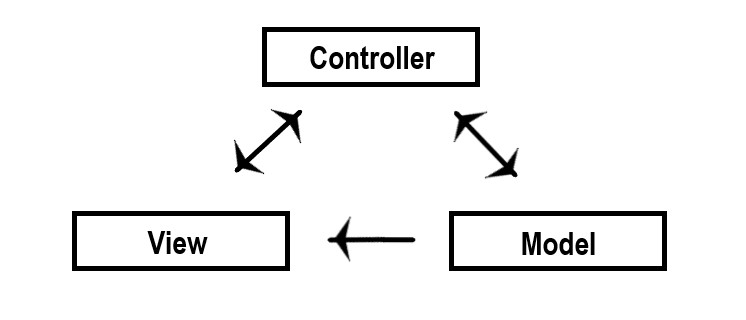


Figura 1 - Modelo MVC

* + 1. MODEL

Camada em que é ocorrida a manipulação, leitura e escrita, também de validações dos dados recebidos.

É nessa camada que, por boas práticas, efetuamos a validação dos dados em propriedades do C#, além de uma predefinição de como a View deve mostrar essas informações a partir de um conceito conhecido como **Data Annotation**, que são nada mais que “classes de atributos que são usadas para definir metadados e que podemos usar para realizar a validação dos dados.” (Macoratti). O conceito de Data Annotation se estende além da model, e pode ser aplicada na camada de entidade de um projeto, como foi o caso do projeto construído e o caso mostrado abaixo

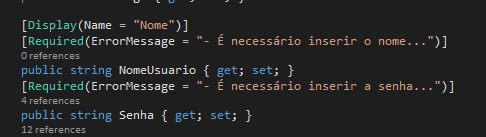


Figura 2 - Modelo de Data Annotation

* + 1. VIEW

Camada de visualização do usuário, onde inserimos todas as informações de layout e demonstração do sistema, é por onde o sistema se “comunica” com o usuário, para que ele insira as informações requeridas pela página.

Na camada de View, ocorrem as validações iniciais, em *Client* **¹,** se as informações foram preenchidas corretamente conforme pedido. Geralmente possui código HTML e CSS, e no Visual Studio e em projetos C#, localizam-se na pasta de Views, dentro do projeto Web. Além de HTML, podemos ter PUG, ou SASS para desenvolvimento do Front-end.

* + 1. CONTROLLER

Camada de controle, responsável por receber todas as requisições do usuário. Na camada de Controller, ocorrem chamadas às outras camadas. No caso do projeto a ser criado, essa camada ficará responsável por chamar os métodos internos para preenchimento dos dados ou inserção dos mesmos,

dependendo do fluxo que o usuário se encontra. Como contexto, essa camada executaria o início da regra de negócio da aplicação, recebendo os dados diretamente da View.

Uma vez dentro da controller, e com os dados recebidos, a classe se encarrega de efetuar os métodos programados para que ocorram validações, alterações, e até mesmo novas inserções. Em C#, é a classe que contém os métodos de retorno **ActionResult**.

¹Client: Navegador, geralmente é o usuário e seu navegado

# **API**

Uma API, acrônimo para **Application Programming Interface,** é uma ferramenta utilizada para compartilhar dados e conteúdos entre softwares e aplicações. Essas APIs geralmente são usadas em uma gama de dados contextos, como dinamicamente inserindo dados dentro de uma base de dados, como selecionando os valores contidos na mesma, ou até mesmo efetuando deleção. Geralmente, para um sistema legado sofrer atualizações, é recorrida à utilização de uma API ou Middleware que traga os dados dessa API para esse sistema, já que é fácil desenvolver uma API de busca de dados por exemplo. Há 3 tipos de API, a **Privada**, **Parceiros** e **Pública**.

APIs privadas, são utilizadas apenas por empresas de forma interna

APIs de parceiros, são utilizadas para compartilhamento com companheiros e parceiros de negócio, capaz de “fornecer fluxos de receita adicionais sem comprometer a qualidade” (REDHAT)

APIs públicas, possuem sua disponibilidade generalizada, permitindo o uso de qualquer pessoa ou empresa, permitindo maior crescimento e desenvolvimento de determinado aplicativo, sendo extremamente útil para aplicações Open Source. De acordo com o tópico escrito pela empresa RedHat, a exposição de APIs a parceiros ou publico pode fazer com que ocorra uma expansão da marca de qualquer empresa, além de “Facilitar a inovação aberta ou aumentar a eficiência por meio da colaboração e de desenvolvimento externos.” (REDHAT).

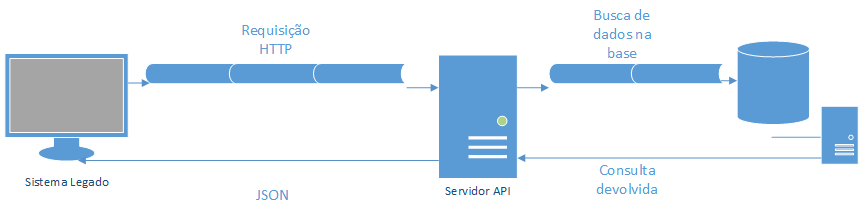


Figura 3 - Modelo de requisição de uma API

Em resumo, uma API nada mais é que uma interface, que ligará o software à uma base de dados apartada na rede web. O diagrama acima demonstra o fluxo, o sistema desenvolvido executa uma requisição HTTP a um servidor que contém a API, esse servidor envia a requisição à base de dados, efetua a consulta, e a base retorna o resultado da consulta, a consulta retorna ao servidor API, que por sua vez, retorna um JSON à aplicação, e então o sistema pode trabalhar com os dados devolvidos.

A API, trabalhando principalmente por HTTPRequest possui 4 métodos de suma importância:

* HttpGet: Tem a função de ir ao servidor e executar uma função de select, sem que ocorra alteração alguma de dados, caso contrário, será barrado
* HttpPost: Tem a função de enviar ao servidor informações de um objeto ou variáveis primitivas, e então, grava na base de dados
* HttpDelete: Tem a função de executar a deleção de determinado dado a partir de algum campo único ou selecionado
* HttpPut: Tem a função de executar um update na base, efetuando alguma alteração em algum dado já existente

# **BASE DE DADOS**

Banco ou base de dados, consiste em um conjunto de dados organizados e armazenados em um computador, de forma que um sistema tenha acesso às informações ali presentes por um SGBD (Sistema Gerenciador de Banco de Dados).

“[...]é uma coleção de programas que permitem aos usuários criarem e manipularem uma base de dados. Um SGBD é, assim, um sistema de software de propósito geral que facilita o processo de definir, construir e manipular bases de dados de diversas aplicações.”(TAKAI, ITALIANO, FERREIRA, 2005)

Os SGBDs(Sistema Gerenciador de Banco de Dados) mais utilizados seriam a versão relacional e não relacional, conhecida como NoSQL, “Estes últimos, cada vez mais vem conquistando espaço nas organizações” (SCUDERO,2016). Isso se deve graças à manutenção, dado o fato de que bases de dados relacionais tendem a crescer de uma forma não muito controlada, necessitando de máquinas um pouco mais robustas com o passar do tempo para conter tamanha base. Por outro lado, é deixado claro por Scudero que: “os SGBDs não relacionais, não exigem máquinas muito poderosas e, por isso, possuem uma manutenção facilitada, exigindo menos mão de obra.” (SCUDERO,2016).

Linguagens de programação como Node.JS, por exemplo, exibem uma afinidade com base de dados não relacional graças à sua performance com as buscas. Por isso, MongoDB vem se tornando cada vez mais popular no mercado em junção com Node.JS. O Banco de dados utilizado pelo sistema é a base SQL, que possuem sua própria linguagem. O diagrama abaixo aponta o esquema de base de dados utilizada, com suas respectivas tabelas e relacionamentos.

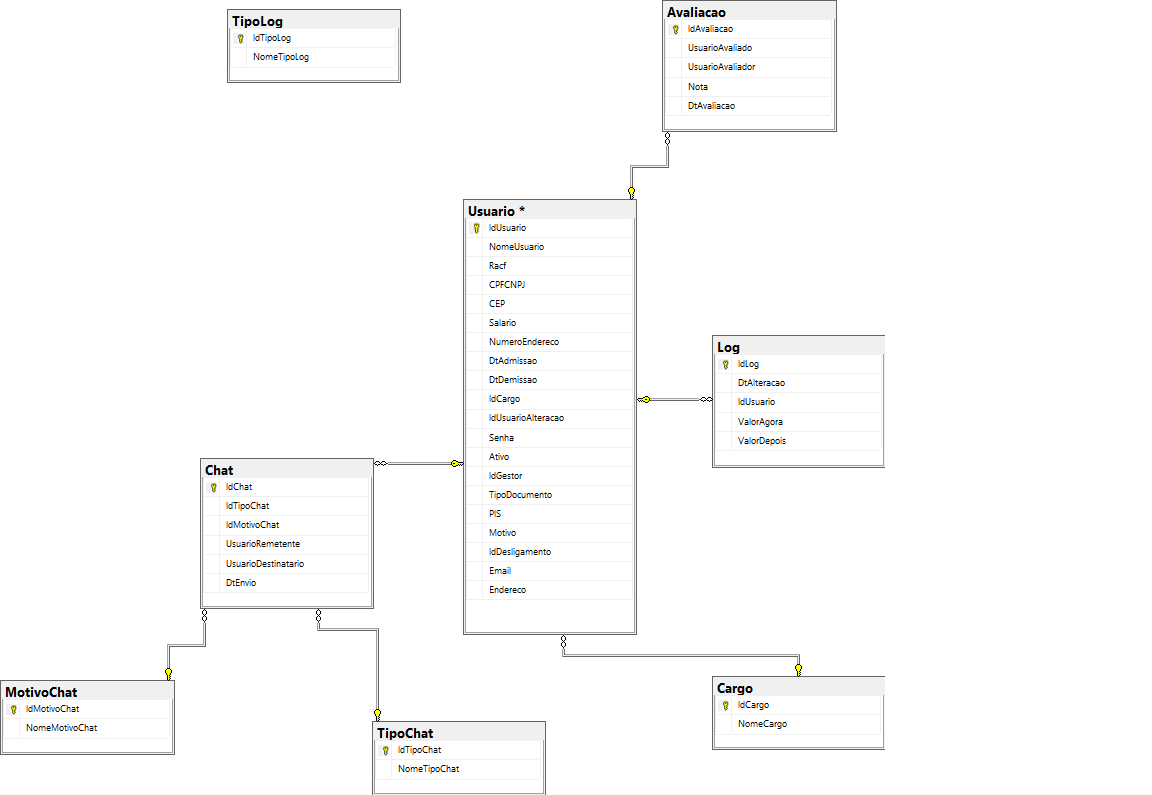


Figura 4 - Diagrama de banco de dados

Foi utilizado relacionamento simples entre as tabelas. Esse relacionamento é feito com uma tabela principal, que contém referência à uma tabela secundária. A chave primária (PK) das tabelas tem como função a indicação dos dados naquela posição da tabela, a referência da segunda tabela, adotada à primeira, é conhecida como Foreign Key (FK).

Quando se trabalha com base de dados, para que o sistema possa pegar as informações, é feito um Modelo Relacionamento-Entidade entre o sistema e a base, no qual é criada uma camada de armazenamento desses dados, conhecida como Repositório.

Quanto maior é o sistema, ou mais antigo, essa responsabilidade de acesso não é passada para o sistema, e sim para uma API, que, como supracitado, efetua a conexão com o banco de dados, seja relacional ou não, e retorna seus valores para o programa, e então, esses dados passam a ser trabalhados, com validações ou regras de negócio pertinentes àquele modelo.

# **Divisão em 3 camadas**

Para o processamento dos dados dentro do sistema, todo o projeto foi dividido em 3 camadas, Entidade, Camada de negócios, e camada de acesso a dados. Com o objetivo de separação das regras de negócio e de acesso a dados do sistema Web.

Entidade: têm como intuito realizar a representação de uma base ou de alguma entidade para o sistema efetuar o trabalho dos dados encima do mesmo. Dentro dessa camada, é *sine qua non* que ocorram determinadas validações relacionadas à regra de negócio.

Camada de negócios: camada que realiza as regras de negócio encima de determinada entidade. A camada de negócio possui total acesso à camada de acesso a dados, porém, não possui acesso algum às entidades, apenas efetua sua validação conforme o desejo do desenvolvedor. Por conter as regras de negócio do sistema, torna-se uma camada extremamente robusta, a qual é necessária aplicação de boas práticas de programação, conhecidas como conceitos de SOLID, para maior estabilidade do sistema e mais facilidade de implementação de novas regras de negócio ou novos componentes necessários ao sistema.

Camada de acesso a dados: camada essencial para acesso à dados contidos em base de dados, ou de qualquer outra fonte, para inserção dos mesmos no sistema, para que ocorra o trabalho dos dados nas camadas superiores. Nessa camada ficam localizadas as queries(comandos) do banco de dados, que podem ser simples comandos em string ou procedures complexas, que retornariam um objeto completo, que foi colocado em uma entidade. Para facilitar a codificação, foi utilizado a biblioteca Dapper, instalado via Nuget Package Manager.

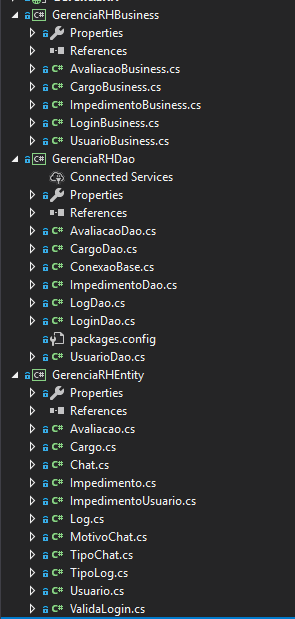


Figura 5 - Exemplo divisão em 3 camadas

A imagem acima demonstra a divisão prática do projeto em 3 camadas, Business (Camada de negócios), Dao (Camada de acesso a dados), e Entity (Camada de entidades).

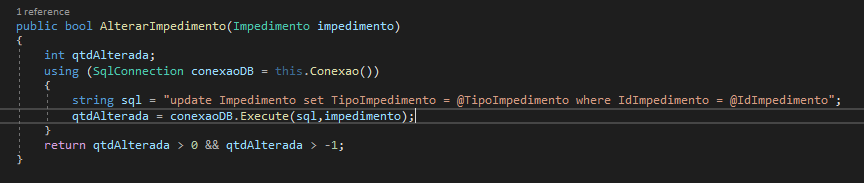


Figura 6 - Exemplo de Query à base

# **CRIPTOGRAFIA**

# HASH

A criptografia se aprimorou com o decorrer do tempo, desde os egípcios até hoje. Em seus conceitos gerais, a criptografia é utilizada principalmente na segurança de informações que antes não estavam tão protegidas, e com o simples intuito de mascarar aquilo que a mensagem realmente significa.

Com a evolução dos meios de comunicação, a criptografia foi instaurada em grande parte desses meios, um exemplo disso é a aplicação dos recursos Web ou nos recursos empresariais que desejam ser escondidos.

**CINCO PRINCIPAIS FUNÇÕES**

• Privacidade: Que ninguém consiga ler a mensagem além da pessoa ou público alvo

• Autenticação: Processo de comprovação da identidade de outrem

• Integridade: Garante que o receptor da mensagem tenha recebido a mensagem íntegra, sem alterações.

• Não-Repúdio: Dado mecanismo que prova que quem realmente escreveu a mensagem foi o desejado autor.

• Troca de chaves: A maneira de como as chaves criptográficas são trocadas entre o mandante e o receptor.

**TIPOS DE ALGORITIMOS CRIPTOGRÁFICOS**

CHAVE SIMÉTRICA

É a criptografia de forma simples, já que tanto o mandante como o receptor possuem as chaves para terem acesso à informação, possui chaves que vão de 8 (mais fraca) a até 1024 bits (mais forte) e foi desenvolvido por Ron Rivest. (Exemplos: DES, IDEA, RC)

CHAVE ASSIMÉTRICA

É a chave criptográfica um pouco mais complexa, uma vez que são utilizadas duas chaves para o acesso à mensagem, a chave pública e a chave privada. Com a chave pública, é feita a codificação da mensagem desejada, e com a chave privada, sua decriptografia. Tanto a chave Simétrica como assimétrica são processos reversíveis de criptografia da mensagem. (Exemplos: RSA, ElGamal)

RECURSO HASH

A criptografia Hash, também conhecida como “one-way” é um tipo de recurso que usufrui do tipo de dado String, formado geralmente por 16 bits, 20 bits e até mesmo 512 bits. Possuí características únicas que, de acordo com Thiago Castelló: “o valor de entrada da string é indeterminado, porém a saída já tem um valor fixo, é uma função One-Way e é livre de colisão” (CASTELLÓ, 2016).

A chave Hash, que quer dizer “picar” ou “cortar” tem como objetivo computar um resumo de mensagem criando uma assinatura digital. Geralmente é utilizada com a chave assimétrica para manter a integridade de um arquivo digital. Deve ter apenas o caminho de ida, jamais o de volta, já que é One-Way, além de ser fácil e prático sua produção.

# UTILIZAÇÂO

Uma vez que é necessária a segurança dos usuários, torna-se difícil a capacidade de obtenção de um item que está criptografado, com esse intuito, o sistema irá dispor de uma criptografia MD5 simples, que irá efetuar a geração de uma chave Hash para a senha do usuário.

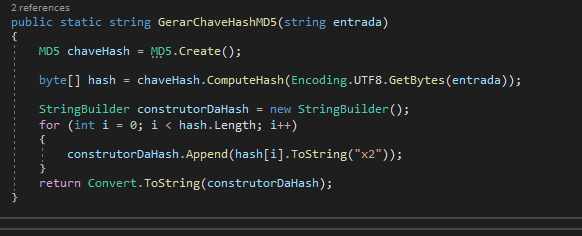


Figura 7 - Método de criptografia utilizado

# **DESENVOLVIMENTO**

Neste capítulo será descrito como foi sucedido todo o desenvolvimento do trabalho, tendo como base o capítulo 2

# **CRIAÇÃO DO PROJETO**

De início, o projeto como um todo teria o foco em um ambiente Web com acesso à um servidor que conteria a base de dados das informações necessárias, portanto, o primeiro passo tomado foi a criação do template da solução na ferramenta Visual Studio, a criação será descrita em partes.

# PROJETO EM CAMADAS

Durante a criação e divisão do projeto, já foi pensado o como seria feita a construção do mesmo em um ambiente de fácil desenvolvimento, portanto, o projeto foi criado com o seguinte template:

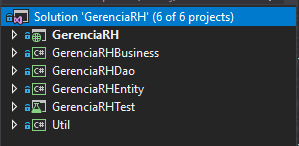


Figura 8 - Exemplificação divisão em camadas

* GerenciaRH: Parte WEB
* GerenciaRHBusiness: Regras de negócio de acesso à base
* GerenciaRHDao: Regras de acesso à base (comandos)
* GerenciaRHEntity: Camada que possui a Entidade representante
* GerenciaRHTest: Camada simples de teste unitário para teste da aplicação
* Util: Camada de utilitários, que conterá os algoritmos extras

# DIAGRAMA DA BASE DE DADOS

Para o início da construção do projeto, foi utilizado um diagrama de construção de base de dados

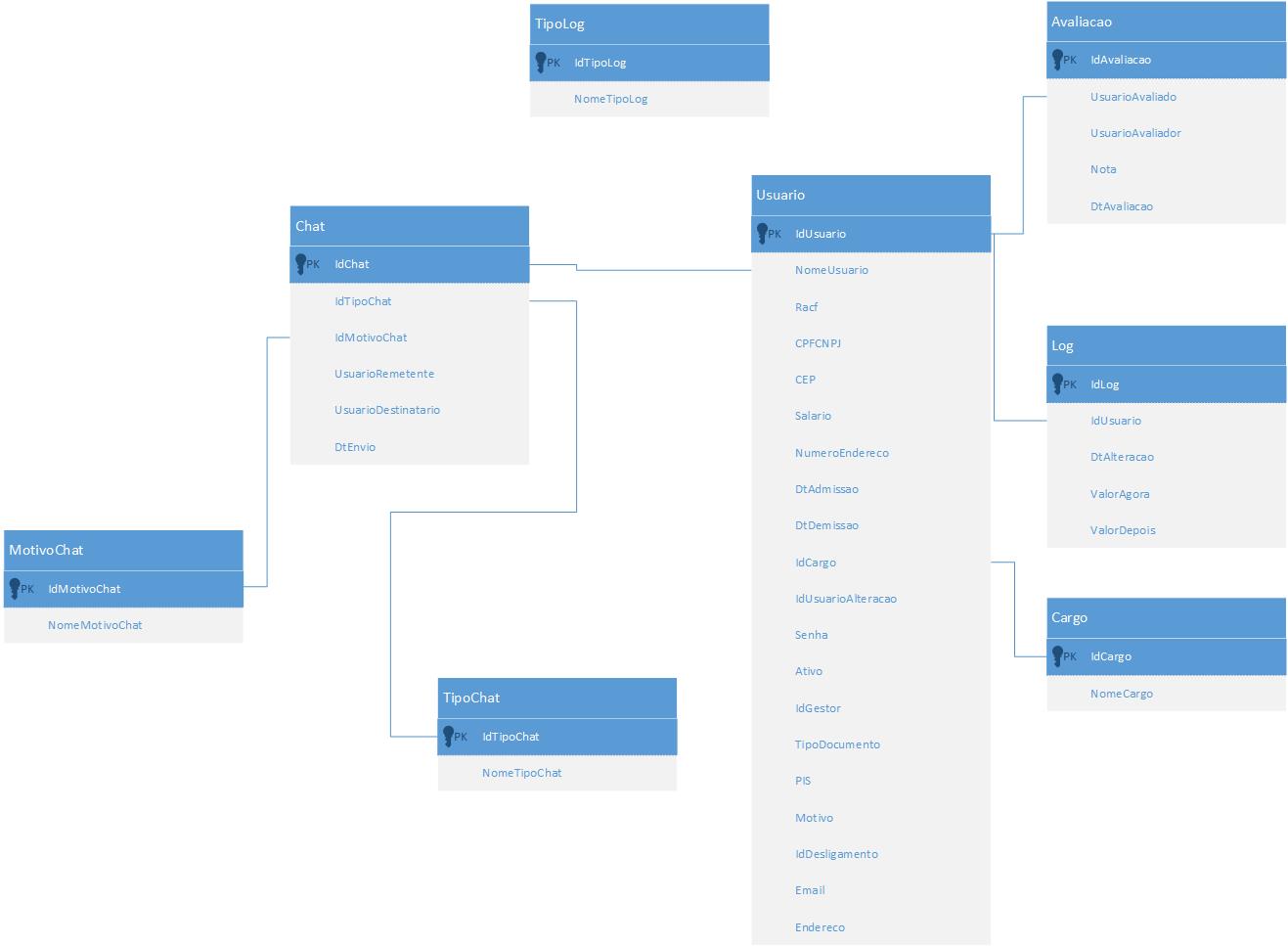


Figura 9 - Diagrama Base de dados do sistema

Baseando-se nesse diagrama, foi instalado e construída uma base dentro do protótipo. E então efetuado um cadastro simples de cada entidade dentro das tabelas, para facilidade durante o desenvolvimento do projeto.

# CONSTRUÇÃO ENTIDADE

Tendo construído a base, foi necessária a comparação da base com os tipos de dados do C#, baseando-se no modelo Relacionamento-Entidade, foi construída uma classe de entidade para cada tabela dentro do sistema, fazendo com que o banco de dados então ficasse “mapeado” pelo sistema. Assim que mapeado, foi implementado o conceito de Data Annotation para futuras validações das entidades que necessitariam de preenchimento, evitando futuras exceções do sistema. O exemplo abaixo representa o modelo da classe Usuário, com a entidade do banco de dados Usuário

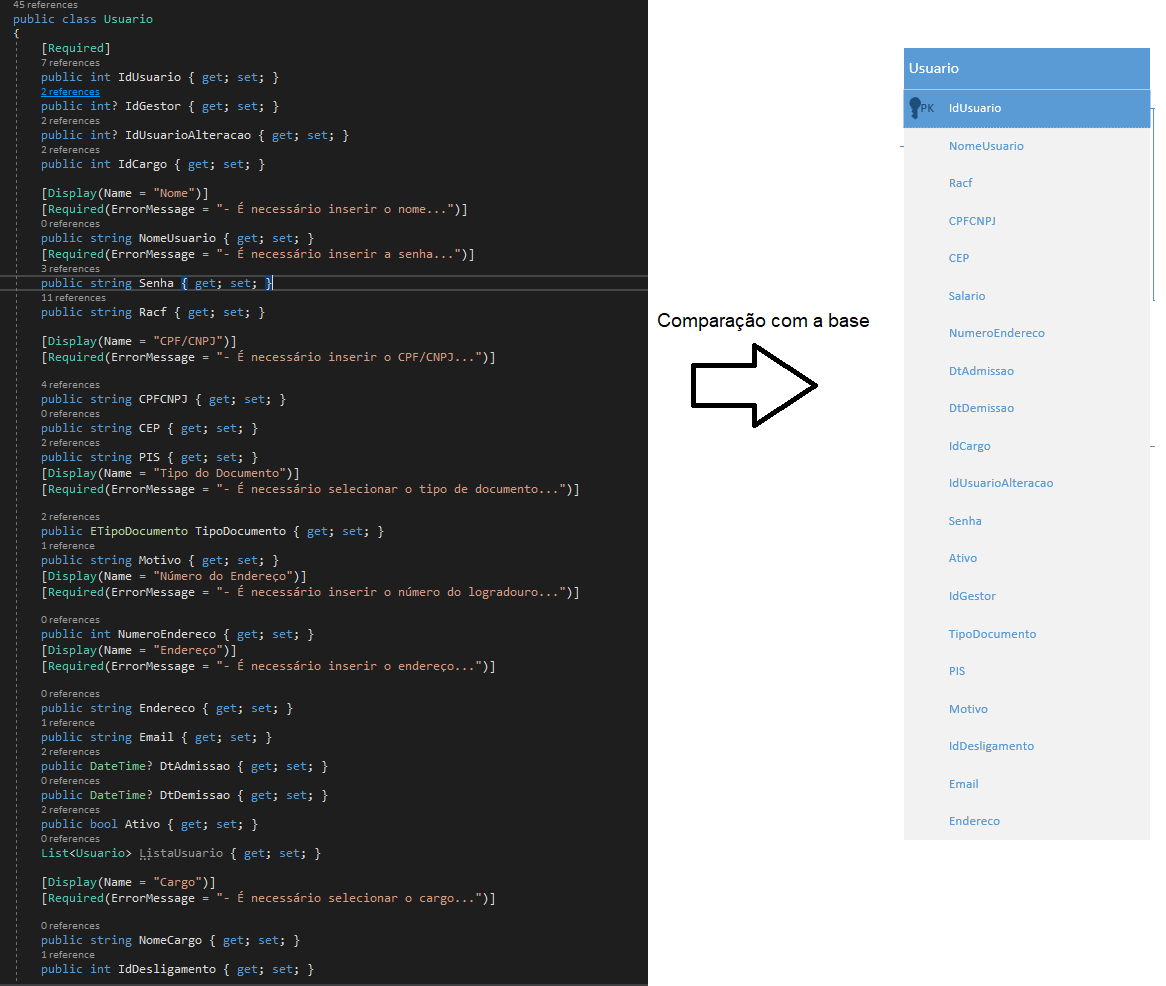


Figura 10 - Comparação base de dados com entidade

Assim como as entidades da base possuem um tipo, para que ocorra o mapeamento perfeito, é necessário que a entidade do projeto possua os mesmos tipos variáveis da base, caso contrário, não será possível armazenar o valor, e ocorrerá uma exceção.

# CAMADA DE ACESSO A DADOS

A construção da camada de acesso a dados ocorreu com a utilização de uma herança de classe, com o intuito de economizar linhas de código e aproveitar a herança da orientação a objeto. A classe pai, possui o método de busca da string de conexão com a base de dados, permitindo que todas as filhas possam acessar a base sem problema algum.

As classes filhas, foram construídas com a utilização de um framework conhecido como Dapper¹, o mesmo executa a query a partir do C# com os parâmetros passados nos métodos, fazendo com que fique extremamente robusto e fácil a passagem do objeto para a base, e seu retorno da mesma forma. A documentação do framework Dapper afirma que “Dapper é um micro ORM ou um mapeador de objetos simples que ajuda a mapear as queries nativas do objeto C#[...]” (DESCONHECIDO, 2017, tradução minha)

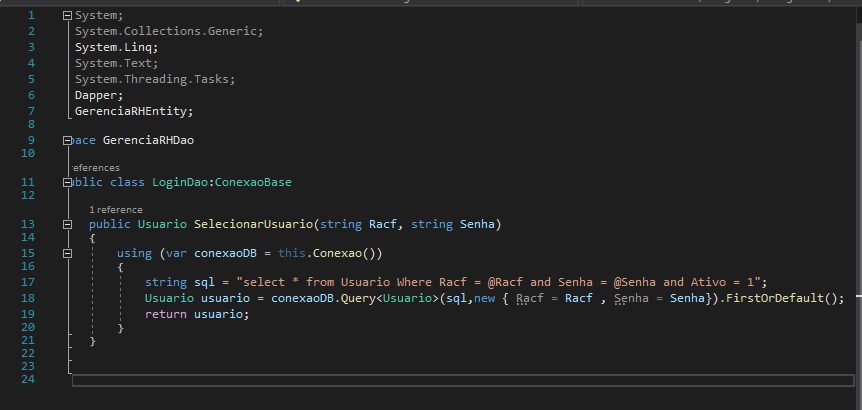


Figura 11 - Exemplo de uso do Dapper

É possível ver na imagem que os parâmetros passados no método são passados para a query string a partir de um @, durante a execução, Dapper transforma os parâmetros da string em objetos, e os envia à base para execução da query.

# CAMADA DE NEGÓCIOS

Após a construção da camada de acesso a dados, foi efetuada a construção da camada de negócios da aplicação, com o intuito de que ocorressem as validações necessárias conforme exigidas pelas regras de negócio do sistema, essa camada faz com que a regra seja cumprida e devolvida às camadas superiores para exposição ao usuário ou talvez para execução de outro método da camada de negócios.

Como regra básica da divisão em 3 camadas, a regra de negócio tem acesso apenas à conexão com a base de dados, jamais à entidade diretamente, uma vez que essa viria nula, assim como ele também não possui acesso às camadas superiores, apenas envia suas informações.

No conceito mais bruto, poderia ser tido como o “Middleware” de um sistema, já que trata os dados, efetua as validações, e as envia para outra camada para que o fluxo não pare.

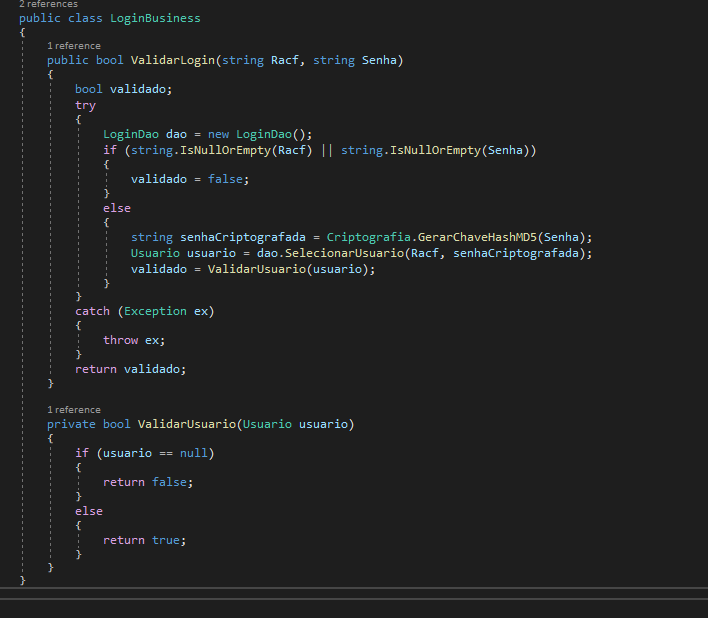


Figura 12 - Exemplo de validação no Login

# CONSTRUÇÃO DO WEB MVC

Na continuidade da construção, após efetuar toda a regra de negócio da aplicação contida na Business, foi criada então a camada Web, que seria a conexão com o usuário propriamente dita.

A camada web iria ser construída de acordo com os padrões MVC. Após a construção da aplicação web, foi então desenvolvida a implementação daquilo que já havia sido criado (camadas inferiores supracitadas) com a camada superior de acesso do usuário, fechando assim o ciclo de desenvolvimento do aplicativo, deixando de ser back-end, para mistura com front-end ao utilizar HTML e CSS. Para que as informações fossem passadas do back-end para o front-end, foi utilizado o conceito de databinding das Models, dentro das Views, utilizando o Razor (ferramenta nativa do Visual Studio). Todas as validações utilizando Data Annotation foram então passadas para as entidades, dado seu desacoplamento do sistema, e as models apenas referenciavam um conjunto dessas classes já validadas.

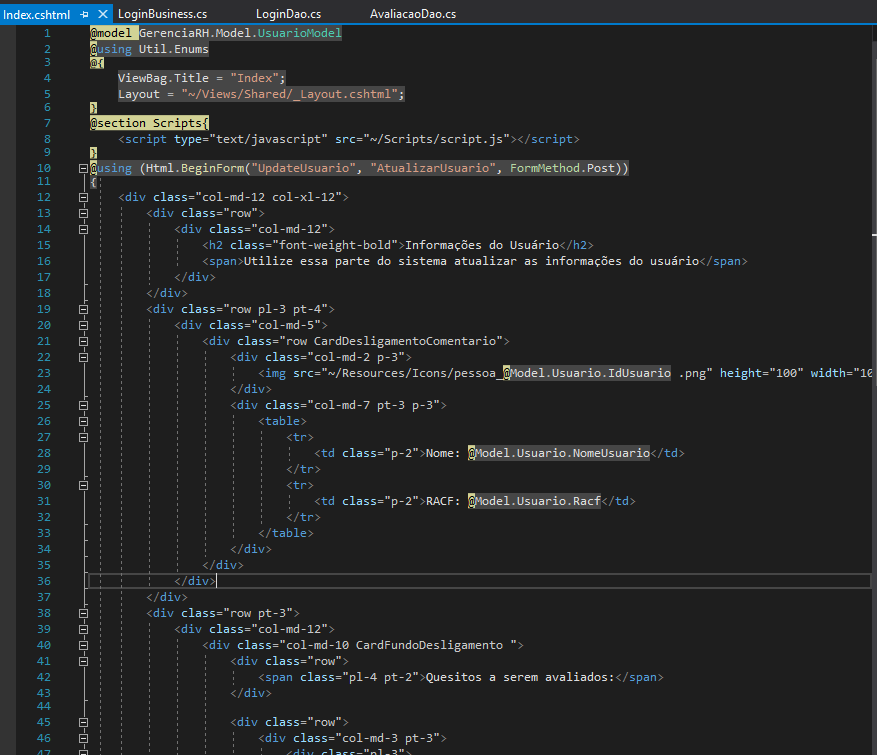


Figura 13 - Uso de data binding Model View

# **FLUXO WEB**

O fluxo web será como o usuário manterá seu fluxo dentro do sistema, a partir da conexão com a internet, é possível acessar a o aplicativo publicado em um determinado servidor web, uma vez que o backend e o front serão mantidos na mesma localidade. O Fluxo web do sistema consiste em:

* Login do usuário ao sistema
* Desligamento de um funcionário de sua área
* Avaliação de um funcionário de sua área
* Obtenção de relatórios atuais, mensais e anuais
* Edição de usuário
* Envio de notificação por mensagem ao ser desligado
* Implementações futuras como chat para discussão de avaliação ou até mesmo resposta ao e-mail recebido do sistema

Assim que ocorrer a abertura do sistema no ambiente web, será obtido a tela de login diretamente, e é então pedido um usuário e senha para acesso ao sistema

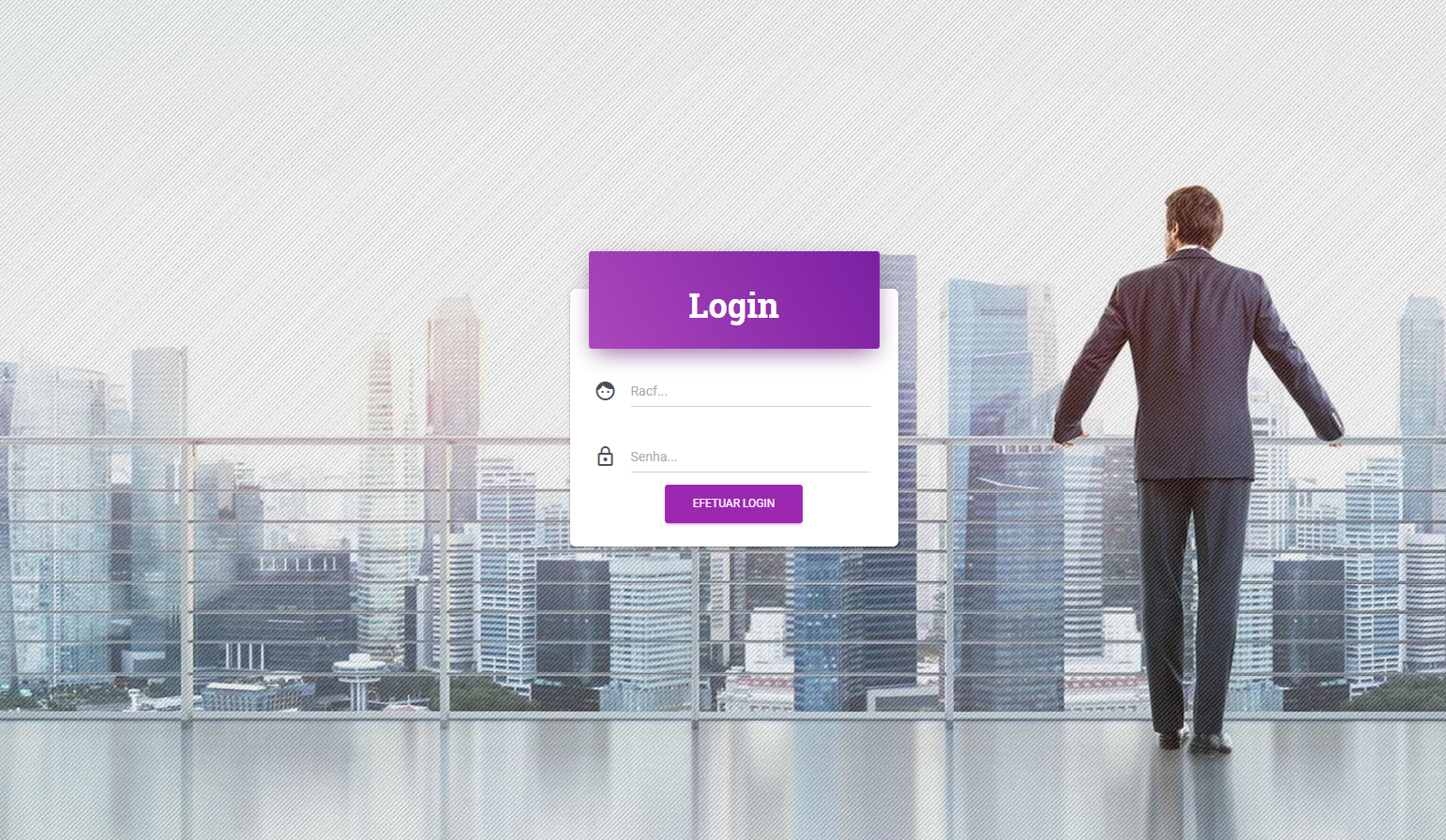


Figura 14 - Tela de Login do sistema

Assim que inserir seu login e senha, caso correto, o usuário então ganhará acesso ao sistema, entrando na home

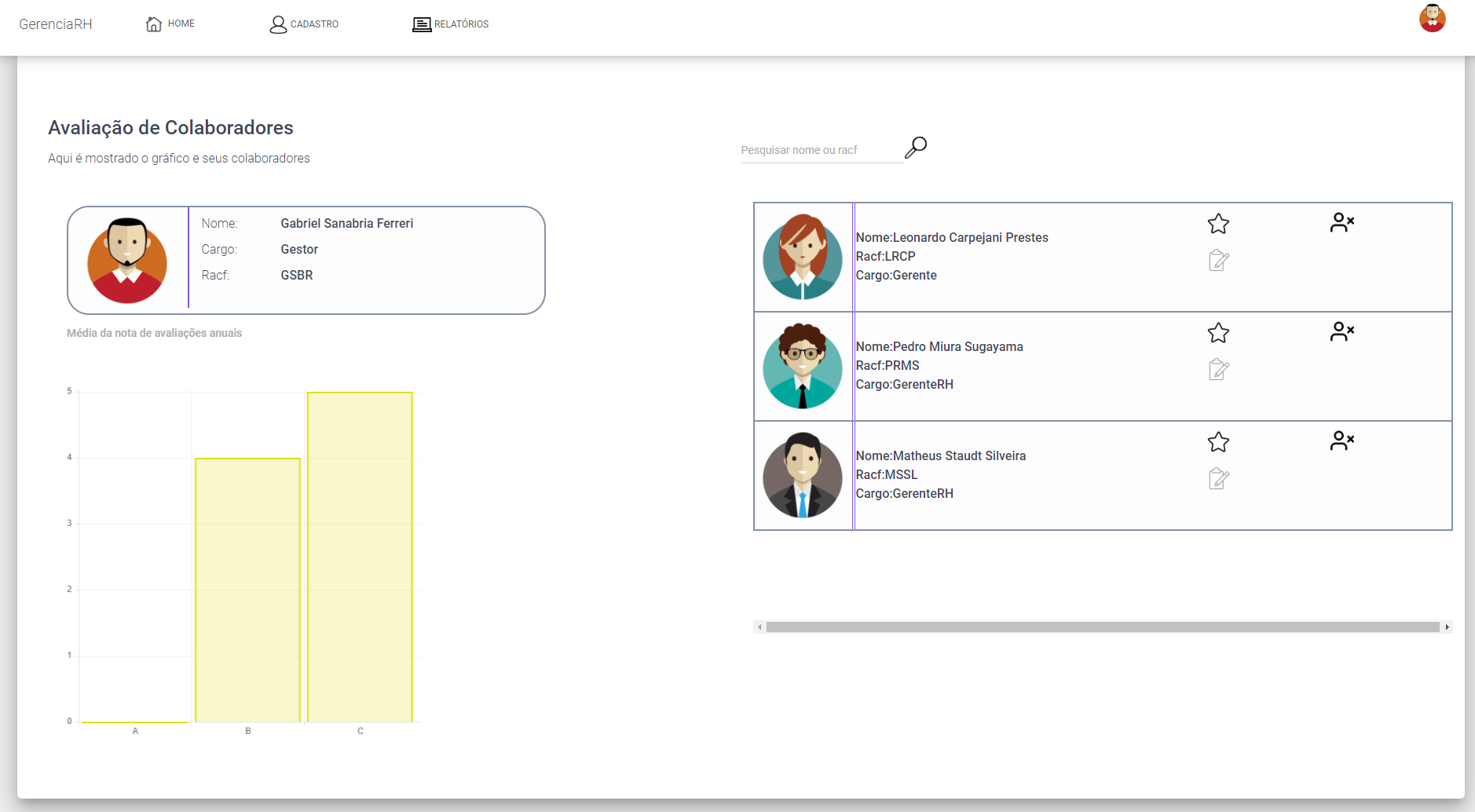


Figura 15 - Home page

Na página de home, o usuário terá então uma visão geral de seu time, tendo a capacidade de efetuar uma as seguintes ações:

* Efetuar Avaliação
* Efetuar Desligamento
* Efetuar Edição do usuário subordinado
* Efetuar um cadastro no sistema
* Gerar relatórios
* Verificar gráfico com quantidade de médias das avaliações atuais geradas

# FLUXO AVALIAÇÃO

Ao clicar em avaliação, o usuário poderá efetuar a avaliação de seu colaborador escolhido.

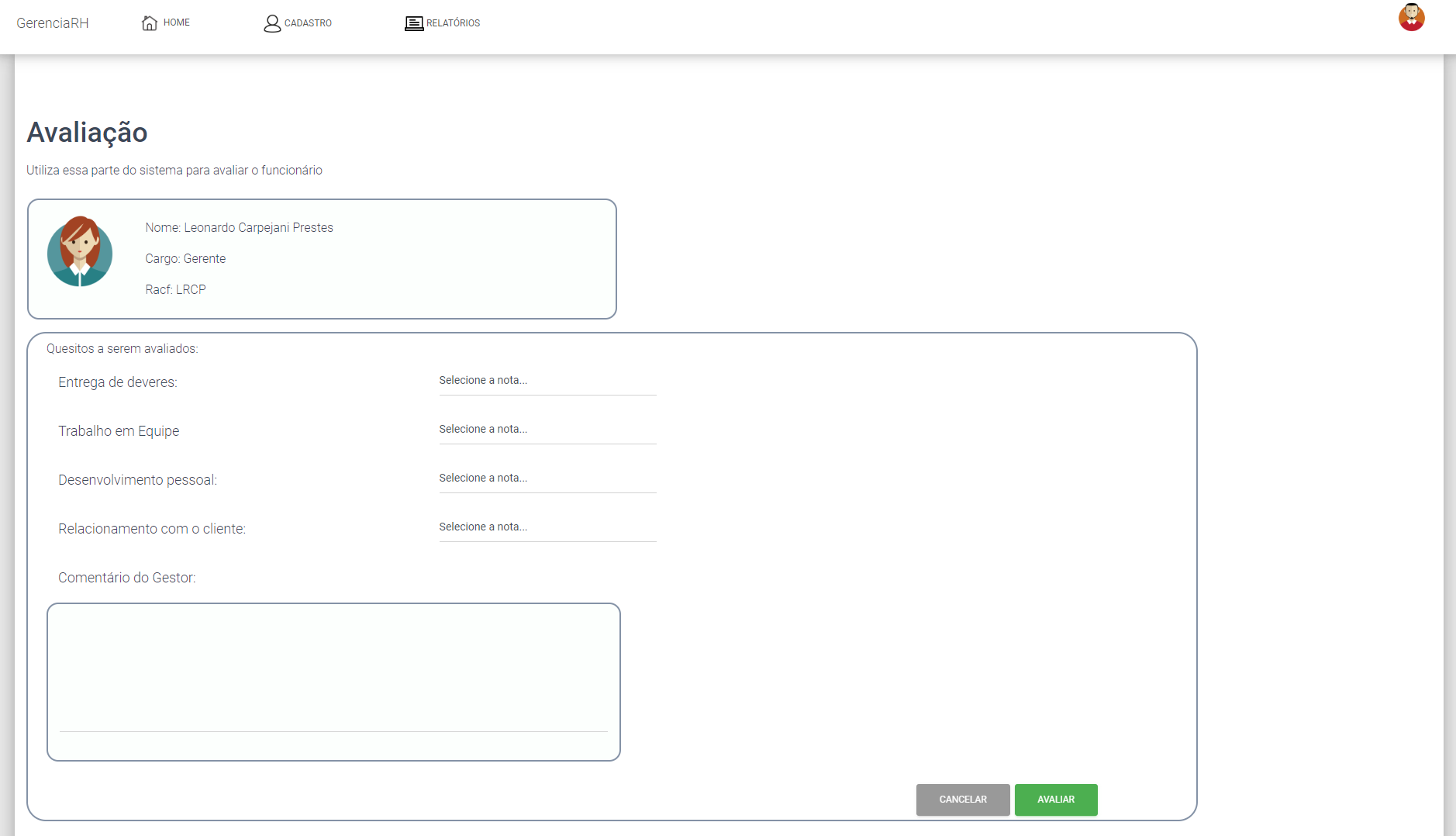


Figura 16 - Tela Inicial Avaliação

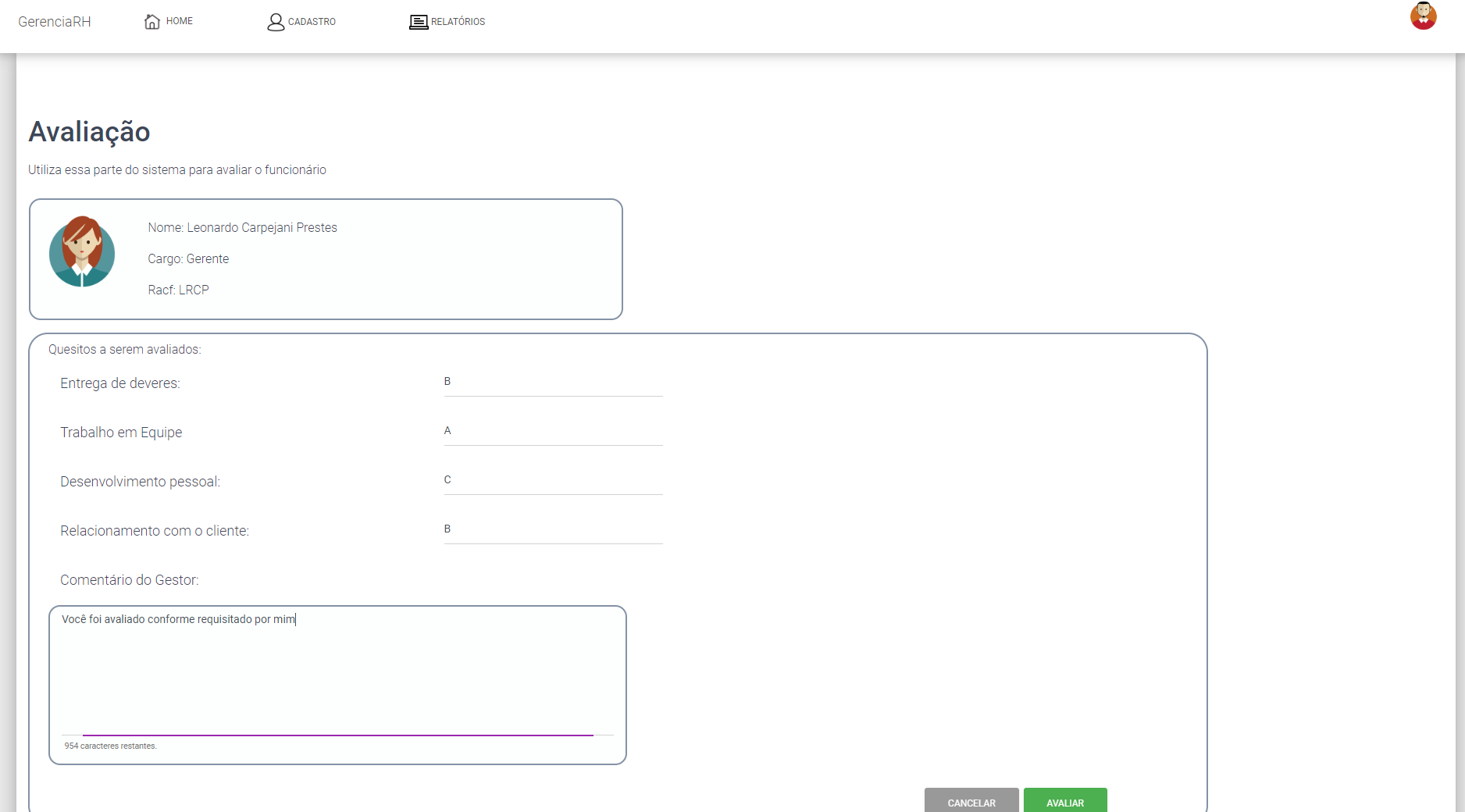


Figura 17 - Tela de avaliação com campos preenchidos

Assim que efetuar a avaliação, é enviado novamente à home, porém com um aviso e seu gráfico com valores atualizados e terminando assim o fluxo de avaliação de seu funcionário.

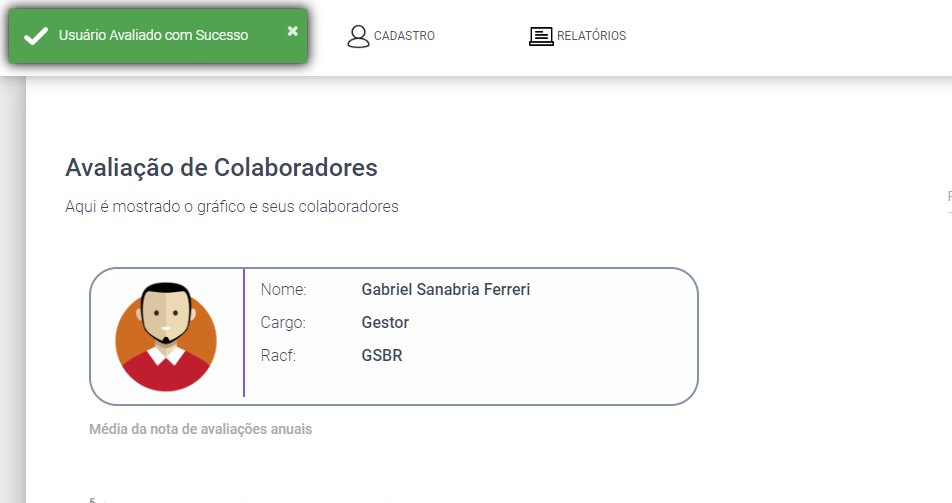


Figura 18 - Fim do fluxo avaliação

# FLUXO DESLIGAMENTO

Assim que o usuário efetuar o clique no ícone do desligamento, será disponibilizada para ele uma visualização da tela, com o funcionário que assim foi clicado para desligamento

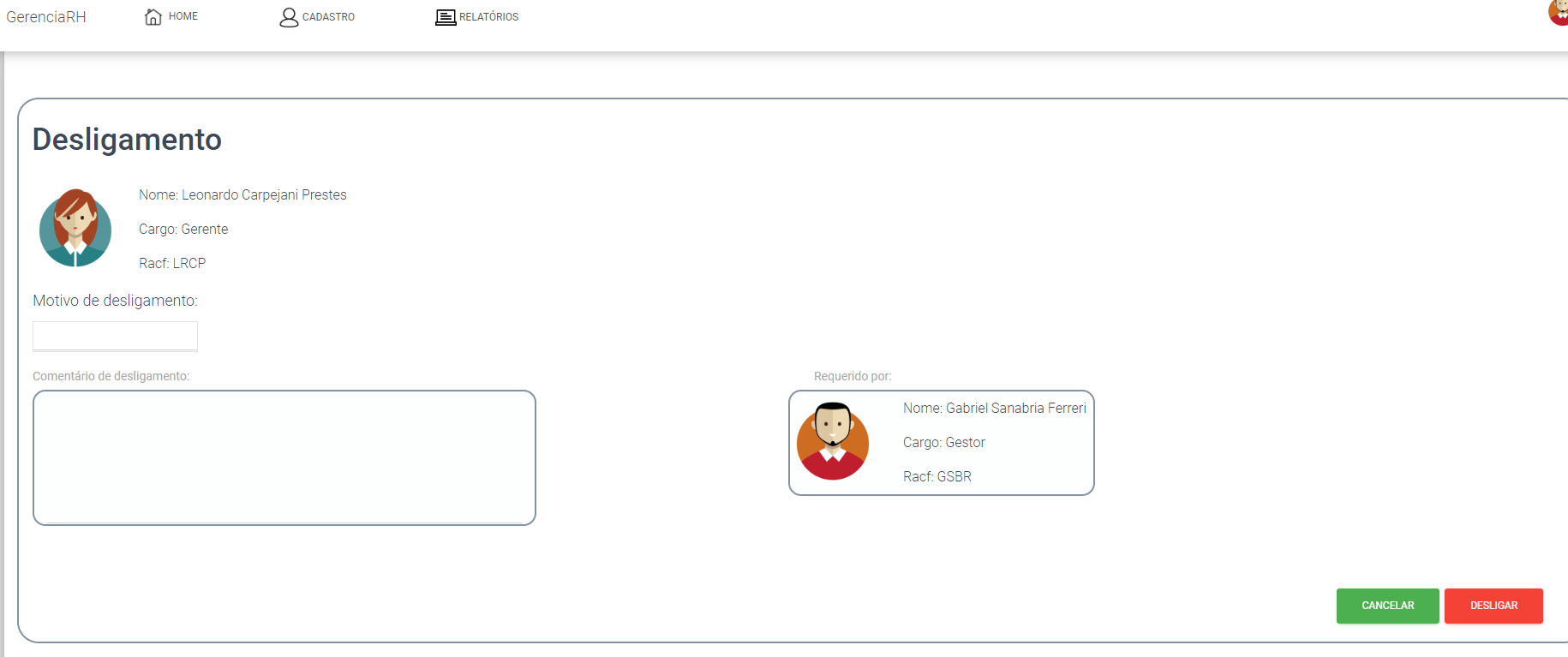


Figura 19 - Início fluxo de desligamento

Assim que o usuário efetuar o clique em Desligar, o mesmo não aparecerá mais na sua tela de home, e o seu colaborador terá recebido um e-mail formalmente comunicando o desligamento e comparecimento ao RH para esclarecer assuntos pendentes ou o como ocorreu o desligamento.

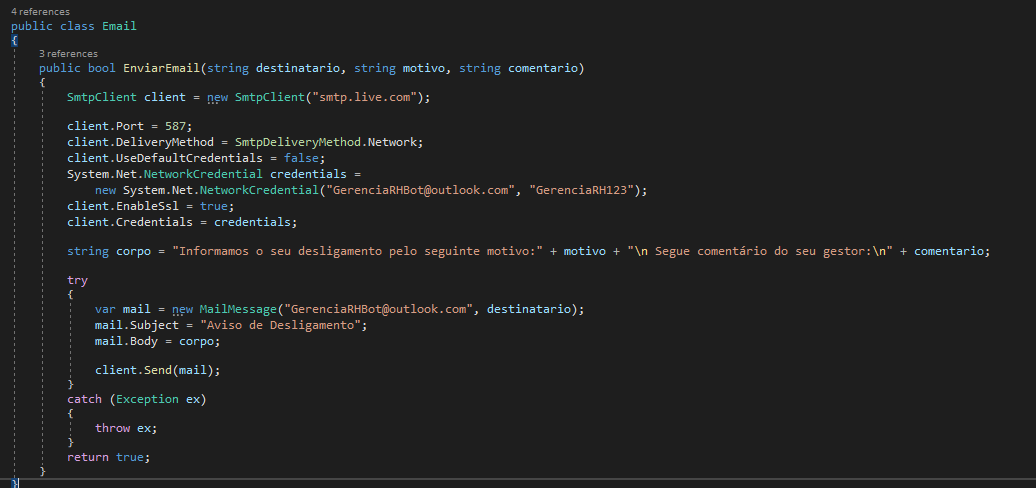


Figura 20 - Código método de envio de e-mail

# FLUXO DE EDIÇÃO DE COLABORADOR

Assim que o usuário na home, efetuar o clique no ícone de edição, o mesmo será movido para uma pagina web de edição dos dados daquele usuário dentro do sistema

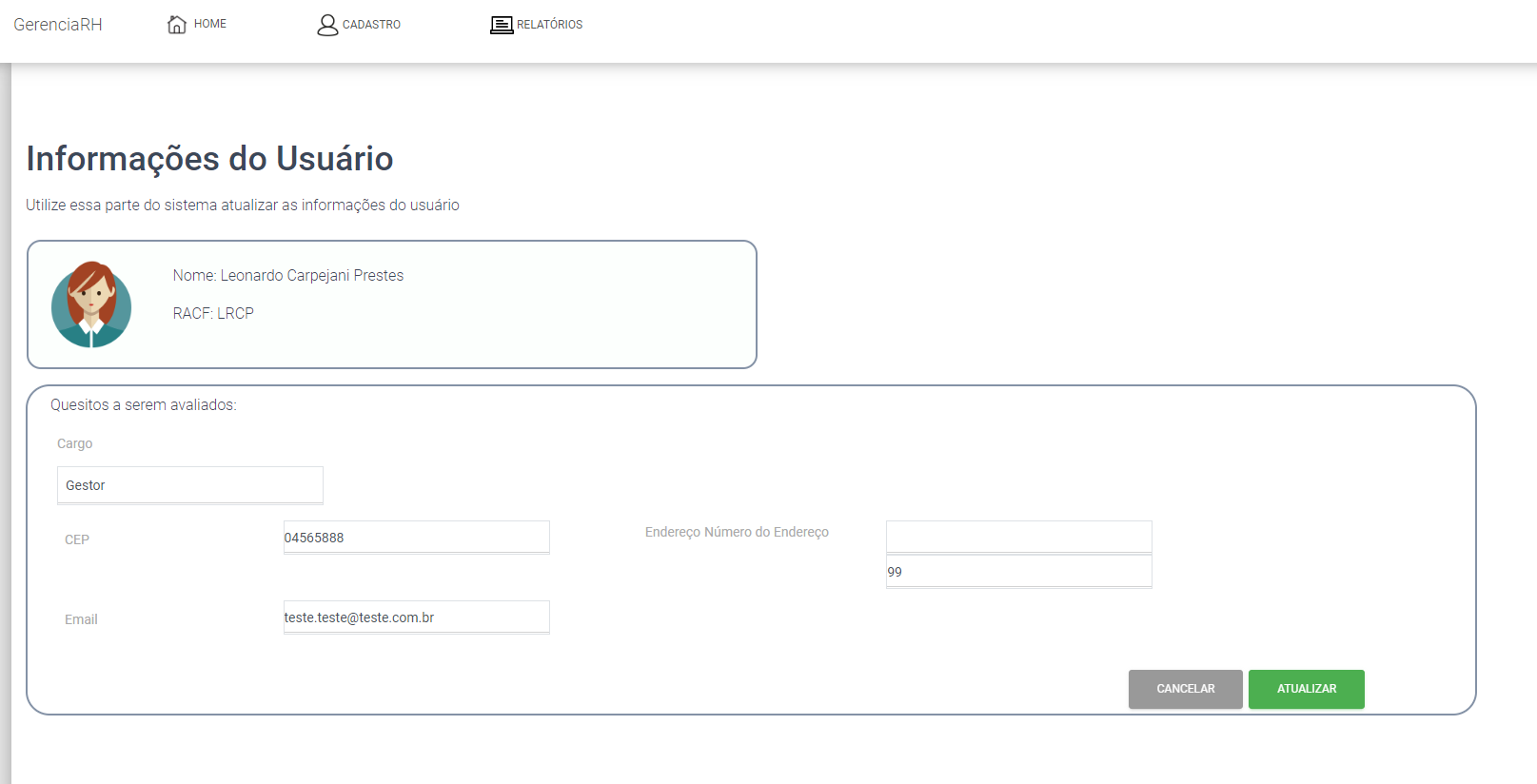


Figura 21 - Tela de edição

Assim que clicar em atualizar, retornará à home com um aviso de que seu colaborador foi atualizado com sucesso.

# FLUXO DE CADASTRO

Ao clicar na aba de cadastro, na sua NavBar, o usuário será realocado para a View de Cadastro de um novo usuário no seu sistema, por consequência, para sua área em específico, no qual, ocorrerá validação dos campos e então, caso preenchido, será devidamente inserido no sistema para acesso posterior

A validação utilizada, foi feita pelo Data Annotation.

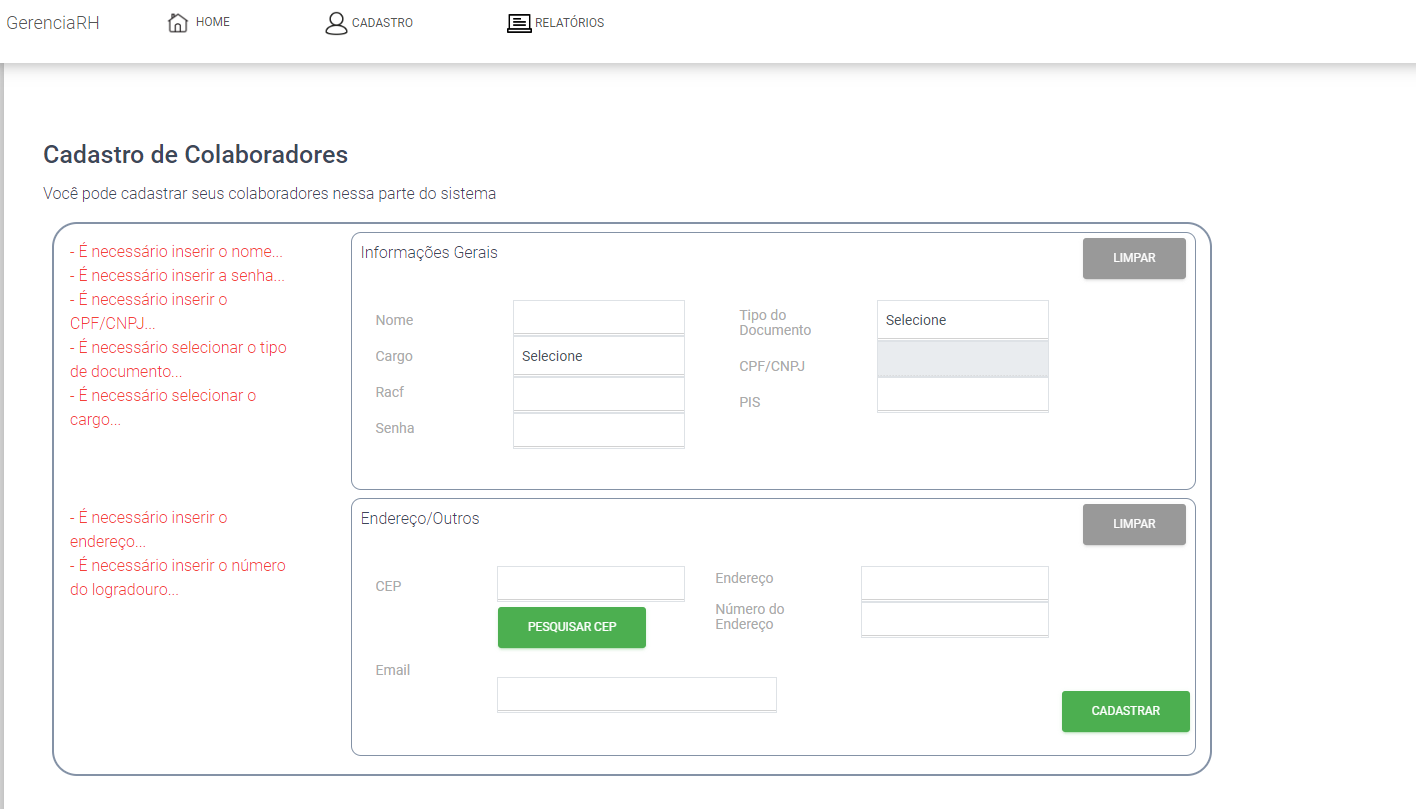


Figura 22 - Validação cadastro novo usuário

A tela possui também uma implementação única de um componente que acessa à uma API cedida pelos correios, para busca de endereço via CEP. Ao inserir os valores no campo CEP e pressionar o botão Pesquisar CEP, o endereço é automaticamente preenchido

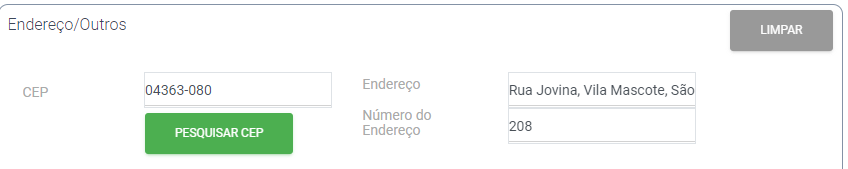


Figura 23 - Preenchimento automático pela API

Após cadastro, usuário retorna para home com seu novo colaborador em sua lista e uma mensagem de sucesso

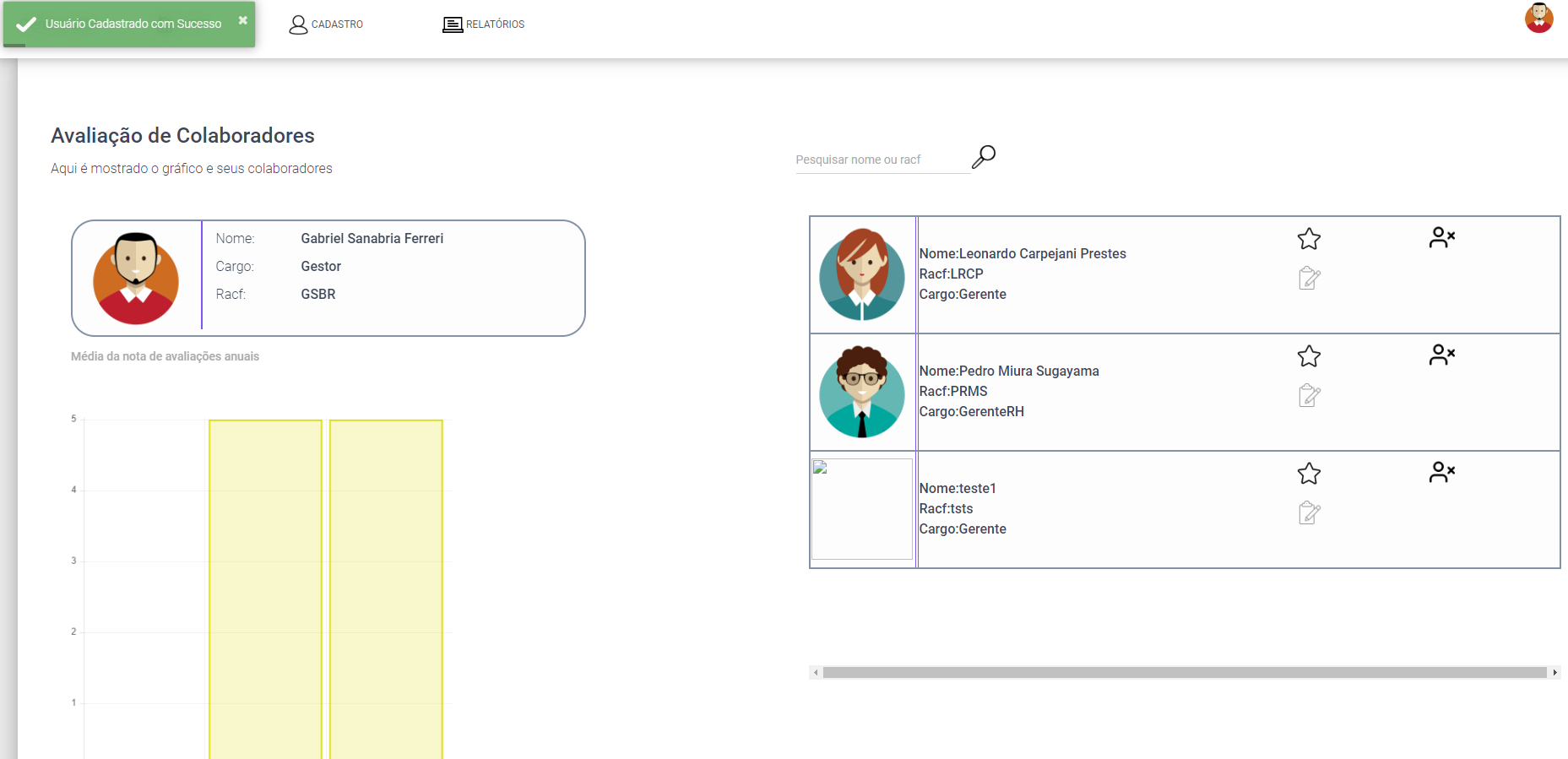


Figura 24 - Cadastro Finalizado

# FLUXO DE RELATORIO

Ao clicar em relatório, o usuário irá visualizar uma tela com relatórios, aonde será possível visualizar o gasto mensal de todos os funcionários contidos sob sua gestão. O gráfico funcionará igual ao da Home, atualizado dinamicamente conforme os valores contidos em base de dados

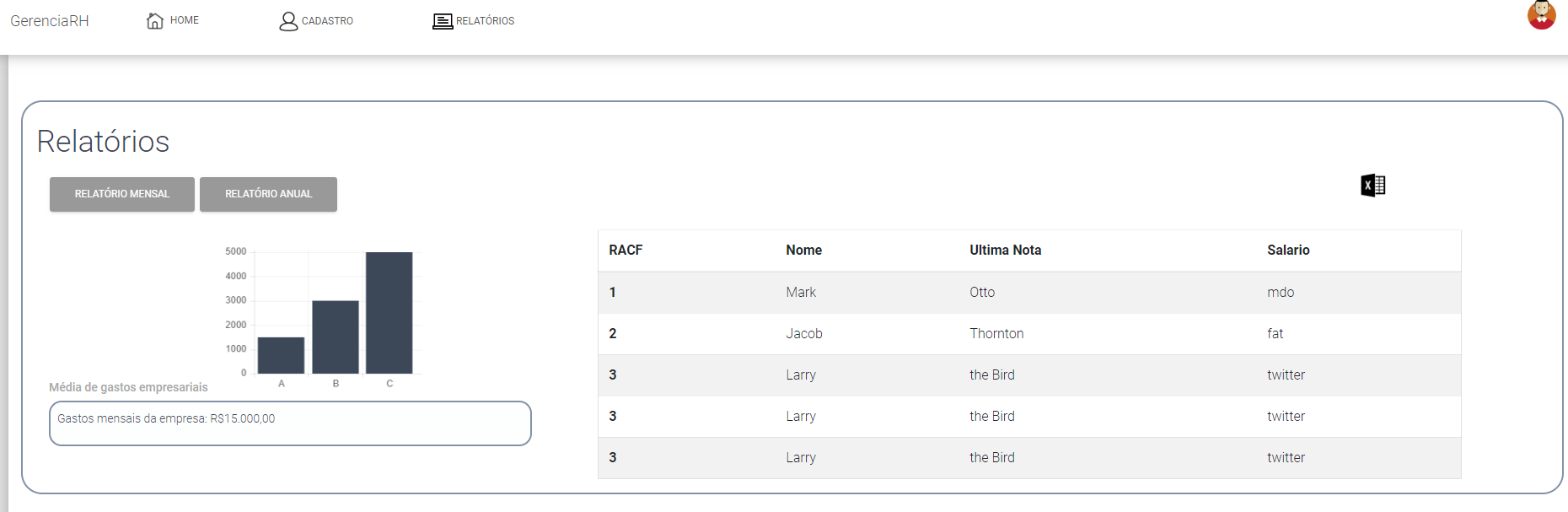


Figura 25 - Tela de relatório

# **ELICITAÇÃO FICTÍCIA DO SISTEMA**

# **ELICITAÇÃO**

CONTEXTUALIZAÇÃO

**O que é pedido**: É pedida a criação de uma ferramenta que efetue a gestão de colaboradores de áreas apartadas em particular, no qual apenas o gestor terá acesso ao sistema

**Problema**: RH Ineficiente para processos de demissão ou relatório.

**Afeta**: Colaboradores das áreas requeridas e gestor

**Impacto**: Médio

**Solução**: Criação de uma plataforma de Gestão e RH que tenha capacidade de efetuar relatórios e início do processo de demissão

**Posição Geral**: O aplicativo será criado para o cliente juntamente de um analista de requisitos entregue pela própria empresa junto com um analista de negócios para avaliação de que a ferramenta, que cumpra o que é pedido.

**Sentença** de Posição

**Para**: Empresa fictícia

**Quer**: Um aplicativo que efetue o auxilio do gestor dentro da empresa, permitindo processos iniciais de demissão e relatórios

**O que faz o aplicativo**: Permite que seja feito o cadastro de um gestor no sistema, logo depois, o gestor irá efetuar o cadastro de seus funcionários para gestão de salários totais da área

**DESCRIÇÃO DOS ENVOLVIDOS**

Os envolvidos com o sistema serão, a empresa, que comprou a ferramenta, e seus clientes a partir da comunicação efetuada pelo sistema. Todo ou qualquer colaborador de cargo de gerente ou superior, poderá efetuar o cadastro de seus colaboradores no sistema para controle e gestão dos mesmos.

**AMBIENTE DO USUÁRIO**

O usuário irá efetuar o login, dado seu acesso. Após entrar no sistema, cadastrará cada um dos funcionários, e conforme regras da empresa, efetuará avaliações dentro da aplicação de tempos em tempos. Serão disponibilizados Dashboards de avaliações e dashboards mensais do desempenho e gasto de sua respectiva área.

**VISÃO GERAL DO QUE FOI CONTRATADO (PRODUTO)**

O produto será baseado em ambiente web. Para toda ou qualquer entrada no sistema, deverá ser efetuado um login conforme cadastrado na base de dados para que então, ocorra a validação, por meio de Hash, da senha cadastrada com o usuário, e o mesmo consiga entrar no sistema, tendo então o completo acesso ao sistema e suas funcionalidades

**ANÁLISE E NEGOCIAÇÃO**

O desenvolvimento do aplicativo tem como objetivo facilitar a gestão dos gerentes e superiores, com o intuito de aliviar a demanda do departamento de RH da empresa, ao solicitarem planilhas e relatórios, ou pedidos de demissão, farão isso pelo software

**APLICATIVO**

a. Deve ser efetuado um login inicial, de um superior, para que então possam ser cadastrados os colaboradores sob sua gestão da área.

b. Deve ocorrer o envio de e-mail automático a partir de uma caixa anônima, o pedido de desligamento de qualquer funcionário

c. Deve ser colocada a opção de gerar relatório.

d. Não deve ser possível responder à caixa de e-mail, ou seja, o mesmo tem que efetuar a deleção automática de qualquer e-mail recebido

e. Deve ocorrer o armazenamento em um arquivo de log, na base de dados, de toda ou qualquer ação do usuário

**CLIENTE**

a. Os colaboradores devem ser cadastrados no sistema de forma completa, com todas as informações pedidas no formulário

# **ESPECIFICAÇÃO DO SOFTWARE**

# INTERFACE DO APLICATIVO

|  |  |  |
| --- | --- | --- |
| Número | Interfaces | Descrição |
| 1 | Tela de Login | Usuário efetuará acesso ao sistema por essa tela |
| 2 | Tela de início | Usuário terá acesso a dashboard de notas e a todas as outras telas |
| 3 | Tela de edição | Usuário poderá editar dados contidos de determinado colaborador |
| 4 | Tela de Cadastro | Usuário irá efetuar a inserção das informações de determinado usuário |
| 5 | Tela de relatório | Usuário conseguirá ter acesso à tela de dashboard para controle de gastos |
| 6 | Tela de Desligamento | Usuário conseguirá efetuar desligamento de qualquer colaborador sob a gestão de sua gerência |
| 7 | Tela de Avaliação | Usuário conseguirá efetuar avaliação de determinado colaborador |

# FUNÇÕES DO SISTEMA

|  |  |  |
| --- | --- | --- |
| Número | Funções | Descrição |
| 1 | Cadastro de colaborador | O sistema deverá permitir o cadastro de colaborador dentro do sistema logado |
| 2 | Desligamento de colaborador | O sistema deverá permitir o desligamento e implementar conforme regulamentos da empresa |
| 3 | Avaliação de Colaborador | O sistema deverá permitir avaliações constantes a princípio, porém, assim que possível, normalizar às regras da empresa |
| 4 | Edição de colaborador | O sistema deve permitir edição de informações cruciais ao gestor da área |
| 5 | Emissão de Relatórios | O sistema deve permitir a emissão de relatórios para facilitação do trabalho do gestor |

# RESTRIÇÕES DE AMBIENTE

|  |  |  |
| --- | --- | --- |
| Número | Restrição | Descrição |
| 1 | Sistema Operacional | SO mínimo: Windows 7 |
| 2 | Conexão à internet | Conexão a todo momento é necessária |
| 3 | Navegador | Navegador obrigatório é Google Chrome |

# DESCRIÇÃO DE USUÁRIOS

|  |  |  |
| --- | --- | --- |
| Número | Usuários | Descrição |
| 1 | Administrador | Acesso total ao sistema para cadastro inicial ou resolução de problemas |
| 2 | Gestor | Usuário do sistema que irá efetuar cadastro de usuários de sua respectiva área |

# **MODELAGEM DO SISTEMA**

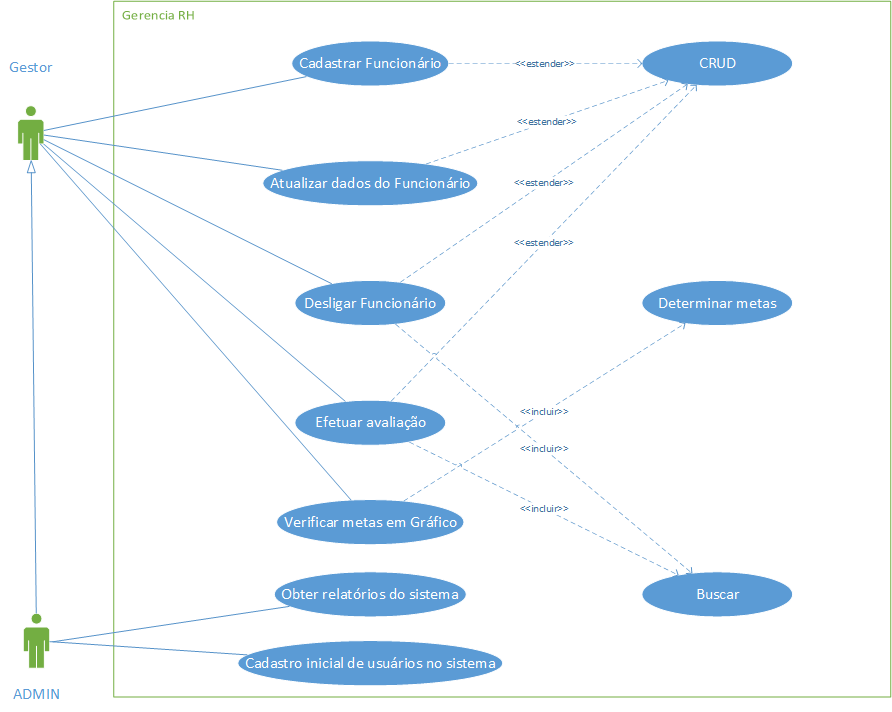


Figura 26 - Casos de uso

O fluxo perfeito do sistema será baseado da seguinte forma: Usuário gestor efetua login, cadastra seus funcionários conforme dados obtidos de algum ambiente da empresa, e ao acessar a tela home, terá acesso a qualquer fluxo desejado. Se desejar alterar informações, basta clicar no ícone. Se desejar desligar algum funcionário, será executado da mesma forma. E assim por diante. Metas serão feitas futuramente no software a partir dos dados contidos nos gráficos (somatórias das notas A, B, C) por time

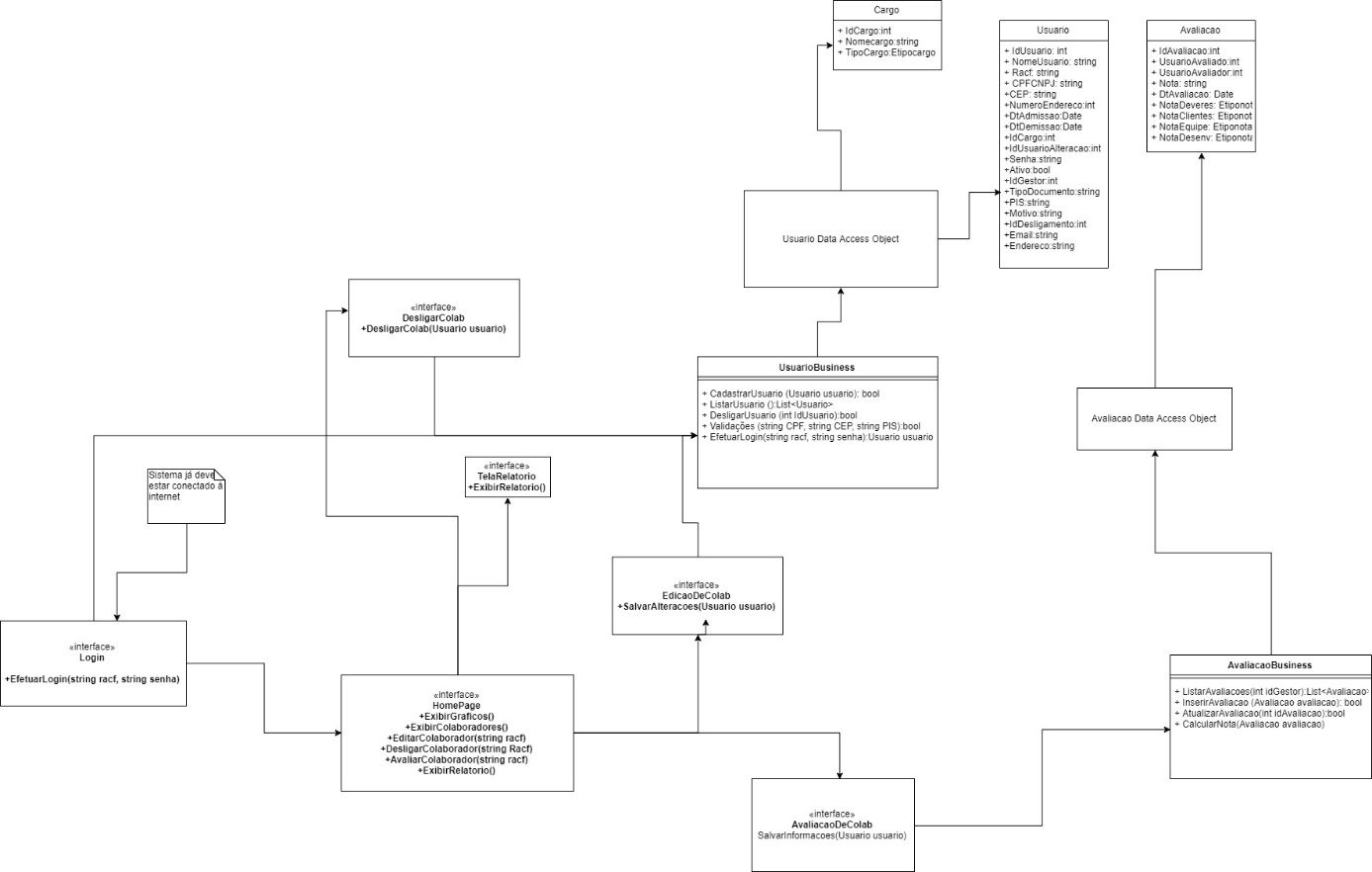


Figura 27 - Diagrama de classes

# PLANOS DE TESTE

**Caso de uso:** Teste de cadastro de funcionário

**Situação:** Inclusão do usuário

**Teste 1:** Todos os campos devidamente inseridos

**Teste bem-sucedido:** Guardar na base as informações do usuário, retornar usuário para homepage com aviso: “Usuário cadastrado com sucesso”

**Teste malsucedido:** Retornará aviso ao usuário, informando com mensagem: “Erro ao gravar na base, entre em contato com um administrador”.

**Teste 2:** Campos parcialmente inseridos

**Teste bem-sucedido:** Retornará ao usuário a mensagem de validação: “Nem todos os campos foram preenchidos corretamente”, não permitindo armazenamento na base

**Teste malsucedido:** Retornará ao usuário uma mensagem especificando o erro.

**Caso de uso:** Atualizar dados do funcionário

**Situação:** Alteração do usuário

**Teste 1:** Campos devidamente preenchidos com alterações

**Teste bem-sucedido:** Guardará informações na base, avisará usuário com mensagem: “Usuário atualizado com sucesso”

**Teste malsucedido:** Manterá os campos preenchidos, e avisará o usuário com a mensagem: “Erro ao atualizar usuário na base”

**Teste 2:** Esvaziar um campo

**Teste bem-sucedido:** Informará ao usuário que há falta de preenchimento de campos

**Teste malsucedido:** Retornará ao usuário uma mensagem especificando o erro.

**Caso de uso:** Desligar funcionário

**Situação:** Desligando funcionário no sistema

**Teste 1:** Selecionando e desligando usuário com campos preenchidos

**Teste bem-sucedido:** Guardará a exclusão de forma lógica com um campo booleano na tabela, informará o usuário com a mensagem de exclusão bem-sucedida

**Teste malsucedido:** Informará o usuário que não pôde ser feito o desligamento

**Teste 2:** Desligando funcionário com motivo vazio

**Teste bem-sucedido:** Informará ao usuário para preencher o campo de motivo com um aviso

**Teste malsucedido:** Informará ao usuário para preencher o campo de motivo com um aviso

**Caso de uso:** Efetuar Avaliação

**Situação:** Avaliação do usuário no sistema

**Teste 1:** Inserindo avaliação com todos os campos devidamente preenchidos

**Teste bem-sucedido:** Guardará a informação informará o usuário com a mensagem de avaliação bem-sucedida

**Teste malsucedido:** Informará o usuário que não pôde ser feito a avaliação dentro do sistema

**Teste 2:** Avaliando funcionário com campos vazios

**Teste bem-sucedido:** Informará ao usuário para preencher os campos

**Teste malsucedido:** Informará que ocorreu algum erro durante a avaliação do colaborador e armazenará em log

**Caso de uso:** Verificar metas em gráfico

**Situação:** Conseguir verificar a somatória das notas na homepage

**Teste 1:** Entrando na homepage para total de notas

**Teste bem-sucedido:** Exibirá para o usuário um dashboard completo contendo as informações dos últimos conjuntos de notas concedidas

**Teste malsucedido:** Informará ao usuário que não pôde ser feito a busca das informações na base de dados.

**Teste 2:** Conseguir pegar as informações de salários na tela de relatório

**Teste bem-sucedido:** Exibirá para o usuário um gráfico completo, informando a junção dos salários dentro da base

**Teste malsucedido:** Informará ao usuário que ocorreu um erro ao buscar as informações

**Caso de uso:** Obter relatório do sistema

**Situação:** Obter relatório do sistema

**Teste 1:** Verificar relatório de sucessão de eventos dentro do sistema

**Teste bem-sucedido:** Mostrará para o admin a visualização das ações efetuadas pela quantidade X de usuários do sistema

**Teste malsucedido:** Informará ao admin que não foi possível efetuar a busca dos últimos dados

**Caso de uso:** Cadastro inicial de usuário no sistema

**Situação:** Cadastrar primeiro usuário

**Teste 1:** Cadastro inicial de usuário pelo sistema

**Teste bem-sucedido:** Informará ao admin que o cadastro foi efetuado com sucesso dentro do sistema

**Teste malsucedido:** Informará ao admin que não foi possível efetuar o cadastro

**Teste 2:** Cadastro inicial de usuário pelo sistema com campos vazios

**Teste bem-sucedido:** Informará ao admin que o cadastro não pode ser feito com campos obrigatórios vazios

**Teste malsucedido:** Informará ao admin que não foi possível efetuar o cadastro devido a algum erro na base

# **DISCUSSÃO**

Neste capítulo será apresentada a discussão encima do que foi apresentado no projeto.

# FINALIDADES E VANTAGES

O intuito final do desenvolvimento do projeto é demonstrar a facilidade que se possui ao desenvolver uma ferramenta para ambientes web que tenham um servidor de armazenamento. Afinal, toda ou qualquer parte burocrática de instalação do software, implementação física, ou qualquer outra necessidade de hardware, tornou-se um obstáculo pequeno ou mínimo. O sistema possui vantagens primárias como:

* Fácil acesso devido ao desenvolvimento ser em ambiente WEB
* Fácil implementação
* Separação em camadas daquilo que será executado

# FACIL ACESSO

O ambiente web é conhecido por uma vasta quantia de qualidades perante determinadas situações, dentre aquelas que mais se tornam destoantes, seria o fato de que a interoperabilidade de um ambiente web é extremamente abrangente, uma vez que, qualquer sistema operacional consegue acessar uma página da internet. Tendo isso em mente, o sistema, por ser desenvolvido nesse ambiente, torna-se extremamente acessível para qualquer sistema operacional, com dependências de hardware mínimas, como um simples navegador que tenha 512 megabytes de RAM. Também é tido como ponto de vantagem nesse quesito a manutenibilidade do sistema web, principalmente se for desenvolvido com separação em camadas. Uma camada, ao se abstrair de outra, torna-se fracamente acoplada, permitindo que ocorra o desenvolvimento de novas ferramentas dentro do sistema mais fácil.

# FÁCIL IMPLEMENTAÇÃO

A implementação web também alcança níveis de facilidade extremos. Como supracitado, todo ou qualquer ambiente desktop que venha a ser instalado, possui consigo uma gama de impedimentos para determinado ambiente. A Web burla grande parte dessas limitações, deixando o sistema mais fácil de ser implementado. No caso do software, é necessária apenas informações básicas do usuário primário, e então, o mesmo pode continuar o fluxo do sistema cadastrando seus colaboradores e assim por diante.

# SEPARAÇÃO EM CAMADAS

Como supracitado, o sistema desenvolvido em camadas apartadas faz com que camadas superiores possuam menos responsabilidade sobre as demais, efetuando a soltura do acoplamento e permitindo um desenvolvimento de futuras ferramentas contidas no sistema. Também é válido ressaltar que isso tornaria o Bug Fix mais fácil e prático. Unindo a responsabilidade aliviada com conceitos de injeções de dependência ou outros princípios SOLID faria com que o aplicativo ficasse cada vez mais genérico para desenvolvimento, ou seja, mais prático, fácil e rápido.

# CONCLUSÃO

O trabalho acima efetuou a descrição, passo a passo, de como ocorreu o desenvolvimento de uma ferramenta web baseada em ASP.NET MVC. Ao desenvolver um sistema com separação de camadas, é possível afirmar que novas ferramentas e novos algoritmos podem ser adicionados ao sistema ou melhorar aquilo que já é possuído. O intuito final era demonstrar a facilidade, praticidade, e a complexidade de um desenvolvimento sistemático, que usufrui de diagramas de base de dados, casos de uso, e entre outros aspectos da engenharia de software.

# **REFERÊNCIAS BIBLIOGRÁFICAS**

MELOTO, Paulo. **Sistema Web vs Sistema Local**. 21/09/2016. Disponível em: <https://sistemasoma.com.br/blog/tecnologia/sistema-web-vs-sistema-local>Acesso em: 09 de Novembro de 2019.

DESCONHECIDO. **Conceitos de Software e Engenharia de Software.** Disponível em:

<https://www.devmedia.com.br/conceitos-de-software-e-engenharia-desoftware/15730>. Acesso em 02 de Novembro.2019

DESCONHECIDO. **Conceitos de Engenharia de Software**. Disponível em:

<https://www.academia.edu/6899720/Conceitos\_de\_Engenharia\_de\_Software>. Acesso em 01 de Novembro de 2019

CASTELLÓ, Thiago; VAZ, Verônica. **Tipos De Criptografia**. Disponível em:

<http://www.gta.ufrj.br/grad/07\_1/ass-dig/index.html>. Acesso em: 02 Novembro 2019.

ALECRIM, Emerson. **Criptografia**. 2005. Disponível em:

<http://www.infowester.com/criptografia.php>. Acesso em: 01 nov. 2019.

GARRET, Filipe. **O que é Criptografia?** 2014. Disponível em:

<http://www.techtudo.com.br/artigos/noticia/2012/06/o-que-e-criptografia.html>. Acesso em: 03 nov. 2019.

DESCONHECIDO. **Criptografia**. 2012. Disponível em:

<http://cartilha.cert.br/criptografia/>. Acesso em: 01 nov. 2019.

KESSLER, Gary C.. **An Overview Of Cryptography**. 1999. Disponível em:

<http://www.garykessler.net/library/crypto.html#password>. Acesso em: 04 nov. 2019.

VIANA, Ericksen. **Criptografia: Conceitos e Aplicações.** Disponível em: 30

<http://www.devmedia.com.br/criptografia-conceito-e-aplicacoes-revista-easy-netmagazine-27/26761>. Acesso em: 01 nov. 2019.

CONDE, A.R.; SOBRADO, P.P. **Vulnerabilidades Criptográficas**. Disponível em:<https://eseida.wikispaces.com/file/view/Vulnerabilidades+Criptográficas.pdf>. Acesso em: 01 nov. 2019.

SOARES, Salomão. **Criptografia**. 2012. Disponível em: <http://www.ceap.br/material/MAT15032012194445.pdf>. Acesso em: 01 nov. 2019.

BARRON, Brenda. **12 Important Advantages of Web Responsive Design.** 2018. Disponível em < https://business.tutsplus.com/articles/advantages-of-responsive-web-design--cms-30703>. Acesso em: 03 nov.2019

ZACCANINI, Rafael. **O que é e por que utilizar o ASP.NET MVC**. 2010. Disponível em < https://www.devmedia.com.br/o-que-e-e-por-que-utilizar-o-asp-net-mvc/18544>. Acesso em: 05 nov 2019

MACORATTI, José. **C# Validando dados com Data Annotations com Windows Forms.** Disponível em <http://www.macoratti.net/13/12/c\_vdda.htm>.Acesso em: 09 nov 2019

MEDEIROS, Higor. **Introdução ao padrão MVC.** 2013. Disponível em < https://www.devmedia.com.br/introducao-ao-padrao-mvc/29308>. Acesso em: 08 nov 2019

DESCONHECIDO. **O que é uma API?**, Disponível em < https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces>. Acesso em: 08 nov 2019

DESCONHECIDO. **Scholarly Publishing – MIT Libraries.** Disponível em < https://libraries.mit.edu/scholarly/publishing/apis-for-scholarly-resources/>. Acesso em: 05 nov. 2019

TAKAI, Osvaldo; ITALIANO, Isabel; FERREIRA, Eduardo. **Introdução a banco de dados.** Disponível em < https://www.ime.usp.br/~jef/apostila.pdf>. Acesso em: 09 nov. 2019

DESCONHECIDO. **Banco de dados relacional e não relacional: Quando utilizar?.** Disponível em < https://mercadoemfoco.unisul.br/banco-de-dados-relacional-e-nao-relacional-quando-utilizar/>. Acesso em 09 nov. 2019

SCUDERO, Erick. **TOP 10 Principais SGBDs do mercado global.** Disponível em < https://becode.com.br/principais-sgbds/>. Acesso em: 09 nov 2019

DESCONHECIDO. **Material Kit**. Disponível em < <https://www.creative-tim.com/product/material-kit>>. Acesso em: 02 jun 2019

# APÊNDICE – CÓDIGO DO PROGRAMA

# Entidades

public class ValidaLogin

{

    public bool Sucesso { get; set; }

    public string Mensagem { get; set; }

}

public class Usuario

    {

        [Required]

        public int IdUsuario { get; set; }

        public int? IdGestor { get; set; }

        public int? IdUsuarioAlteracao { get; set; }

        public int IdCargo { get; set; }

        [Display(Name = "Nome")]

        [Required(ErrorMessage = "- É necessário inserir o nome...")]

        public string NomeUsuario { get; set; }

        [Required(ErrorMessage = "- É necessário inserir a senha...")]

        public string Senha { get; set; }

        public string Racf { get; set; }

        [Display(Name = "CPF/CNPJ")]

        [Required(ErrorMessage = "- É necessário inserir o CPF/CNPJ...")]

        public string CPFCNPJ { get; set; }

        public string CEP { get; set; }

        public string PIS { get; set; }

        [Display(Name = "Tipo do Documento")]

        [Required(ErrorMessage = "- É necessário selecionar o tipo de documento...")]

        public ETipoDocumento TipoDocumento { get; set; }

        public string Motivo { get; set; }

        [Display(Name = "Número do Endereço")]

        [Required(ErrorMessage = "- É necessário inserir o número do logradouro...")]

        public int NumeroEndereco { get; set; }

        [Display(Name = "Endereço")]

        [Required(ErrorMessage = "- É necessário inserir o endereço...")]

        public string Endereco { get; set; }

        public string Nota { get; set; }

        public long Salario { get; set; }

        public string Email { get; set; }

        public DateTime? DtAdmissao { get; set; }

        public DateTime? DtDemissao { get; set; }

        public bool Ativo { get; set; }

        List<Usuario> ListaUsuario { get; set; }

        [Display(Name = "Cargo")]

        [Required(ErrorMessage = "- É necessário selecionar o cargo...")]

        public string NomeCargo { get; set; }

        public int IdDesligamento { get; set; }

    }

public class TipoLog

    {

        public int IdTipoLog { get; set; }

        public int NomeTipoLog { get; set; }

    }

public class TipoChat

   {

       public int IdTipoChat { get; set; }

       public string NomeTipoChat { get; set; }

   }

public class Salarios

    {

        public int SalarioGerente { get; set; }

        public int SalarioGerenteRH { get; set; }

        public int SalarioFuncionarioRH { get; set; }

        public int SalarioFuncionario { get; set; }

    }

public class MotivoChat

    {

        public int IdMotivoChat { get; set; }

        public string NomeMotivoChat { get; set; }

    }

public class Log

    {

        public int IdLog { get; set; }

        public int IdUsuario { get; set; }

        public int IdUsuarioAlteracao { get; set; }

        public string ValorAgora { get; set; }

        public string ValorDepois { get; set; }

        public DateTime DtAlteracao { get; set; }

    }

public class ImpedimentoUsuario

    {

        public int IdImpedimentoUsuario { get; set; }

        public int IdUsuario { get; set; }

        public int IdImpedimento { get; set; }

    }

public class Chat

   {

       public int IdChat { get; set; }

       public int IdTipoChat { get; set; }

       public int IdMotivoChat { get; set; }

       public int UsuarioRemetente { get; set; }

       public int UsuarioDestinatario { get; set; }

       public DateTime DtEnvio { get; set; }

   }

public class CEP

    {

        [JsonProperty(PropertyName = "cep")]

        public string Cep { get; set; }

        [JsonProperty(PropertyName = "logradouro")]

        public string Logradouro { get; set; }

        [JsonProperty(PropertyName = "complemento")]

        public string Complemento { get; set; }

        [JsonProperty(PropertyName = "bairro")]

        public string Bairro { get; set; }

        [JsonProperty(PropertyName = "localidade")]

        public string Localidade { get; set; }

        [JsonProperty(PropertyName = "uf")]

        public string Uf { get; set; }

    }

public class Cargo

    {

        public int IdCargo { get; set; }

        public string NomeCargo{ get; set; }

        List<Cargo> cargos { get; set; }

        public ETipoCargo TipoCargo { get; set; }

    }

public class Avaliacao

    {

        public int IdAvaliacao { get; set; }

        public int UsuarioAvaliado { get; set; }

        public int UsuarioAvaliador { get; set; }

        public string Nota { get; set; }

        public string UltimaAvaliacao { get; set; }

        public DateTime DtAvaliacao{ get; set; }

        public ETipoNota NotaDeveres { get; set; }

        public ETipoNota NotaClientes { get; set; }

        public ETipoNota NotaEquipe { get; set; }

        public ETipoNota NotaDesenv { get; set; }

    }

# Data Access Object

public class UsuarioDao : ConexaoBase

    {

        public bool AdicionarUsuario(Usuario usuario)

        {

            using (var conexaoBD = this.Conexao())

            {

                string sql = "insert into Usuario values(@NomeUsuario,@Racf,@CPFCNPJ,@CEP,@NumeroEndereco,@DtAdmissao,@DtDemissao,@IdCargo,@IdUsuarioAlteracao,@Senha,@Ativo,@IdGestor,@TipoDocumento,@PIS,@Motivo,@IdDesligamento,@Email,@Endereco)";

                var retorno = conexaoBD.Execute(sql,usuario);

                return retorno > 0;

            }

        }

        public List<Usuario> ListarUsuarios(Usuario usuario)

        {

            List<Usuario> retorno = new List<Usuario>();

            using (var conexaoBD = this.Conexao())

            {

                string sql = "select IdUsuario,NomeUsuario,Racf,CPFCNPJ,CEP,NumeroEndereco,DtAdmissao,DtDemissao,CA.IdCargo,IdUsuarioAlteracao,Senha,Ativo,Endereco, CA.NomeCargo"+

                    " from Usuario inner join Cargo as CA on Usuario.IdCargo = CA.IdCargo Where IdGestor = @IdGestor and Ativo = 1";

                retorno = conexaoBD.Query<Usuario>(sql,new { IdGestor = usuario.IdUsuario }).ToList();

            }

            return retorno;

        }

        public bool UpdateUsuario(Usuario usuario)

        {

            int retorno;

            using (var conexaoBD = this.Conexao())

            {

                string sql = "update Usuario set "

                    +"NomeUsuario = @NomeUsuario"

                    +" , Racf= @Racf"

                    +" ,CEP= @CEP"

                    +" ,NumeroEndereco= @NumeroEndereco "

                    +" ,DtAdmissao= @DtAdmissao"

                    +" ,DtDemissao= @DtDemissao"

                    +" ,IdCargo= @IdCargo"

                    +" ,IdUsuarioAlteracao= @IdUsuarioAlteracao"

                    +" where IdUsuario = @IdUsuario";

                retorno = conexaoBD.Execute(sql,usuario);

            }

           return retorno > 0;

        }

        public bool DeletarUsuario(Usuario usuario)

        {

            using (var conexaoBD = this.Conexao())

            {

                string sql = "update Usuario set Ativo = 0, DtDemissao = GETDATE(), Motivo = @Motivo, IdDesligamento = @IdDesligamento where Racf = @Racf";

                int retorno = conexaoBD.Execute(sql,usuario);

                return retorno > 0;            }

        }

        public Usuario BuscarUsuario(string racf)

        {

            using (var conexaoBD = this.Conexao())

            {

                string Sql = " select us.IdUsuario,us.NomeUsuario,us.CEP,us.CPFCNPJ,us.DtAdmissao,us.NumeroEndereco,us.IdCargo,us.Racf,us.IdGestor,us.Email,us.Endereco,ca.NomeCargo from Usuario us join Cargo ca on us.IdCargo = ca.IdCargo where us.Racf = @Racf";

                Usuario usuario = conexaoBD.Query<Usuario>(Sql, new { Racf = racf }).FirstOrDefault();

                return usuario;

            }

        }

        public List<Usuario> Relatorio()

        {

            List<Usuario> retorno = new List<Usuario>();

            using (var conexaoBD = this.Conexao())

            {

                string sql = "select us.Racf, us.NomeUsuario, av.Nota, ca.Salario from Usuario us"

                        +" join Cargo ca on us.IdCargo = ca.IdCargo"

                        + " join Avaliacao av on us.IdUsuario = av.UsuarioAvaliado"

                        + " where av.UltimaAvaliacao = 1";

                retorno = conexaoBD.Query<Usuario>(sql).ToList();

            }

            return retorno;

        }

    }

public class LoginDao:ConexaoBase

    {

        public Usuario SelecionarUsuario(string Racf, string Senha)

        {

            using (var conexaoDB = this.Conexao())

            {

                string sql = "select \* from Usuario Where Racf = @Racf and Senha = @Senha and Ativo = 1";

                Usuario usuario = conexaoDB.Query<Usuario>(sql,new { Racf = Racf , Senha = Senha}).FirstOrDefault();

                return usuario;

            }

        }

    }

public class ImpedimentoDao : ConexaoBase

   {

       public List<Impedimento> ListarImpedimentos(Impedimento impedimento)

       {

           List<Impedimento> impedimentos;

           using (SqlConnection conexaoDB = this.Conexao())

           {

               string sql = " select TipoImpedimento from Impedimento  Where  (@IdImpedimento is null or @IdImpedimento = 0) or(IdImpedimento = @IdImpedimento) and(@TipoImpedimento is null) or(TipoImpedimento = @TipoImpedimento)";

               impedimentos = conexaoDB.Query<Impedimento>(sql,impedimento).ToList();

           }

           return impedimentos;

       }

       public bool InserirImpedimento(Impedimento impedimento)

       {

           int qtdInserida;

           using (SqlConnection conexaoDB = this.Conexao())

           {

               qtdInserida = conexaoDB.Execute("");

           }

           return qtdInserida > 0 && qtdInserida > -1;

       }

       public bool AlterarImpedimento(Impedimento impedimento)

       {

           int qtdAlterada;

           using (SqlConnection conexaoDB = this.Conexao())

           {

               string sql = "update Impedimento set TipoImpedimento = @TipoImpedimento where IdImpedimento = @IdImpedimento";

               qtdAlterada = conexaoDB.Execute(sql,impedimento);

           }

           return qtdAlterada > 0 && qtdAlterada > -1;

       }

   }

public class ConexaoBase

   {

       string conexao = ConfigurationManager.ConnectionStrings["conexao"].ToString();

       public SqlConnection Conexao()

       {

           SqlConnection conn = new SqlConnection(conexao);

           conn.Open();

           return conn;

       }

   }

public class CargoDao : ConexaoBase

   {

       public List<Cargo> ListarCargos(Cargo cargo)

       {

           List<Cargo> listacargo = new List<Cargo>();

           using(var conexaoBD = this.Conexao())

           {

               string sql = "select  NomeCargo from Cargo Where (@IdCargo is null or @IdCargo = 0) or(IdCargo = @IdCargo) and(@NomeCargo is null) or(NomeCargo = @NomeCargo)";

               listacargo = conexaoBD.Query<Cargo>(sql, cargo).ToList();

           }

           return listacargo;

       }

   }

public class AvaliacaoDao:ConexaoBase

    {

        public List<Avaliacao> ListarAvaliacoes(int idGestor)

        {

            List<Avaliacao> avaliacoes;

            using (SqlConnection conexaoDB = this.Conexao())

            {

                string sql = "SELECT \* FROM AVALIACAO WHERE UsuarioAvaliador = @IdGestor AND YEAR(DtAvaliacao) = YEAR(GETDATE())";

                avaliacoes = conexaoDB.Query<Avaliacao>(sql,new { IdGestor = idGestor}).ToList();

            }

            return avaliacoes;

        }

        public bool InserirAvaliacao(Avaliacao avaliacao)

        {

            int qtdInserida;

            using (SqlConnection conexaoDB = this.Conexao())

            {

                string sql = "insert into Avaliacao values (@UsuarioAvaliado, @UsuarioAvaliador, @Nota, @DtAvaliacao,1)";

                qtdInserida = conexaoDB.Execute(sql, avaliacao);

            }

            return qtdInserida > -1 ||  qtdInserida > 0;

        }

        public bool AlterarAvaliacao(Avaliacao avaliacao)

        {

            int qtdAlterada;

            using (SqlConnection conexaoDB = this.Conexao())

            {

                string sql = "update Avaliacao set UltimaNota = 0 where UsuarioAvaliado = @UsuarioAvaliado";

                qtdAlterada = conexaoDB.Execute(sql,new { UsuarioAvaliado = avaliacao.UsuarioAvaliado});

            }

            return qtdAlterada > 0 || qtdAlterada > -1;

        }

    }

# Business

public class UsuarioBusiness

   {

       UsuarioDao Dao = new UsuarioDao();

       public ValidaLogin CadastrarUsuario(Usuario usuario, int usuarioLogado)

       {

           ValidaLogin validaLogin = new ValidaLogin();

           bool sucesso = true;

           try

           {

               if (usuario.TipoDocumento == Util.Enums.ETipoDocumento.CPF)

               {

                   if (usuario.CPFCNPJ == null || ValidarCPF(usuario.CPFCNPJ) == false)

                   {

                       validaLogin.Mensagem = "CPF inválido";

                       sucesso = false;

                       return validaLogin;

                   }

                   if (usuario.PIS == null || ValidarPis(usuario.PIS) == false )

                   {

                       validaLogin.Mensagem = "PIS Inválido";

                       sucesso = false;

                       return validaLogin;

                   }

                   if (usuario.Racf == null || usuario.Racf.Length != 4 )

                   {

                       validaLogin.Mensagem = "RACF Inválida";

                       sucesso = false;

                       return validaLogin;

                   }

                   usuario.DtAdmissao = DateTime.Now;

                   usuario.Ativo = true;

                   usuario.IdGestor = usuarioLogado;

                   usuario.IdUsuarioAlteracao = usuarioLogado;

                   if (BuscarUsuario(usuario.Racf) != null)

                   {

                       validaLogin.Mensagem = "Usuário ja Cadastrado";

                       sucesso = false;

                       return validaLogin;

                   }

                   if (AdicionarUsuario(usuario) == false)

                   {

                       validaLogin.Mensagem = "Falha ao inserir usuário";

                       sucesso = false;

                       return validaLogin;

                   }

                   validaLogin.Sucesso = sucesso;

                   return validaLogin;

               }

               else if (usuario.TipoDocumento == Util.Enums.ETipoDocumento.CNPJ)

               {

                   if (usuario.CPFCNPJ == null || ValidarCPF(usuario.CPFCNPJ) == false)

                   {

                       validaLogin.Mensagem = "Usuário ja Cadastrado";

                       sucesso = false;

                       return validaLogin;

                   }

                   if (usuario.Racf == null || usuario.Racf.Length != 4)

                   {

                       validaLogin.Mensagem = "RACF Inválida";

                       sucesso = false;

                       return validaLogin;

                   }

                   usuario.DtAdmissao = DateTime.Now;

                   usuario.Ativo = true;

                   usuario.IdGestor = usuarioLogado;

                   usuario.IdUsuarioAlteracao = usuarioLogado;

                   if (BuscarUsuario(usuario.Racf) != null)

                   {

                       validaLogin.Mensagem = "Usuário ja Cadastrado";

                       sucesso = false;

                       return validaLogin;

                   }

                   if (AdicionarUsuario(usuario) == false)

                   {

                       validaLogin.Mensagem = "Falha ao inserir usuário";

                       sucesso = false;

                       return validaLogin;

                   }

                   validaLogin.Sucesso = sucesso;

                   return validaLogin;

               }

               else

               {

                   validaLogin.Mensagem = "CPF/CNPJ não digitados";

                   sucesso = false;

                   return validaLogin;

               }

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public bool ValidarPis(string pis)

       {

           int[] multiplicador = new int[10] { 3, 2, 9, 8, 7, 6, 5, 4, 3, 2 };

           int soma;

           int resto;

           pis = pis.Trim();

           pis = pis.Replace("-", "").Replace(".", "").PadLeft(11, '0');

           if (pis.Trim().Length != 11)

               return false;

           soma = 0;

           for (int i = 0; i < 10; i++)

               soma += int.Parse(pis[i].ToString()) \* multiplicador[i];

           resto = soma % 11;

           if (resto < 2)

               resto = 0;

           else

               resto = 11 - resto;

           return pis.EndsWith(resto.ToString());

       }

       public bool ValidarCNPJ(string cnpj)

       {

           int[] multiplicador1 = new int[12] { 5, 4, 3, 2, 9, 8, 7, 6, 5, 4, 3, 2 };

           int[] multiplicador2 = new int[13] { 6, 5, 4, 3, 2, 9, 8, 7, 6, 5, 4, 3, 2 };

           int soma;

           int resto;

           string digito;

           string tempCnpj;

           cnpj = cnpj.Trim();

           cnpj = cnpj.Replace(".", "").Replace("-", "").Replace("/", "");

           if (cnpj.Length != 14)

               return false;

           tempCnpj = cnpj.Substring(0, 12);

           soma = 0;

           for (int i = 0; i < 12; i++)

               soma += int.Parse(tempCnpj[i].ToString()) \* multiplicador1[i];

           resto = (soma % 11);

           if (resto < 2)

               resto = 0;

           else

               resto = 11 - resto;

           digito = resto.ToString();

           tempCnpj = tempCnpj + digito;

           soma = 0;

           for (int i = 0; i < 13; i++)

               soma += int.Parse(tempCnpj[i].ToString()) \* multiplicador2[i];

           resto = (soma % 11);

           if (resto < 2)

               resto = 0;

           else

               resto = 11 - resto;

           digito = digito + resto.ToString();

           return cnpj.EndsWith(digito);

       }

       public bool ValidarCPF(string cpf)

       {

           int[] multiplicador1 = new int[9] { 10, 9, 8, 7, 6, 5, 4, 3, 2 };

           int[] multiplicador2 = new int[10] { 11, 10, 9, 8, 7, 6, 5, 4, 3, 2 };

           string tempCpf;

           string digito;

           int soma;

           int resto;

           cpf = cpf.Trim();

           cpf = cpf.Replace(".", "").Replace("-", "");

           if (cpf.Length != 11)

               return false;

           tempCpf = cpf.Substring(0, 9);

           soma = 0;

           for (int i = 0; i < 9; i++)

               soma += int.Parse(tempCpf[i].ToString()) \* multiplicador1[i];

           resto = soma % 11;

           if (resto < 2)

               resto = 0;

           else

               resto = 11 - resto;

           digito = resto.ToString();

           tempCpf = tempCpf + digito;

           soma = 0;

           for (int i = 0; i < 10; i++)

               soma += int.Parse(tempCpf[i].ToString()) \* multiplicador2[i];

           resto = soma % 11;

           if (resto < 2)

               resto = 0;

           else

               resto = 11 - resto;

           digito = digito + resto.ToString();

           return cpf.EndsWith(digito);

       }

       public bool AdicionarUsuario(Usuario usuario)

       {

           try

           {

               string senhaCriptografada = Criptografia.GerarChaveHashMD5(usuario.Senha);

               usuario.Senha = senhaCriptografada;

               bool adicionou = Dao.AdicionarUsuario(usuario);

               return adicionou;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public List<Usuario> ListarUsuarios(Usuario usuario)

       {

           try

           {

               List<Usuario> usuarios = Dao.ListarUsuarios(usuario);

               return usuarios;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public bool DeletarUsuario(Usuario usuario)

       {

           try

           {

               bool deletado = Dao.DeletarUsuario(usuario);

               return deletado;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public bool UpdateUsuario(Usuario usuario)

       {

           try

           {

               bool efetuado = Dao.UpdateUsuario(usuario);

               return efetuado;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public Usuario BuscarUsuario(string racf)

       {

           try

           {

               Usuario usuario = Dao.BuscarUsuario(racf);

               return usuario;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

       public List<Usuario> Relatorio()

       {

           try

           {

               List<Usuario> usuarios = Dao.Relatorio();

               return usuarios;

           }

           catch (Exception ex)

           {

               throw ex;

           }

       }

   }

public class LoginBusiness

    {

        public bool ValidarLogin(string Racf, string Senha)

        {

            bool validado;

            try

            {

                LoginDao dao = new LoginDao();

                if (string.IsNullOrEmpty(Racf) || string.IsNullOrEmpty(Senha))

                {

                    validado = false;

                }

                else

                {

                    string senhaCriptografada = Criptografia.GerarChaveHashMD5(Senha);

                    Usuario usuario = dao.SelecionarUsuario(Racf, senhaCriptografada);

                    validado = ValidarUsuario(usuario);

                }

            }

            catch (Exception ex)

            {

                throw ex;

            }

            return validado;

        }

        private bool ValidarUsuario(Usuario usuario)

        {

            if (usuario == null)

            {

                return false;

            }

            else

            {

                return true;

            }

        }

    }

public class ImpedimentoBusiness

   {

       private readonly ImpedimentoDao Dao = new ImpedimentoDao();

       public List<Impedimento> listarImpedimentos(Impedimento impedimento)

       {

           List<Impedimento> impedimentos = Dao.ListarImpedimentos(impedimento);

           return impedimentos.Count > 0 ? impedimentos : new List<Impedimento>();

       }

       public bool InserirImpedimento (Impedimento impedimento)

       {

           bool inserido = Dao.InserirImpedimento(impedimento);

           return inserido;

       }

       public bool UpdateImpedimento (Impedimento impedimento)

       {

           bool alterado = Dao.AlterarImpedimento(impedimento);

           return alterado;

       }

   }

public class CepBusiness

    {

        public CEP ConsultarEndereco(string cep)

        {

            CEP OCep = new CEP();

            try

            {

                WebClient client = new WebClient();

                var result = client.DownloadString("http://viacep.com.br/ws/" + cep + "/json/");

                result = result.Replace("£", "").Replace("Ã", "ã");

                if (result.Contains("erro"))

                {

                    return OCep;

                }

                else

                {

                    OCep = JsonConvert.DeserializeObject<CEP>(result);

                }

            }

            catch (Exception ex)

            {

            }

            return OCep;

        }

    }

public class CargoBusiness

    {

        CargoDao Dao = new CargoDao();

        public List<Cargo> ListarCargos (Cargo cargo)

        {

            List<Cargo> cargos = Dao.ListarCargos(cargo);

            return cargos;

        }

    }

public class AvaliacaoBusiness

    {

        private readonly AvaliacaoDao Dao = new AvaliacaoDao();

        public List<Avaliacao> ListarAvaliacoes(int idGestor)

        {

            try

            {

                List<Avaliacao> avaliacoes = Dao.ListarAvaliacoes(idGestor);

                return avaliacoes;

            }

            catch (Exception ex)

            {

                throw ex;

            }

        }

        public bool InserirAvaliacao(Avaliacao avaliacao)

        {

            try

            {

                bool verificacao = Dao.AlterarAvaliacao(avaliacao);

                avaliacao.Nota = CalcularNota(avaliacao);

                avaliacao.DtAvaliacao = DateTime.Now;

                bool inserido = Dao.InserirAvaliacao(avaliacao);

                return inserido;

            }

            catch (Exception ex)

            {

                throw ex;

            }

        }

        //teste

        public string CalcularNota(Avaliacao avaliacao)

        {

            int notaFinal = 0;

            double media = 0;

            int teste;

            string nota = string.Empty;

            try

            {

                int notaCliente = (int)avaliacao.NotaClientes;

                int notaDesenv = (int)avaliacao.NotaDesenv;

                int notaDeveres = (int)avaliacao.NotaDeveres;

                int notaEquipe = (int)avaliacao.NotaEquipe;

                media = Convert.ToDouble((notaCliente + notaDesenv + notaDeveres + notaEquipe)) / 4;

                var truncado = Math.Round(media + 1,2);

                if (truncado > 3)

                {

                    nota = "A";

                }

                else if(truncado > 2 && truncado <= 3)

                {

                    nota = "B";

                }

                else

                {

                    nota = "C";

                }

            }

            catch (Exception ex)

            {

                throw ex;

            }

            return nota;

        }

    }

# Util

public class Criptografia

    {

        public static string GerarChaveHashMD5(string entrada)

        {

            MD5 chaveHash = MD5.Create();

            byte[] hash = chaveHash.ComputeHash(Encoding.UTF8.GetBytes(entrada));

            StringBuilder construtorDaHash = new StringBuilder();

            for (int i = 0; i < hash.Length; i++)

            {

                construtorDaHash.Append(hash[i].ToString("x2"));

            }

            return construtorDaHash.ToString();

        }

    }

public class Email

    {

        public bool EnviarEmail(string destinatario, string motivo, string comentario)

        {

            SmtpClient client = new SmtpClient("smtp.live.com");

            client.Port = 587;

            client.DeliveryMethod = SmtpDeliveryMethod.Network;

            client.UseDefaultCredentials = false;

            System.Net.NetworkCredential credentials =

                new System.Net.NetworkCredential("GerenciaRHBot@outlook.com", "GerenciaRH123");

            client.EnableSsl = true;

            client.Credentials = credentials;

            string corpo = "Informamos o seu desligamento pelo seguinte motivo:" + motivo + "\n Segue comentário do seu gestor:\n" + comentario;

            try

            {

                var mail = new MailMessage("GerenciaRHBot@outlook.com", destinatario);

                mail.Subject = "Aviso de Desligamento";

                mail.Body = corpo;

                client.Send(mail);

            }

            catch (Exception ex)

            {

                throw ex;

            }

            return true;

        }

    }

public class Enumeradores

    {

        enum TipoLog

        {

            Exclusao = 1,

            Inclusao = 2,

            Atualizacao = 3

        }

        enum TipoChat

        {

            Duvida = 1,

            Reclamacao = 2,

            Avaliacao = 3

        }

        enum TipoCargo

        {

            Gestor = 1,

            Gerente = 2,

            GerenteRH = 3,

            FuncionarioRH = 4,

            Funcionario = 5,

        }

    }

# Projeto WEB

public class AtualizarUsuarioController : Controller

    {

        UsuarioBusiness \_usrBusiness = new UsuarioBusiness();

        // GET: Desligamento

        public ActionResult AtualizarIndex(string racfUsuario)

        {

            return Json(new { redirect = "/AtualizarUsuario/Index?racfUsuario=" + racfUsuario + "" }, JsonRequestBehavior.AllowGet);

        }

        public ActionResult Index(string racfUsuario)

        {

            Usuario usuario = \_usrBusiness.BuscarUsuario(racfUsuario);

            UsuarioModel model = new UsuarioModel();

            model.Usuario = usuario;

            var tipoCargo = (ETipoCargo)usuario.IdCargo;

            Cargo cargo = new Cargo() { TipoCargo = tipoCargo };

            model.Cargo = cargo;

            return View(model);

        }

        public ActionResult UpdateUsuario(UsuarioModel model)

        {

            Usuario usuario = new Usuario();

            var idCargo = (int)model.Cargo.TipoCargo;

            usuario = \_usrBusiness.BuscarUsuario(model.Usuario.Racf);

            usuario.Email = model.Usuario.Email;

            usuario.CEP = model.Usuario.CEP;

            usuario.Endereco = model.Usuario.Endereco;

            usuario.NumeroEndereco = model.Usuario.NumeroEndereco;

            usuario.IdCargo = idCargo;

            bool sucesso = \_usrBusiness.UpdateUsuario(usuario);

            if (sucesso == true)

            {

                TempData["Sucesso"] = "U";

                return RedirectToAction("Index", "Home");

            }

            else

            {

                string mensagem = Resources.Mensagens.mensagemErroAvaliacao;

                TempData["Erro"] = mensagem;

                return RedirectToAction("Index", "Home");

            }

            return View();

        }

    }

public class AvaliacaoController : Controller

    {

        UsuarioBusiness \_usrBusiness = new UsuarioBusiness();

        AvaliacaoBusiness avaliacaoBusiness = new AvaliacaoBusiness();

        string mensagem = "";

        // GET: Avaliacao

        public ActionResult AvaliacaoIndex(string racfUsuario)

        {

            return Json(new { redirect = "/Avaliacao/Index?racfUsuario=" + racfUsuario +"" }, JsonRequestBehavior.AllowGet);

        }

        public ActionResult Index(string racfUsuario)

        {

            try

            {

                Usuario usuarioASerAvaliado = \_usrBusiness.BuscarUsuario(racfUsuario);

                UsuarioModel model = new UsuarioModel

                {

                    UsuarioAvaliacao = usuarioASerAvaliado

                };

                return View(model);

            }

            catch (Exception)

            {

                mensagem = Resources.Mensagens.mensagemErroAcessoAvaliacao;

                ViewData["Erro"] = mensagem;

                return RedirectToAction("Index", "Home");

            }

        }

        [HttpPost]

        public ActionResult EfetuarAvaliacao(UsuarioModel model, UsuarioLogado usuarioLogado,Usuario usuario)

        {

            try

            {

                model.Avaliacao.UsuarioAvaliador = usuarioLogado.usuarioLogado.IdUsuario;

                model.Avaliacao.UsuarioAvaliado = model.UsuarioAvaliacao.IdUsuario;

                bool avaliado = avaliacaoBusiness.InserirAvaliacao(model.Avaliacao);

                if (avaliado == true)

                {

                    TempData["Sucesso"] = "A";

                    return RedirectToAction("Index", "Home");

                }

                else

                {

                    mensagem = Resources.Mensagens.mensagemErroAvaliacao;

                    TempData["Erro"] = mensagem;

                    return RedirectToAction("Index", "Home");

                }

            }

            catch (Exception)

            {

                mensagem = Resources.Mensagens.mensagemErroBase;

                TempData["Erro"] = mensagem;

                return RedirectToAction("Index", "Home");

            }

        }

    }

public class CadastroController : Controller

    {

        // GET: Cadastro

        public ActionResult Index()

        {

            return View();

        }

        public ActionResult EfetuarCadastro(UsuarioModel model, UsuarioLogado usuarioLogado)

        {

            UsuarioBusiness usuarioBusiness = new UsuarioBusiness();

            ValidaLogin validaLogin = new ValidaLogin();

            var idCargo = (int)model.Cargo.TipoCargo;

            model.Usuario.IdCargo = idCargo;

            validaLogin = usuarioBusiness.CadastrarUsuario(model.Usuario, usuarioLogado.usuarioLogado.IdUsuario);

            if (validaLogin.Sucesso == true)

            {

                TempData["Sucesso"] = "S";

                return RedirectToAction("Index", "Home");

            }

            else

            {

                ViewBag.Sucesso = "N";

                ViewBag.Mensagem = validaLogin.Mensagem;

                return View("Index");

            }

        }

        public JsonResult ConsultaCEP(string CEP)

        {

            CepBusiness cepBusiness = new CepBusiness();

            var dados = cepBusiness.ConsultarEndereco(CEP);

            if(dados.Logradouro == null)

            {

                return Json("error");

            }

            return Json(dados);

        }

    }

public class DesligamentoController : Controller

   {

       UsuarioBusiness \_usrBusiness = new UsuarioBusiness();

       string mensagem = "";

       // GET: Desligamento

       public ActionResult DesligamentoIndex(string racfUsuario, string racfGestor)

       {

           return Json(new { redirect = "/Desligamento/Index?racfUsuario=" + racfUsuario + "&racfGestor=" + racfGestor + "" }, JsonRequestBehavior.AllowGet);

       }

       public ActionResult Index (string racfUsuario, string racfGestor)

       {

           try

           {

               Usuario usuarioDesligado = \_usrBusiness.BuscarUsuario(racfUsuario);

               Usuario gestor = \_usrBusiness.BuscarUsuario(racfGestor);

               UsuarioModel model = new UsuarioModel

               {

                   Usuario = gestor,

                   UsuarioDesligamento = usuarioDesligado

               };

               return View(model);

           }

           catch (Exception)

           {

               mensagem = Resources.Mensagens.mensagemErroBase;

               TempData["Erro"] = mensagem;

               return RedirectToAction("Home", "Index");

           }

       }

       //TODO - SALVAR NA BASE QUE USUARIO FOI DESLIGADO, RETORNANDO AVISO DE SUCESSO PRA VIEW ANTERIOR

       public ActionResult SalvarDesligamento(UsuarioModel model,UsuarioLogado usuarioLogado)

       {

           try

           {

               model.Usuario.Racf = model.UsuarioDesligamento.Racf;

               model.Usuario.IdDesligamento = usuarioLogado.usuarioLogado.IdUsuario;

               if (\_usrBusiness.DeletarUsuario(model.Usuario) == false)

               {

                   mensagem = Resources.Mensagens.mensagemErroDesligamento;

                   TempData["Erro"] = mensagem;

                   return RedirectToAction("Index","Home");

               }

               else

               {

                   Email email = new Email();

                   email.EnviarEmail(model.UsuarioDesligamento.Email, model.Usuario.Motivo, model.ComentarioDesligamento);

                   TempData["Sucesso"] = "D";

                   return RedirectToAction("Index", "Home");

               }

           }

           catch (Exception)

           {

               mensagem = Resources.Mensagens.mensagemErroBase;

               TempData["Erro"] = mensagem;

               return RedirectToAction("Index", "Home");

           }

       }

   }

public class HomeController : Controller

    {

        public ActionResult Index(string racf)

        {

            UsuarioModel modelUsuario = new UsuarioModel();

            AvaliacaoBusiness avaliacaoBusiness = new AvaliacaoBusiness();

            UsuarioBusiness business = new UsuarioBusiness();

            List<Avaliacao> listAvaliacao = new List<Avaliacao>();

            if (string.IsNullOrEmpty(racf) && Session["usuarioLogado"] != null)

            {

                modelUsuario.Usuario = (Usuario)Session["usuarioLogado"];

            }

            else

            {

                modelUsuario.Usuario = business.BuscarUsuario(racf);

            }

            if (modelUsuario.Usuario == null)

            {

                return RedirectToAction("Index", "Login");

            }

            modelUsuario.listUsuario = business.ListarUsuarios(modelUsuario.Usuario);

            listAvaliacao = avaliacaoBusiness.ListarAvaliacoes(modelUsuario.Usuario.IdUsuario);

            modelUsuario.QuantidadeA = listAvaliacao.Where(x => x.Nota == "A").Count();

            modelUsuario.QuantidadeB = listAvaliacao.Where(x => x.Nota == "B").Count();

            modelUsuario.QuantidadeC = listAvaliacao.Where(x => x.Nota == "C").Count();

            return View(modelUsuario);

        }

        public ActionResult PesquisarUsuario(string chave)

        {

            return PartialView();

        }

    }

public class LoginController : Controller

    {

        string mensagem = "";

        // GET: Login

        public ActionResult Index()

        {

            LoginModel model = new LoginModel();

            return View(model);

        }

        public ActionResult EfetuarLogin(LoginModel model)

        {

            try

            {

                LoginBusiness Loginbusiness = new LoginBusiness();

                bool loginEfetuado = Loginbusiness.ValidarLogin(model.Usuario.Racf, model.Usuario.Senha);

                if (loginEfetuado)

                {

                    UsuarioBusiness usuarioBusiness = new UsuarioBusiness();

                    Usuario userLogado = usuarioBusiness.BuscarUsuario(model.Usuario.Racf);

                    Session["usuarioLogado"] = userLogado;

                    return RedirectToAction("Index", "Home", new { racf = model.Usuario.Racf });

                }

                else

                {

                    mensagem = Resources.Mensagens.mensagemErroLogin;

                    TempData["Erro"] = mensagem;

                    return View("Index");

                }

            }

            catch (Exception)

            {

                mensagem = Resources.Mensagens.mensagemErroBase;

                TempData["Erro"] = mensagem;

                return View("Index");

            }

        }

    }

public class RelatorioController : Controller

    {

        // GET: Relatorio

        public ActionResult Index()

        {

            UsuarioBusiness usuarioBusiness = new UsuarioBusiness();

            var usuario = (Usuario)Session["usuarioLogado"];

            List<Usuario> usuarios = new List<Usuario>();

            RelatorioModel relatorioModel = new RelatorioModel();

            relatorioModel.listUsuarios = usuarioBusiness.Relatorio();

            usuarios = usuarioBusiness.ListarUsuarios(usuario);

            relatorioModel.Gerente = usuarios.Where(x => x.IdCargo == 2).ToList().Count;

            relatorioModel.GerenteRH = usuarios.Where(x => x.IdCargo == 3).ToList().Count;

            relatorioModel.FuncionarioRH = usuarios.Where(x => x.IdCargo == 4).ToList().Count;

            relatorioModel.Funcionario = usuarios.Where(x => x.IdCargo == 5).ToList().Count;

            relatorioModel.Salarios.SalarioFuncionario = (int)usuarios.Sum(x => x.Salario);

            relatorioModel.Salarios.SalarioFuncionarioRH = (int)usuarios.Sum(x => x.Salario);

            relatorioModel.Salarios.SalarioGerente = (int)usuarios.Sum(x => x.Salario);

            relatorioModel.Salarios.SalarioGerenteRH = (int)usuarios.Sum(x => x.Salario);

            relatorioModel.Total = relatorioModel.Salarios.SalarioFuncionario + relatorioModel.Salarios.SalarioFuncionarioRH + relatorioModel.Salarios.SalarioGerente + relatorioModel.Salarios.SalarioGerenteRH;

            return View(relatorioModel);

        }

    }

public class LoginModel

    {

        public Usuario Usuario { get; set; }

        public string Mensagem { get; set; }

        public LoginModel()

        {

            this.Usuario = new Usuario();

            this.Mensagem = "";

        }

    }

public class LogModel

    {

        public Log Log{ get; set; }

        public List<Usuario> Usuarios { get; set; }

        public List<Log> Logs{ get; set; }

        public string Pesquisa { get; set; }

        public DateTime DtInicio { get; set; }

        public DateTime DtFim { get; set; }

    }

public class RelatorioModel

    {

        public List<Usuario> listUsuarios { get; set; }

        public Salarios Salarios { get; set; }

        public int GerenteRH { get; set; }

        public int Gerente { get; set; }

        public int FuncionarioRH { get; set; }

        public int Funcionario { get; set; }

        public int Total { get; set; }

    }

public class UsuarioLogado

    {

        public Usuario usuarioLogado

        { get

            {

                if (HttpContext.Current.Session != null && HttpContext.Current.Session["usuarioLogado"] != null)

                {

                    return HttpContext.Current.Session["usuarioLogado"] as Usuario;

                }

                else

                {

                    return new Usuario();

                }

            }

            set

            {

                HttpContext.Current.Session["usuarioLogado"] = value;

            }

        }

    }

public class UsuarioModel

    {

        public Avaliacao Avaliacao { get; set; }

        public Usuario Usuario { get; set; }

        public Usuario UsuarioDesligamento { get; set; }

        public Usuario UsuarioAvaliacao { get; set; }

        public List<Usuario> listUsuario { get; set; }

        public UsuarioLogado usuarioLogado { get; set; }

        public Cargo Cargo { get; set; }

        public int QuantidadeA { get; set; }

        public int QuantidadeB { get; set; }

        public int QuantidadeC { get; set; }

        public string ComentarioDesligamento { get; set; }

        public string ComentarioAvaliacaoGestor { get; set; }

        public int CargoSelecionado { get; set; }

    }

Script.JS

$(document).ready(function () {

    $("#btnLimparUm").on('click', function () {

        document.getElementById('txtNomeUsuario').value = "";

        document.getElementById('txtRacf').value = "";

        document.getElementById('txtCpf').value = "";

        document.getElementById('txtPis').value = "";

        document.getElementById('txtRazao').value = "";

        //document.getElementById('txtCargo').value = "Selecione";

        //$("#txtCargo").val('Selecione');

        $('#txtCargo').prop('selectedIndex', 0);

    });

    $("#btnLimparDois").on('click', function () {

        document.getElementById('txtCep').value = "";

        document.getElementById('txtEmail').value = "";

        document.getElementById('txtEndereco').value = "";

        document.getElementById('txtNumeroEndereco').value = "";

    });

    $("#btnPesquisar").click(function () {

        var CEP = $("#txtCep").val();

        $.ajax({

            type: "POST",

            url: "/Cadastro/ConsultaCEP",

            data: { CEP: CEP },

            success: function (data) {

                teste = data;

                if (data != "error") {

                    document.getElementById('txtEndereco').value = teste.Logradouro + ", " + teste.Bairro + ", " + teste.Localidade;

                }

                else {

                    alert('Erro na consulta do CEP');

                }

            },

        });

    });

    $("#btnPesquisarUsuario").keyup(function () {

        var value = $(this).val().toLowerCase();

        $("#tbUsuarios tr").filter(function () {

            $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)

        });

    });

    $('#modalSucesso').modal();

    $(document).on("input", "#comentarioColaboradorDesligado", function () {

        var limite = 401;

        var informativo = "caracteres restantes.";

        var caracteresDigitados = $(this).val().length;

        var caracteresRestantes = limite - caracteresDigitados;

        if (caracteresRestantes <= 0) {

            var comentario = $("textarea[name=comentario]").val();

            $("textarea[name=comentario]").val(comentario.substr(0, limite));

            $(".caracteres").text("0 " + informativo);

        } else {

            $(".caracteres").text(caracteresRestantes + " " + informativo);

        }

    });

    $(document).on("input", "#comentarioAvaliacao", function () {

        var limite = 1000;

        var informativo = "caracteres restantes.";

        var caracteresDigitados = $(this).val().length;

        var caracteresRestantes = limite - caracteresDigitados;

        if (caracteresRestantes <= 0) {

            var comentario = $("textarea[name=comentario]").val();

            $("textarea[name=comentario]").val(comentario.substr(0, limite));

            $(".caracteres").text("0 " + informativo);

        } else {

            $(".caracteres").text(caracteresRestantes + " " + informativo);

        }

    });

});

Content.CSS

body {

    padding-top: 50px;

    padding-bottom: 20px;

}

/\* Set padding to keep content from hitting the edges \*/

.body-content {

    padding-left: 15px;

    padding-right: 15px;

}

/\* Override the default bootstrap behavior where horizontal description lists

   will truncate terms that are too long to fit in the left column

\*/

.dl-horizontal dt {

    white-space: normal;

}

/\* Set width on the form input elements since they're 100% wide by default \*/

input,

select,

textarea {

    max-width: 280px;

}

/\*TELA DE INDEX DO LOGIN E HOME\*/

.botaoMenu {

    padding-left: 2rem;

    padding-right: 2rem;

}

.cardColaborador {

    background-color: #FDFDFD;

    border: 2px solid #8492A6;

    border-radius: 28px;

    width: 610px;

    height: 139px;

}

.separador {

    border: 1px solid #7E55F3;

}

.barraPesquisaColab {

    border: 2px solid #8492A6;

    width: 633px;

    height: 52px;

}

.GraficoNota{

    width:500px;

    height:500px;

}

.CardFundoAnalista {

    background-color: #F4F8F6;

    border: 2px solid #8492A6;

    border-radius: 25px;

    width: 615px;

    height: 600px;

}

/\*TELA DE INDEX DE CADASTRO\*/

.cardFundoCadastro {

    background-color: #FFFFFF;

    border: 2px solid #8492A6;

    border-radius: 25px;

    width: 1160px;

    height: 554px;

}

.cardInformacoes {

    background-color: #FFFFFF;

    border: 1px solid #8492A6;

    border-radius: 10px;

    width: 845px;

    height: 258px;

}

/\*HIGHLIGHT DE TELA ATUAL SELECIONADA\*/

a {

    text-decoration: none;

    color: #000;

}

    a:hover, a:focus {

        color: #8492A6;

        text-decoration: none;

        outline: none;

    }

.active {

    font-size: 30px;

    color: #8492A6;

    border-bottom: 5px solid #8492A6;

}

@media(max-width:1468px){

    .cardColaborador {

        background-color: #FDFDFD;

        border: 2px solid #8492A6;

        border-radius: 28px;

        width:auto;

        height:auto;

    }

    #GraficoNota{

        width:425px;

        height:425px;

    }

    .CardFundoAnalista {

        background-color: #F4F8F6;

        border: 2px solid #8492A6;

        border-radius: 25px;

        width: auto;

        height: auto;

    }

    .cardFundoCadastro {

        background-color: #FFFFFF;

        border: 2px solid #8492A6;

        border-radius: 10px;

        height: auto;

    }

    .form-control .border{

        width:auto;

        height:auto;

    }

    .row {

        width: auto;

        height: auto;

    }

}

@media(max-width:612px){

    .CardDesligamentoComentario {

        background-color: #FCFFFD;

        border: 2px solid #8492A6;

        border-radius: 15px;

        width: auto;

    }

}

/\*TOOLTIP VIA CSS\*/

.tooltipCSS {

    position: relative;

    display: inline-block;

}

    .tooltipCSS .tooltiptextCSS {

        visibility: hidden;

        width: 120px;

        background-color: black;

        color: #fff;

        text-align: center;

        border-radius: 6px;

        padding: 5px 0;

        /\* Position the tooltip \*/

        position: absolute;

        z-index: 1;

    }

    .tooltipCSS:hover .tooltiptextCSS {

        visibility: visible;

    }

/\*CSS DESLIGAMENTO\*/

.CardFundoDesligamento {

    background-color: #FFFFFF;

    border: 2px solid #8492A6;

    border-radius: 25px;

}

.CardDesligamentoComentario {

    background-color: #FCFFFD;

    border: 2px solid #8492A6;

    border-radius: 15px;

}

/\*TOASTER CSS\*/

#toast-container.toast-top-full-width > div {

    width: 96%;

    margin-left: auto;

    margin-right: auto;

}

Atualizar Usuario - Tela

@model GerenciaRH.Model.UsuarioModel

@using Util.Enums

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

}

@using (Html.BeginForm("UpdateUsuario", "AtualizarUsuario", FormMethod.Post))

{

    @Html.HiddenFor(x => x.Usuario.Racf)

    <div class="col-md-12 col-xl-12">

        <div class="row">

            <div class="col-md-12">

                <h2 class="font-weight-bold">Informações do Usuário</h2>

                <span>Utilize essa parte do sistema atualizar as informações do usuário</span>

            </div>

        </div>

        <div class="row pl-3 pt-4">

            <div class="col-md-5">

                <div class="row CardDesligamentoComentario">

                    <div class="col-md-2 p-3">

                        <img src="~/Resources/Icons/pessoa\_@Model.Usuario.IdUsuario .png" height="100" width="100" />

                    </div>

                    <div class="col-md-7 pt-3 p-3">

                        <table>

                            <tr>

                                <td class="p-2">Nome: @Model.Usuario.NomeUsuario</td>

                            </tr>

                            <tr>

                                <td class="p-2">RACF: @Model.Usuario.Racf</td>

                            </tr>

                        </table>

                    </div>

                </div>

            </div>

        </div>

        <div class="row pt-3">

            <div class="col-md-12">

                <div class="col-md-10 CardFundoDesligamento ">

                    <div class="row">

                        <span class="pl-4 pt-2">Quesitos a serem avaliados:</span>

                    </div>

                    <div class="row">

                        <div class="col-md-3 pt-3">

                            <div class="pl-3">

                                @Html.LabelFor(x => x.Usuario.NomeCargo, new { @class = "pb-2" })

                                @Html.DropDownListFor(o => o.Cargo.TipoCargo, Enum.GetValues(typeof(ETipoCargo)).Cast<ETipoCargo>

                                                  ().Select(x => new SelectListItem { Text = x.ToString(), Value = ((int)x).ToString() }), @Model.Cargo.NomeCargo, htmlAttributes: new { @class = "form-control border pl-2", @id = "txtCargo2" })

                            </div>

                        </div>

                    </div>

                    <div class="row">

                        <div class="col-md-2">

                            <div class="pl-1 p-4">

                                @Html.LabelFor(x => x.Usuario.CEP, new { @class = "pb-2" })

                                <br />

                                <br />

                                <br />

                                @Html.LabelFor(x => x.Usuario.Email, new { @class = "pb-2" })

                            </div>

                        </div>

                        <div class="col-md-3 pt-3">

                            <div>

                                @Html.TextBoxFor(x => x.Usuario.CEP, new { @class = "form-control border", @id = "txtCep" })

                                <br />

                                <br />

                                @Html.TextBoxFor(x => x.Usuario.Email, new { @class = "form-control border", @style = "width:600px", @id = "txtEmail" })

                            </div>

                        </div>

                        <div class="col-md-2">

                            <div class="pt-3">

                                @Html.LabelFor(x => x.Usuario.Endereco, new { @class = "pb-2" })

                                @Html.LabelFor(x => x.Usuario.NumeroEndereco, new { @class = "pb-2" })

                            </div>

                        </div>

                        <div class="col-md-3 pt-3">

                            <div>

                                @Html.TextBoxFor(x => x.Usuario.Endereco, new { @class = "form-control border", @id = "txtEndereco" })

                                @Html.TextBoxFor(x => x.Usuario.NumeroEndereco, new { @class = "form-control border", @id = "txtNumeroEndereco" })

                            </div>

                        </div>

                    </div>

                    <div class="row">

                        <div class="col-md-9">

                        </div>

                        <div class="col-md-3">

                            <button class="btn btn-dark" type="button" data-toggle="modal" data-target="#modalCancelarAvaliacao">Cancelar</button>

                            <button class="btn btn-success" type="submit">Atualizar</button>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

}

<!--MODAL-->

<div class="modal fade" id="modalCancelarAvaliacao" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" data-backdrop="false">

    <div class="modal-dialog" role="document">

        <div class="modal-content">

            <div class="modal-header">

                <h5 class="modal-title" id="exampleModalLabel">Cancelar Avaliação?</h5>

            </div>

            <div class="modal-body">

                <p>Tem certeza que deseja cancelar a alteração?</p>

            </div>

            <div class="modal-footer">

                <button type="button" class="btn btn-primary" onclick="location.href='@Url.Action("Index","Home")'">Retornar para home</button>

                <button type="button" class="btn btn-secondary" data-dismiss="modal">Continuar</button>

            </div>

        </div>

    </div>

</div>

Avaliacao – Tela

@model GerenciaRH.Model.UsuarioModel

@using Util.Enums

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

}

@using (Html.BeginForm("EfetuarAvaliacao", "Avaliacao", FormMethod.Post))

{

    @Html.HiddenFor(x => x.UsuarioAvaliacao.IdUsuario)

    @Html.HiddenFor(x => x.Usuario.Racf)

    <div class="col-md-12 col-xl-12">

        <div class="row">

            <div class="col-md-12">

                <h2 class="font-weight-bold">Avaliação</h2>

                <span>Utiliza essa parte do sistema para avaliar o funcionário</span>

            </div>

        </div>

        <div class="row pl-3 pt-4">

            <div class="col-md-5">

                <div class="row CardDesligamentoComentario">

                    <div class="col-md-2 p-3">

                        <img src="~/Resources/Icons/pessoa\_@Model.UsuarioAvaliacao.IdUsuario .png" height="100" width="100" />

                    </div>

                    <div class="col-md-7 pt-3 p-3">

                        <table>

                            <tr>

                                <td class="p-2">Nome: @Model.UsuarioAvaliacao.NomeUsuario</td>

                            </tr>

                            <tr>

                                <td class="p-2">Cargo: @Model.UsuarioAvaliacao.NomeCargo</td>

                            </tr>

                            <tr>

                                <td class="p-2">Racf: @Model.UsuarioAvaliacao.Racf</td>

                            </tr>

                        </table>

                    </div>

                </div>

            </div>

        </div>

        <div class="row pt-3">

            <div class="col-md-12">

                <div class="col-md-10 CardFundoDesligamento ">

                    <div class="row">

                        <span class="pl-4 pt-2">Quesitos a serem avaliados:</span>

                    </div>

                    <div class="row pl-4 p-2">

                        <div class="col-md-4">

                            <h4>Entrega de deveres:</h4>

                        </div>

                        <div class="col-md-8">

                            @Html.DropDownListFor(x => x.Avaliacao.NotaDeveres, new SelectList(Enum.GetValues(typeof(ETipoNota))), "Selecione a nota...", new { @class = "form-control" })

                        </div>

                    </div>

                    <div class="row pl-4 p-2">

                        <div class="col-md-4">

                            <h4>Trabalho em Equipe</h4>

                        </div>

                        <div class="col-md-8">

                            @Html.DropDownListFor(x => x.Avaliacao.NotaEquipe, new SelectList(Enum.GetValues(typeof(ETipoNota))), "Selecione a nota...", new { @class = "form-control" })

                        </div>

                    </div>

                    <div class="row pl-4 p-2">

                        <div class="col-md-4">

                            <h4>Desenvolvimento pessoal:</h4>

                        </div>

                        <div class="col-md-8">

                            @Html.DropDownListFor(x => x.Avaliacao.NotaDesenv, new SelectList(Enum.GetValues(typeof(ETipoNota))), "Selecione a nota...", new { @class = "form-control" })

                        </div>

                    </div>

                    <div class="row pl-4 p-2">

                        <div class="col-md-4">

                            <h4>Relacionamento com o cliente:</h4>

                        </div>

                        <div class="col-md-8">

                            @Html.DropDownListFor(x => x.Avaliacao.NotaClientes, new SelectList(Enum.GetValues(typeof(ETipoNota))), "Selecione a nota...", new { @class = "form-control" })

                        </div>

                    </div>

                    <div class="row pl-4 p-2">

                        <div class="col-md-2">

                            <h4>Comentário do Gestor:</h4>

                        </div>

                    </div>

                    <div class="row pl-4 pb-4">

                        <div class="col-md-6 CardDesligamentoComentario">

                            @Html.TextAreaFor(x => x.ComentarioAvaliacaoGestor, new { @class = "form-control col-md-12", @rows = "8", @id = "comentarioAvaliacao" })

                            <p><label for="review"></label> <small class="caracteres"></small></p>

                        </div>

                    </div>

                    <div class="row">

                        <div class="col-md-9">

                        </div>

                        <div class="col-md-3">

                            <button class="btn btn-dark" type="button" data-toggle="modal" data-target="#modalCancelarAvaliacao">Cancelar</button>

                            <button class="btn btn-success" type="submit">Avaliar</button>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

}

<!--MODAL-->

<div class="modal fade" id="modalCancelarAvaliacao" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" data-backdrop="false">

    <div class="modal-dialog" role="document">

        <div class="modal-content">

            <div class="modal-header">

                <h5 class="modal-title" id="exampleModalLabel">Cancelar Avaliação?</h5>

            </div>

            <div class="modal-body">

                <p>Tem certeza que deseja cancelar a avaliação?</p>

            </div>

            <div class="modal-footer">

                <button type="button" class="btn btn-primary" onclick="location.href='@Url.Action("Index","Home")'">Retornar para home</button>

                <button type="button" class="btn btn-secondary" data-dismiss="modal">Continuar</button>

            </div>

        </div>

    </div>

</div>

Cadastro – Tela

@model GerenciaRH.Model.UsuarioModel

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@using Util.Enums;

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

    <script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>

}

<body>

    @if (ViewBag.Sucesso != null)

    {

        <div class="modal fade" id="modalSucesso" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" data-backdrop="false">

            <div class="modal-dialog" role="document">

                <div class="modal-content">

                    <div class="modal-header">

                        <h5 class="modal-title" id="exampleModalLabel">Aviso do Sistema</h5>

                        <button type="button" class="close" data-dismiss="modal" aria-label="Close">

                            <span aria-hidden="true">&times;</span>

                        </button>

                    </div>

                    <div class="modal-body">

                        <a>@ViewBag.Mensagem</a>

                    </div>

                    <div class="modal-footer">

                        <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>

                    </div>

                </div>

            </div>

        </div>

    }

    <div class="col-xl-12 col-md-12">

        <div class="row">

            <div class="col-md-6">

                <div class="pl-4">

                    <h3 class="font-weight-bold">

                        Cadastro de Colaboradores

                    </h3>

                    <span>Você pode cadastrar seus colaboradores nessa parte do sistema</span>

                </div>

            </div>

        </div>

        <div class="row">

            @using (Html.BeginForm("EfetuarCadastro", "Cadastro", FormMethod.Post))

            {

                <div class="col-md-12 pt-3 pl-5">

                    <div class="cardFundoCadastro">

                        <div class="row">

                            <div class="col-md-3">

                                <div class="p-3">

                                    @Html.ValidationMessageFor(x => x.Usuario.NomeUsuario, "", new { @class = "text-danger" })

                                    <br />

                                    @Html.ValidationMessageFor(x => x.Usuario.Senha, "", new { @class = "text-danger" })

                                    <br />

                                    @Html.ValidationMessageFor(x => x.Usuario.CPFCNPJ, "", new { @class = "text-danger" })

                                    <br />

                                    @Html.ValidationMessageFor(x => x.Usuario.TipoDocumento, "", new { @class = "text-danger" })

                                    <br />

                                    @Html.ValidationMessageFor(x => x.Usuario.NomeCargo, "", new { @class = "text-danger" })

                                </div>

                            </div>

                            <div class="col-md-8 pt-2">

                                <div class="cardInformacoes">

                                    <div class="row">

                                        <div class="col-md-3 pl-4 pt-2">

                                            <span>Informações Gerais</span>

                                        </div>

                                        <div class="col-md-9">

                                            <div class="float-right pr-2">

                                                <button class="btn btn-dark" id="btnLimparUm" type="button">Limpar</button>

                                            </div>

                                        </div>

                                    </div>

                                    <div class="row">

                                        <div class="col-md-2">

                                            <div class="pl-1 p-4">

                                                @Html.LabelFor(x => x.Usuario.NomeUsuario, new { @class = "pb-2" })

                                                @Html.LabelFor(x => x.Usuario.NomeCargo, new { @class = "pb-2" })

                                                @Html.LabelFor(x => x.Usuario.Racf, new { @class = "pb-2" })

                                                @Html.LabelFor(x => x.Usuario.Senha, new { @class = "pb-2" })

                                            </div>

                                        </div>

                                        <div class="col-md-3 pt-3">

                                            <div class="pl-3">

                                                @Html.TextBoxFor(x => x.Usuario.NomeUsuario, new { @class = "form-control border", @id = "txtNomeUsuario" })

                                                @Html.DropDownListFor(o => o.Cargo.TipoCargo, Enum.GetValues(typeof(ETipoCargo)).Cast<ETipoCargo>

                                                    ().Select(x => new SelectListItem { Text = x.ToString(), Value = ((int)x).ToString() }), "Selecione", htmlAttributes: new { @class = "form-control border pl-2", @id = "txtCargo" })

                                                @Html.TextBoxFor(x => x.Usuario.Racf, new { @class = "form-control border", @id = "txtRacf" })

                                                @Html.TextBoxFor(x => x.Usuario.Senha, new { @class = "form-control border", @id = "txtSenha", @maxlength = "6", @type = "password" })

                                            </div>

                                        </div>

                                        <div class="col-md-2">

                                            <div class="pl-1 p-4">

                                                @Html.LabelFor(x => x.Usuario.TipoDocumento, new { @class = "pb-2" })

                                                <br />

                                                @Html.LabelFor(x => x.Usuario.CPFCNPJ, new { @class = "pb-2" })

                                                @Html.LabelFor(x => x.Usuario.PIS, new { @class = "pb-2" })

                                            </div>

                                        </div>

                                        <div class="col-md-3 pt-3">

                                            <div class="pl-3">

                                                @Html.DropDownListFor(x => x.Usuario.TipoDocumento, Enum.GetValues(typeof(ETipoDocumento)).Cast<ETipoDocumento>

                                                   ().Select(x => new SelectListItem { Text = x.ToString(), Value = ((int)x).ToString() }), "Selecione", htmlAttributes: new { @class = "form-control border pl-2", @id = "txtTipoDocumento" })

                                                @Html.TextBoxFor(x => x.Usuario.CPFCNPJ, new { @class = "form-control border", @id = "txtCpf" })

                                                @Html.TextBoxFor(x => x.Usuario.PIS, new { @class = "form-control border", @id = "txtPis" })

                                            </div>

                                        </div>

                                    </div>

                                </div>

                            </div>

                        </div>

                        <div class="row">

                            <div class="col-md-3">

                                <div class="p-3">

                                    @Html.ValidationMessageFor(x => x.Usuario.Endereco, "", new { @class = "text-danger" })

                                    <br />

                                    @Html.ValidationMessageFor(x => x.Usuario.NumeroEndereco, "", new { @class = "text-danger" })

                                </div>

                            </div>

                            <div class="col-md-8 pt-2">

                                <div class="cardInformacoes">

                                    <div class="row">

                                        <div class="col-md-3 pl-4 pt-2">

                                            <span>Endereço/Outros</span>

                                        </div>

                                        <div class="col-md-9">

                                            <div class="float-right pr-2">

                                                <button class="btn btn-dark" id="btnLimparDois" type="button">Limpar</button>

                                            </div>

                                        </div>

                                    </div>

                                    <div class="row">

                                        <div class="col-md-2">

                                            <div class="pl-1 p-4">

                                                @Html.LabelFor(x => x.Usuario.CEP, new { @class = "pb-2" })

                                                <br />

                                                <br />

                                                <br />

                                                @Html.LabelFor(x => x.Usuario.Email, new { @class = "pb-2" })

                                            </div>

                                        </div>

                                        <div class="col-md-3 pt-3">

                                            <div>

                                                @Html.TextBoxFor(x => x.Usuario.CEP, new { @class = "form-control border", @id = "txtCep" })

                                                <button class="btn btn-success" id="btnPesquisar" type="button">Pesquisar CEP</button>

                                                <br />

                                                <br />

                                                @Html.TextBoxFor(x => x.Usuario.Email, new { @class = "form-control border", @style = "width:600px", @id = "txtEmail" , @type = "email" })

                                            </div>

                                        </div>

                                        <div class="col-md-2">

                                            <div class="pt-3">

                                                @Html.LabelFor(x => x.Usuario.Endereco, new { @class = "pb-2" })

                                                @Html.LabelFor(x => x.Usuario.NumeroEndereco, new { @class = "pb-2" })

                                            </div>

                                        </div>

                                        <div class="col-md-3 pt-3">

                                            <div>

                                                @Html.TextBoxFor(x => x.Usuario.Endereco, new { @class = "form-control border", @id = "txtEndereco" })

                                                @Html.TextBoxFor(x => x.Usuario.NumeroEndereco, new { @class = "form-control border", @id = "txtNumeroEndereco" })

                                                <div class="pt-5" style="padding-left:200px;">

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

                                                </div>

                                            </div>

                                        </div>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            }

        </div>

    </div>

</body>

<script>

    $(document).ready(function ($) {

        if ($('#txtTipoDocumento').val() == "") {

            $('#txtCpf').prop('disabled', true)

        }

        $('#txtTipoDocumento').on('change', function () {

            if ($('#txtTipoDocumento').val() == 1) {

                $('#txtCpf').mask('000.000.000-00')

                $('#txtCpf').prop('disabled', false)

            }

            if ($('#txtTipoDocumento').val() == 2) {

                $('#txtCpf').mask('00.000.000/0000-00')

                $('#txtCpf').prop('disabled', false)

            }

            if ($('#txtTipoDocumento').val() == "") {

                $('#txtCpf').val('');

                $('#txtCpf').prop('disabled', true)

            }

        })

            $('#txtPis').mask('000.00000.00-0')

            $('#txtCep').mask('00000-000')

    });

</script>

Desligamento – Tela

@model GerenciaRH.Model.UsuarioModel

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

}

@using (Html.BeginForm("SalvarDesligamento", "Desligamento", FormMethod.Post))

{

    @Html.HiddenFor(x => x.UsuarioDesligamento.Racf)

    @Html.HiddenFor(x => x.Usuario.Racf)

    @Html.HiddenFor(x => x.UsuarioDesligamento.Email)

    @Html.HiddenFor(x => x.Usuario.Email)

    <div class="col-md-12 col-xl-12">

        <div class="col-md-12 CardFundoDesligamento">

            <div class="row">

                <div class="col-md-12">

                    <h2 class="font-weight-bold">Desligamento</h2>

                </div>

            </div>

            <div class="row">

                <div class="pl-3 pt-3">

                    <img src="~/Resources/Icons/pessoa\_@Model.UsuarioDesligamento.IdUsuario .png" width="100" height="100" style="border-radius:50%" />

                </div>

                <div class="pl-3 mt-2">

                    <table>

                        <tr>

                            <td class="p-2">Nome: @Model.UsuarioDesligamento.NomeUsuario</td>

                        </tr>

                        <tr>

                            <td class="p-2">Cargo: @Model.UsuarioDesligamento.NomeCargo</td>

                        </tr>

                        <tr>

                            <td class="p-2">Racf: @Model.UsuarioDesligamento.Racf

                        </tr>

                    </table>

                </div>

            </div>

            <div class="row">

                <div class="pl-3">

                    <h4>Motivo de desligamento:</h4>

                    @Html.TextBoxFor(x => x.Usuario.Motivo, new { @class = "form-control border", @id = "txtMotivo", @required = true })

                </div>

            </div>

            <div class="row pt-3">

                <div class="col-md-12">

                    <div class="row">

                        <div class="col-md-6 pl-3 pb-3">

                            <form>

                                <label for="comentarioColaboradorDesligado">Comentário de desligamento:</label>

                                <div class="form-group CardDesligamentoComentario col-md-8">

                                    @Html.TextAreaFor(x => x.ComentarioDesligamento, new { @class = "form-control col-md-12", @rows = "6", @id = "comentarioColaboradorDesligado", @name = "comentario", @maxlength = "400" })

                                </div>

                            </form>

                            <p><label for="review"></label> <small class="caracteres"></small></p>

                        </div>

                        <div class="col-md-6">

                            <label for="cardRequerente">Requerido por:</label>

                            <div class="row">

                                <div class="row CardDesligamentoComentario" id="cardRequerente">

                                    <div class="pt-2">

                                        <img src="~/Resources/Icons/pessoa\_@Model.Usuario.IdUsuario .png" width="100" height="100" style="border-radius:50%" />

                                    </div>

                                    <div class="pl-4">

                                        <table>

                                            <tr>

                                                <td class="p-2">Nome: @Model.Usuario.NomeUsuario</td>

                                            </tr>

                                            <tr>

                                                <td class="p-2">Cargo: @Model.Usuario.NomeCargo</td>

                                            </tr>

                                            <tr>

                                                <td class="p-2">Racf: @Model.Usuario.Racf

                                            </tr>

                                        </table>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

            <div class="row">

                <div class="col-md-12">

                    <div class="row pb-3">

                        <div class="col-md-10">

                        </div>

                        <div class="col-md-2">

                            <button class="btn btn-success" type="button" data-toggle="modal" data-target="#modalDesejaCancelar">Cancelar</button>

                            <button class="btn btn-danger" type="submit">Desligar</button>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!--MODAL-->

    <div class="modal fade" id="modalDesejaCancelar" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true" data-backdrop="false">

        <div class="modal-dialog" role="document">

            <div class="modal-content">

                <div class="modal-header">

                    <h5 class="modal-title" id="exampleModalLabel">Cancelar Desligamento?</h5>

                </div>

                <div class="modal-body">

                    <p>Tem certeza que deseja cancelar o desligamento?</p>

                </div>

                <div class="modal-footer">

                    <button type="button" class="btn btn-primary" onclick="location.href='@Url.Action("Index","Home")'">Retornar para home</button>

                    <button type="button" class="btn btn-secondary" data-dismiss="modal">Continuar</button>

                </div>

            </div>

        </div>

    </div>

}

Home – Tela

@model GerenciaRH.Model.UsuarioModel

@using Chart.Mvc.ComplexChart;

@using Chart.Mvc.Extensions

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

    <script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>

}

<script type="text/javascript" src="~/Scripts/script.js"></script>

<script src="~/Scripts/Chart.js"></script>

<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

    <div class="col-xl-12 col-md-12">

        <div class="row">

            <div class="col-md-6">

                <div class="pl-4">

                    <h3 class="font-weight-bold">

                        Avaliação de Colaboradores

                    </h3>

                    <span>Aqui é mostrado o gráfico e seus colaboradores</span>

                </div>

                <div id="cardColaboradores" class="pl-5 pt-5 animated bounce">

                    <div class="cardColaborador">

                        <div class="col-12">

                            <div class="row">

                                <div class="col-md-3 pt-2">

                                    <img src="~/Resources/Icons/pessoa\_@Model.Usuario.IdUsuario .png" width="120" height="120" style="border-radius:50%" />

                                </div>

                                <div class="separador">

                                </div>

                                @\*TABELAS COM INFORMAÇÕES DA MODEL\*@

                                <div class="col-md-2">

                                    <div class="pt-2">

                                        <table>

                                            <tr>

                                                <td class="pb-2">

                                                    Nome:

                                                </td>

                                            </tr>

                                            <tr>

                                                <td class="pb-2">

                                                    Cargo:

                                                </td>

                                            </tr>

                                            <tr>

                                                <td class="pb-2">

                                                    Racf:

                                                </td>

                                            </tr>

                                        </table>

                                    </div>

                                </div>

                                <div class="col-md-5">

                                    <div class="pt-2">

                                        <table>

                                            <tr>

                                                <td class="pb-2 font-weight-bold">

                                                    @Model.Usuario.NomeUsuario

                                                </td>

                                            </tr>

                                            <tr>

                                                <td class="pb-2 font-weight-bold">

                                                    @Model.Usuario.NomeCargo

                                                </td>

                                            </tr>

                                            <tr>

                                                <td class="pb-2 font-weight-bold">

                                                    @Model.Usuario.Racf

                                                </td>

                                            </tr>

                                        </table>

                                    </div>

                                </div>

                            </div>

                        </div>

                    </div>

                <label for="#GraficoNota" class="font-weight-bold mt-3">Média da nota de avaliações anuais</label>

                </div>

                @\*COLOCAR GRAFICO AQUI POR FAVOR\*@

                @{

                    var notas = new[] { "A", "B", "C" };

                    var qtdNotas = new List<double> { @Model.QuantidadeA, Model.QuantidadeB, Model.QuantidadeC };

                    var barChart = new BarChart();

                    barChart.ComplexData.Labels.AddRange(notas);

                    barChart.ComplexData.Datasets.AddRange(new List<ComplexDataset>

                                                                                                                                            {

                                                              new ComplexDataset

                                                              {

                                                                      Data = qtdNotas,

                                                                      Label = "Notas dos Funcionarios",

                                                                      FillColor = "rgba(220,220,0,0.2)",

                                                                      StrokeColor = "rgba(220,220,0,1)",

                                                                      PointColor = "rgba(220,220,220,1)",

                                                                      PointHighlightStroke = "rgba(220,220,220,1)",

                                                              }

                                                    });

                }

                <canvas id="GraficoNota" @\*width="500" height="500"\*@ class="GraficoNota pl-5 pt-5"></canvas>

                @Html.CreateChart("GraficoNota", barChart)

            </div>

            <div class="col-md-6">

                <div id="pesquisaUsuario" class="pt-5">

                    <div class="row">

                        <div>

                            <input type="text" class="form-control pr-5" placeholder="Pesquisar nome ou racf" name="pesquisaUsuario" maxlength="10" id="btnPesquisarUsuario" />

                        </div>

                        @\*COLOCAR SCRIPT DE PESQUISA EM DOC OU AQUI E REFERENCIAR\*@

                        <div>

                            <img src="~/Resources/Icons/lupa.png" width="30" height="30" onclick="PesquisarUsuario()" />

                        </div>

                    </div>

                </div>

                <div class="mt-5" style="height: 550px; overflow: auto;">

                    <div>

                        <div id="cardsSubordinados">

                            <div>

                                @Html.Partial("\_CardUsuarios")

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

</body>

</html>

<script>

    $(document).ready(function () {

        if ('@TempData["Erro"]' != "") {

            ToastrOption();

            toastr.error('@TempData["Erro"]');

        }

        switch ('@TempData["Sucesso"]') {

            case "D":

                ToastrOption();

                toastr.success('Usuário Desligado com Sucesso');

                break;

            case "S":

                ToastrOption();

                toastr.success('Usuário Cadastrado com Sucesso');

                break;

            case "A":

                ToastrOption();

                toastr.success('Usuário Avaliado com Sucesso');

                break;

            case "U":

                ToastrOption();

                toastr.success('Usuário atualizado com Sucesso');

                break;

        }

        function ToastrOption() {

            toastr.options = {

                "closeButton": true,

                "debug": false,

                "newestOnTop": false,

                "progressBar": true,

                "positionClass": "toast-top-left",

                "preventDuplicates": false,

                "onclick": null,

                "showDuration": "300",

                "hideDuration": "1000",

                "timeOut": "5000",

                "extendedTimeOut": "1000",

                "showEasing": "swing",

                "hideEasing": "linear",

                "showMethod": "fadeIn",

                "hideMethod": "fadeOut"

            }

        }

    });

function EfetuarDesligamento(racfUsuario, racfGestor) {

    $.ajax({

        type: "GET",

        url: '@Url.Action("DesligamentoIndex", "Desligamento")',

        data: { racfUsuario, racfGestor },

        success: function (data) {

        window.location.href = data.redirect;

        }

        })

        }

        function EfetuarAvaliacao(racfUsuario) {

        $.ajax({

        type: "GET",

        url: '@Url.Action("AvaliacaoIndex", "Avaliacao")',

        data: { racfUsuario },

        success: function (data) {

        window.location.href = data.redirect;

        }

        })

        }

        function EfetuarUpdate(racfUsuario) {

        $.ajax({

        type: "GET",

        url: '@Url.Action("AtualizarIndex", "AtualizarUsuario")',

        data: { racfUsuario},

        success: function (data) {

        window.location.href = data.redirect;

        }

        })

        }

</script>

Login – Tela

@model GerenciaRH.Model.LoginModel

@{

    ViewBag.Title = "Login";

    Layout = null;

}

<!DOCTYPE html>

<html lang="en">

<head>

    @Styles.Render("~/Content/css")

    @Styles.Render("~/Content/materialcss")

    @Scripts.Render("~/bundles/modernizr")

    <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700|Roboto+Slab:400,700|Material+Icons" />

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/latest/css/font-awesome.min.css">

    <link href="~/Content/toastr.css" rel="stylesheet" />

    <script src="~/Scripts/jquery-3.3.1.min.js"></script>

    <script src="~/Scripts/toastr.min.js"></script>

    <style>

        .hero {

            background-image: url('/Resources/background03.jpg');

            transition: background 1s linear;

        }

        /\*LOADER CSS\*/

        .loader {

            border: 5px solid #d6d6d6;

            /\* Light grey \*/

            border-top: 5px solid #9124a3;

            /\* Blue \*/

            border-radius: 50%;

            width: 60px;

            height: 60px;

            animation: spin 2s linear infinite;

            margin: 40px auto;

            display: none;

        }

        .loader-container {

            text-align: center;

            position: relative;

            width: 100%;

            z-index: 1;

            z-index: 999;

        }

        @@keyframes spin {

            0% {

                transform: rotate(0deg);

            }

            100% {

                transform: rotate(360deg);

            }

        }

    </style>

</head>

<body>

    <div id="hero" class="section section-signup page-header hero">

        <div class="container">

            <div class="row">

                <div class="col-lg-4 col-md-6 ml-auto mr-auto">

                    <div class="card card-login">

                        @using (Html.BeginForm("EfetuarLogin", "Login", FormMethod.Post))

                        {

                            <div class="card-header card-header-primary text-center">

                                <h2 class="card-title">Login</h2>

                            </div>

                            <div class="card-body">

                                <div class="input-group">

                                    <div class="input-group-prepend">

                                        <span class="input-group-text">

                                            <i class="material-icons">face</i>

                                        </span>

                                    </div>

                                    @Html.TextBoxFor(x => x.Usuario.Racf, new { @class = "form-control", @placeholder = "Racf..." })

                                </div>

                                <div class="input-group">

                                    <div class="input-group-prepend">

                                        <span class="input-group-text">

                                            <i class="material-icons">lock\_outline</i>

                                        </span>

                                    </div>

                                    @Html.TextBoxFor(x => x.Usuario.Senha, new { @class = "form-control", @placeholder = "Senha...", @type = "password" })

                                </div>

                            </div>

                            <div class="footer text-center" style="padding-bottom:20px">

                                <button id="btnLogin" class="btn btn-primary" type="submit">

                                    Efetuar Login

                                </button>

                            </div>

                            @\*DIV CARREGAMENTO\*@

                            <div class="loader-container">

                                <div class="loader"></div>

                            </div>

                        }

                    </div>

                </div>

            </div>

        </div>

    </div>

</body>

</html>

<footer class="footer" data-background-color="black">

    <div class="copyright ml-auto mr-auto">

        &copy;

        <script>

            document.write(new Date().getFullYear() + " GerenciaRH - Todos os direitos reservados")

        </script>

    </div>

</footer>

<script>

    $(document).ready(function () {

        if ('@TempData["Erro"]' != "") {

            ToastrOptionFailure();

            toastr.error('@TempData["Erro"]');

        }

        $('.card-login>form').submit(function () {

            $('.loader').show();

        });

        function ToastrOptionFailure() {

            toastr.options = {

                "closeButton": true,

                "debug": false,

                "newestOnTop": false,

                "progressBar": true,

                "positionClass": "toast-top-left",

                "preventDuplicates": false,

                "onclick": null,

                "showDuration": "300",

                "hideDuration": "1000",

                "timeOut": "5000",

                "extendedTimeOut": "1000",

                "showEasing": "swing",

                "hideEasing": "linear",

                "showMethod": "fadeIn",

                "hideMethod": "fadeOut"

            }

        }

        var indexAtual = 0;

        var images = ["/Resources/background02.jpg", "/Resources/background03.jpg","/Resources/background04.jpg","/Resources/background05.jpg"];

        var tamanho = images.length - 1

        function alterarPlanoDeFundo() {

            $('#hero').css('background-image', 'url(' + images[indexAtual] + ')');

            indexAtual = (indexAtual < tamanho) ? indexAtual + 1 : 0;

        }

        alterarPlanoDeFundo();

        window.setInterval(alterarPlanoDeFundo, 8000)

    });

</script>

Relatorio – Tela

@model GerenciaRH.Model.RelatorioModel

@using Chart.Mvc.ComplexChart;

@using Chart.Mvc.Extensions

@{

    ViewBag.Title = "Index";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts{

    <script type="text/javascript" src="~/Scripts/script.js"></script>

    <script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>

}

<script type="text/javascript" src="~/Scripts/script.js"></script>

<script src="~/Scripts/Chart.js"></script>

<div class="col-md-12 col-xl-12">

    <div class="col-md-12 CardFundoDesligamento">

        <div class="row pl-3">

            <h2>Relatórios</h2>

        </div>

        <div class="row pl-3">

            <div class="col-md-12">

                <div class="row">

                    @\*<div class="col-md-10">

                        <a href="#relatorioMensal" class="btn btn-dark text-white" role="button" data-toggle="collapse">Relatório Mensal</a>

                        <a href="#relatorioAnual" class="btn btn-dark text-white" role="button" data-toggle="collapse">Relatório Anual</a>

                    </div>\*@

                    <div class="col-md-2">

                        <img class="ml-5" src="~/Resources/Icons/excel.png" width="30px" height="30px" />

                    </div>

                </div>

                <div class="row pt-3" id="relatorioMensal">

                    <div class="col-md-12">

                        <div class="row">

                            <div class="col-md-4">

                                @{

                                    var notas = new[] { "Gerente", "GerenteRH", "FuncionarioRH","Funcionario" };

                                    var qtdNotas = new List<double> {Model.Gerente,Model.GerenteRH,Model.FuncionarioRH,Model.Funcionario};

                                    var barChart = new BarChart();

                                    barChart.ComplexData.Labels.AddRange(notas);

                                    barChart.ComplexData.Datasets.AddRange(new List<ComplexDataset>{new ComplexDataset

                                      {

                                              Data = qtdNotas,

                                              Label = "Média de gastos empresariais",

                                              FillColor = "rgba(60, 72, 88, 1)",

                                              StrokeColor = "rgba(60, 72, 88, 1)",

                                              PointColor = "rgba(60, 72, 88, 1)",

                                              PointHighlightStroke = "rgba(60, 72, 88, 1)",

                                      }

                                                                                             });

                                }

                                <label for="#GraficoGasto" class="font-weight-bold">Média de gastos empresariais</label>

                                <canvas id="GraficoGasto" class="GraficoNota pl-3 pt-3" style="width:250px; height:192px;"></canvas>

                                @Html.CreateChart("GraficoGasto", barChart)

                                <div class="CardDesligamentoComentario">

                                    <p class="p-2">Gastos mensais da empresa: @Model.Total</p>

                                </div>

                            </div>

                            <div class="col-md-7">

                                <table class="table table-striped border ml-5">

                                    <thead>

                                        <tr>

                                            <th scope="col">RACF</th>

                                            <th scope="col">Nome</th>

                                            <th scope="col">Ultima Nota</th>

                                            <th scope="col">Salario</th>

                                        </tr>

                                    </thead>

                                    @foreach (var usuario in Model.listUsuarios)

                                    {

                                        <tr>

                                            <td>

                                                @usuario.Racf

                                            </td>

                                            <td>

                                                @usuario.NomeUsuario

                                            </td>

                                            <td>

                                                @usuario.Nota

                                            </td>

                                            <td>

                                                @usuario.Salario

                                            </td>

                                        </tr>

                                    }

                                </table>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

</div>

Layouts

@model GerenciaRH.Model.UsuarioModel

<link href="~/Content" rel="stylesheet" />

@if (Model.listUsuario.Count > 0 & Model.listUsuario != null)

{

    <table id="tbUsuarios">

        @foreach (var usuario in Model.listUsuario)

        {

            <tr class="cardColaborador" id="tabela\_@usuario.Racf">

                <td>

                    <img src="~/Resources/Icons/pessoa\_@usuario.IdUsuario .png" width="120" height="120" style="border-radius:30%" />

                </td>

                <td class="separator-gray">

                </td>

                <td class=" separador">

                </td>

                <td class="font-weight-bold" width="440px" style="margin-left:30%">

                    Nome:@usuario.NomeUsuario <br />

                    Racf:@usuario.Racf <br />

                    Cargo:@usuario.NomeCargo <br />

                </td>

                <td class="tooltipCSS" onclick="EfetuarAvaliacao('@usuario.Racf')" style="cursor:pointer;margin-right: 110px">

                    <img src="~/Resources/Icons/avaliacao.png" width="45" height="45" />

                    <span class="tooltiptextCSS">Avaliar</span>

                </td>

                <td class="tooltipCSS" onclick="EfetuarDesligamento('@usuario.Racf', '@Model.Usuario.Racf')" style="cursor:pointer; margin-right: 90px">

                    <img src="~/Resources/Icons/demissao.png" class="m-2" height="30" />

                    <span class="tooltiptextCSS">Desligar</span>

                </td>

                <td class="tooltipCSS" onclick="EfetuarUpdate('@usuario.Racf')" style="cursor:pointer; margin-right: 90px">

                    <img src="~/Resources/Icons/Edit.png" class="m-2" height="30" />

                    <span class="tooltiptextCSS">Editar Usuário</span>

                </td>

            </tr>

        }

    </table>

}

<!DOCTYPE html>

<html>

<head>

    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

    <meta charset="utf-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>@ViewBag.Title - GerenciaRH</title>

    @Styles.Render("~/Content/css")

    @Styles.Render("~/Content/materialcss")

    @Scripts.Render("~/bundles/modernizr")

    <link href="~/Content/toastr.css" rel="stylesheet" />

    <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700|Roboto+Slab:400,700|Material+Icons" />

    @\*<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/latest/css/font-awesome.min.css">\*@

    <script src="~/Scripts/jquery-3.3.1.min.js"></script>

    <script src="~/Scripts/toastr.min.js"></script>

    <script src="~/Scripts/jquery.mask.min.js"></script>

</head>

<body class="index-page sidebar-collapse" style="padding-top:0px;">

    @Html.Partial("\_NavBar");

    <div class="main main-raised">

        <div class="section section-basic">

            @RenderBody()

        </div>

    </div>

    <footer class="footer" data-background-color="black">

        <div class="copyright ml-auto mr-auto">

            &copy;

            <script>

                document.write(new Date().getFullYear() + " GerenciaRH - Todos os direitos reservados")

            </script>

        </div>

    </footer>

    @\*@Scripts.Render("~/bundles/bootstrap")\*@

    @Scripts.Render("~/bundles/inputmask")

    @RenderSection("scripts", required: false)

    <!--   Core JS Files   -->

    <script src="~/assets/js/core/jquery.min.js" type="text/javascript"></script>

    <script src="~/assets/js/core/popper.min.js" type="text/javascript"></script>

    <script src="~/assets/js/core/bootstrap-material-design.min.js" type="text/javascript"></script>

    <script src="~/assets/js/plugins/moment.min.js"></script>

    <!-- Plugin for the Datepicker, full documentation here: https://github.com/Eonasdan/bootstrap-datetimepicker -->

    <script src="~/assets/js/plugins/bootstrap-datetimepicker.js" type="text/javascript"></script>

    <!--  Plugin for the Sliders, full documentation here: http://refreshless.com/nouislider/ -->

    <script src="~/assets/js/plugins/nouislider.min.js" type="text/javascript"></script>

    <!--  Google Maps Plugin    -->

    @\*<script src="https://maps.googleapis.com/maps/api/js?key=YOUR\_KEY\_HERE"></script>\*@

    <!-- Control Center for Material Kit: parallax effects, scripts for the example pages etc -->

    <script src="~/assets/js/material-kit.js?v=2.0.6" type="text/javascript"></script>

</body>

</html>

@{

    Layout = null;

}

@if (Model.usuarioLogado != null)

{

    if (Model.usuarioLogado.IdCargo == 1)

    {

        <div class="collapse navbar-collapse">

            <ul class="navbar-nav ml-auto">

                <li class="dropdown nav-item">

                    <a href="#" class="dropdown-toggle nav-link" data-toggle="dropdown">

                        <i class="material-icons">apps</i> Turma

                    </a>

                    <div class="dropdown-menu dropdown-with-icons">

                        <a href="./index.html" class="dropdown-item">

                            <i class="material-icons">layers</i> All Components

                        </a>

                        <a href="https://demos.creative-tim.com/material-kit/docs/2.1/getting-started/introduction.html" class="dropdown-item">

                            <i class="material-icons">content\_paste</i> Documentation

                        </a>

                    </div>

                </li>

            </ul>

        </div>

    }

}

<nav class="navbar navbar-inverse navbar-expand-lg" style="z-index:1000;">

    <div class="navbar-translate">

        <a class="navbar-brand" style="padding-left:2rem;" href="#0">GerenciaRH</a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" aria-expanded="false" aria-label="Toggle navigation">

            <span class="sr-only">Toggle navigation</span>

            <span class="navbar-toggler-icon"></span>

            <span class="navbar-toggler-icon"></span>

            <span class="navbar-toggler-icon"></span>

            <link href="~/Content/Site.css" rel="stylesheet" />

        </button>

    </div>

    <!-- Collect the nav links, forms, and other content for toggling -->

    <div class="collapse navbar-collapse">

        <ul class="navbar-nav">

            <li class="nav-item botaoMenu">

                <a href='@Url.Action("Index", "Home")' class="nav-link"><img src="~/Resources/Icons/home.png" width="30" height="30"/>

                    Home

                </a>

            </li>

            <li class="nav-item botaoMenu">

                <a href='@Url.Action("Index","Cadastro")' class="nav-link"><img src="~/Resources/Icons/pessoa.png" width="25" height="25" />

                    Cadastro

                </a>

            </li>

            <li class="nav-item botaoMenu">

                <a href='@Url.Action("Index","Relatorio")' class="nav-link"><img src="~/Resources/Icons/relatorio.png" width="25" height="25" />

                    Relatórios

                </a>

            </li>

        </ul>

        <ul class="navbar-nav ml-auto pr-5 pb-3">

            <li class="dropdown nav-item">

                <a href="#pablo" class="profile-photo dropdown-toggle nav-link" data-toggle="dropdown">

                    <div class="profile-photo-small">

                        <img src="~/Resources/Icons/pessoa\_1 .png" alt="Circle Image" class="rounded-circle img-fluid pb-1">

                    </div>

                </a>

                <div class="dropdown-menu dropdown-menu-right">

                    <h6 class="dropdown-header">Menu de usuário</h6>

                    <a href="@Url.Action("EfetuarLogin","Login")" class="dropdown-item"><img src="~/Resources/Icons/logoff.png" width="20" height="20">Logoff</a>

                </div>

            </li>

        </ul>

    </div>

    <!-- /.navbar-collapse -->

    <!-- /.container-->

</nav>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

    <meta name="viewport" content="width=device-width" />

    <title>Erro</title>

</head>

<body>

    <hgroup>

        <h1>Erro.</h1>

        <h2>Ocorreu um erro ao processar sua solicitação.</h2>

    </hgroup>

</body>

</html>

# APÊNDICE – BASE DE DADOS

CREATE TABLE [dbo].[Avaliacao](

[IdAvaliacao] [int] IDENTITY(1,1) NOT NULL,

[UsuarioAvaliado] [int] NOT NULL,

[UsuarioAvaliador] [int] NOT NULL,

[Nota] [varchar](1) NOT NULL,

[DtAvaliacao] [date] NOT NULL,

[UltimaAvaliacao] [bit] NULL,

PRIMARY KEY CLUSTERED

(

[IdAvaliacao] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Cargo] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Cargo](

[IdCargo] [int] IDENTITY(1,1) NOT NULL,

[NomeCargo] [varchar](50) NOT NULL,

[Salario] [float] NULL,

PRIMARY KEY CLUSTERED

(

[IdCargo] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Chat] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Chat](

[IdChat] [int] IDENTITY(1,1) NOT NULL,

[IdTipoChat] [int] NOT NULL,

[IdMotivoChat] [int] NOT NULL,

[UsuarioRemetente] [int] NOT NULL,

[UsuarioDestinatario] [int] NOT NULL,

[DtEnvio] [date] NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdChat] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Impedimento] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Impedimento](

[IdImpedimento] [int] IDENTITY(1,1) NOT NULL,

[TipoImpedimento] [varchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdImpedimento] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[ImpedimentoUsuario] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[ImpedimentoUsuario](

[IdImpedimentoUsuario] [int] IDENTITY(1,1) NOT NULL,

[IdUsuario] [int] NOT NULL,

[IdImpedimento] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdImpedimentoUsuario] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Log] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Log](

[IdLog] [int] IDENTITY(1,1) NOT NULL,

[DtAlteracao] [date] NOT NULL,

[IdUsuario] [int] NULL,

[ValorAgora] [varchar](100) NULL,

[ValorDepois] [varchar](100) NULL,

CONSTRAINT [PK\_\_Log\_\_0C54DBC60131AB23] PRIMARY KEY CLUSTERED

(

[IdLog] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[MotivoChat] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[MotivoChat](

[IdMotivoChat] [int] IDENTITY(1,1) NOT NULL,

[NomeMotivoChat] [varchar](20) NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdMotivoChat] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TipoChat] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TipoChat](

[IdTipoChat] [int] IDENTITY(1,1) NOT NULL,

[NomeTipoChat] [varchar](20) NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdTipoChat] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TipoLog] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TipoLog](

[IdTipoLog] [int] IDENTITY(1,1) NOT NULL,

[NomeTipoLog] [varchar](20) NOT NULL,

PRIMARY KEY CLUSTERED

(

[IdTipoLog] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Usuario] Script Date: 10/11/2019 02:07:18 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Usuario](

[IdUsuario] [int] IDENTITY(1,1) NOT NULL,

[NomeUsuario] [varchar](50) NOT NULL,

[Racf] [varchar](4) NOT NULL,

[CPFCNPJ] [varchar](20) NOT NULL,

[CEP] [varchar](9) NULL,

[NumeroEndereco] [int] NOT NULL,

[DtAdmissao] [date] NOT NULL,

[DtDemissao] [date] NULL,

[IdCargo] [int] NOT NULL,

[IdUsuarioAlteracao] [int] NULL,

[Senha] [varchar](100) NULL,

[Ativo] [bit] NOT NULL,

[IdGestor] [int] NULL,

[TipoDocumento] [varchar](4) NULL,

[PIS] [varchar](20) NULL,

[Motivo] [varchar](50) NULL,

[IdDesligamento] [int] NULL,

[Email] [varchar](100) NULL,

[Endereco] [varchar](50) NULL,

CONSTRAINT [PK\_\_Usuario\_\_5B65BF97F6FDF45C] PRIMARY KEY CLUSTERED

(

[IdUsuario] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Avaliacao] WITH CHECK ADD CONSTRAINT [FK\_\_Avaliacao\_\_Usuar\_\_239E4DCF] FOREIGN KEY([UsuarioAvaliado])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[Avaliacao] CHECK CONSTRAINT [FK\_\_Avaliacao\_\_Usuar\_\_239E4DCF]

GO

ALTER TABLE [dbo].[Avaliacao] WITH CHECK ADD CONSTRAINT [FK\_\_Avaliacao\_\_Usuar\_\_24927208] FOREIGN KEY([UsuarioAvaliador])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[Avaliacao] CHECK CONSTRAINT [FK\_\_Avaliacao\_\_Usuar\_\_24927208]

GO

ALTER TABLE [dbo].[Chat] WITH CHECK ADD FOREIGN KEY([IdMotivoChat])

REFERENCES [dbo].[MotivoChat] ([IdMotivoChat])

GO

ALTER TABLE [dbo].[Chat] WITH CHECK ADD FOREIGN KEY([IdTipoChat])

REFERENCES [dbo].[TipoChat] ([IdTipoChat])

GO

ALTER TABLE [dbo].[Chat] WITH CHECK ADD CONSTRAINT [FK\_\_Chat\_\_UsuarioDes\_\_2A4B4B5E] FOREIGN KEY([UsuarioDestinatario])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[Chat] CHECK CONSTRAINT [FK\_\_Chat\_\_UsuarioDes\_\_2A4B4B5E]

GO

ALTER TABLE [dbo].[Chat] WITH CHECK ADD CONSTRAINT [FK\_\_Chat\_\_UsuarioRem\_\_29572725] FOREIGN KEY([UsuarioRemetente])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[Chat] CHECK CONSTRAINT [FK\_\_Chat\_\_UsuarioRem\_\_29572725]

GO

ALTER TABLE [dbo].[ImpedimentoUsuario] WITH CHECK ADD FOREIGN KEY([IdImpedimento])

REFERENCES [dbo].[Impedimento] ([IdImpedimento])

GO

ALTER TABLE [dbo].[ImpedimentoUsuario] WITH CHECK ADD CONSTRAINT [FK\_\_Impedimen\_\_IdUsu\_\_1FCDBCEB] FOREIGN KEY([IdUsuario])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[ImpedimentoUsuario] CHECK CONSTRAINT [FK\_\_Impedimen\_\_IdUsu\_\_1FCDBCEB]

GO

ALTER TABLE [dbo].[Log] WITH CHECK ADD CONSTRAINT [FK\_\_Log\_\_IdUsuario\_\_2D27B809] FOREIGN KEY([IdUsuario])

REFERENCES [dbo].[Usuario] ([IdUsuario])

GO

ALTER TABLE [dbo].[Log] CHECK CONSTRAINT [FK\_\_Log\_\_IdUsuario\_\_2D27B809]

GO

ALTER TABLE [dbo].[Usuario] WITH NOCHECK ADD CONSTRAINT [FK\_\_Usuario\_\_IdCargo\_\_151B244E] FOREIGN KEY([IdCargo])

REFERENCES [dbo].[Cargo] ([IdCargo])

GO

ALTER TABLE [dbo].[Usuario] CHECK CONSTRAINT [FK\_\_Usuario\_\_IdCargo\_\_151B244E]

GO