Projeto Finis
Especificação de Arquitetura
Versão <1.1.3-RELEASE>

Índice Analítico

1. Introdução

1.1 Finalidade

Este documento apresenta uma visão geral abrangente da arquitetura do sistema e utiliza uma série de visões arquiteturais diferentes para ilustrar os diversos aspectos do sistema. Sua intenção é capturar e transmitir as decisões significativas do ponto de vista da arquitetura que foram tomadas em relação ao sistema.

2. Visões da arquitetura

2.1 Visão de Caso de Uso

Os seguintes casos de usos constituem a primeira parte integrante do sistema:

- Gerenciar Clientes
  Gerenciamento do cadastro de clientes, possibilitando ao usuário a inserção, alteração e exclusão de um registro de clientes.

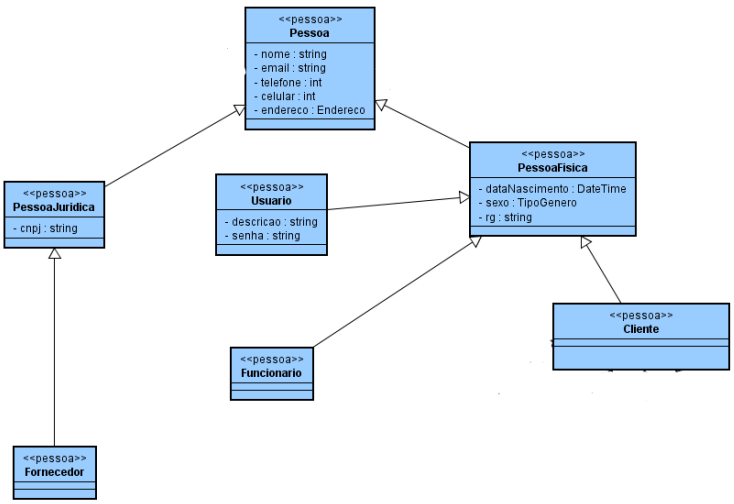
- Gerenciar Exemplares
  O Gerenciamento do cadastro de Exemplares possibilita ao usuário a inserção, alteração e exclusão de um exemplar no sistema.

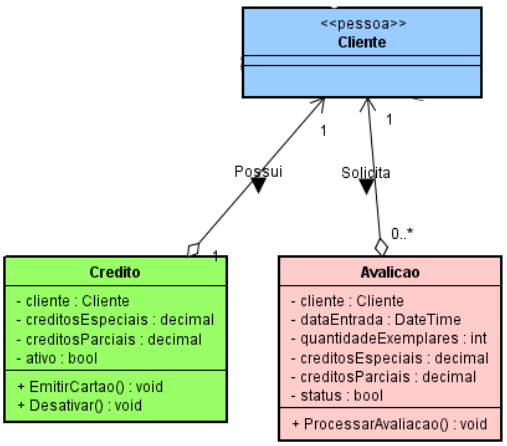
- Gerenciar Avaliações
  Gerenciamento do cadastro de avaliações, as avaliações de exemplares solicitadas pelos clientes são cadastradas possibilitando a inserção, alteração e exclusão destes registros.

- Gerenciar Créditos
  Os créditos que os clientes recebem pelas avaliações realizadas são mantidos no sistema, possibilitando a inserção, alteração e exclusão destes créditos.

2.1 Visão Lógica

O sistema é subdividido em classes que possibilitam a organização, integração e persistência dos dados. As classes de Cliente, juntamente com Funcionário e Usuário são generalizações da classe Pessoa Física que por sua vez é uma generalização da Pessoa. O mesmo ocorre com as classes Funcionário, Pessoa Jurídica respectivamente.





A classe Cliente interage com as classes Crédito e Avaliação, onde os créditos pertencentes a um cliente são mantidos assim como também as avaliações dos clientes são mantidas.

A classe Exemplar mantém os cadastros de todos os registros de exemplares disponíveis para o usuário.

2.4 Visão de Implementação

O modelo de implementação utilizada para a estrutura geral deste projeto se baseia no padrão de projeto arquitetural denominado MVC (*Model-View-Controller*), é um padrão de projeto baseado em outros padrões como o *Observer*, *Composite* e *Strategy*. O MVC, como o nome sugere, possibilita a separação de um projeto em múltiplas camadas, das quais fazem parte: Modelo (*Model*), Visão (*View*) e Controlador (*Controller*).

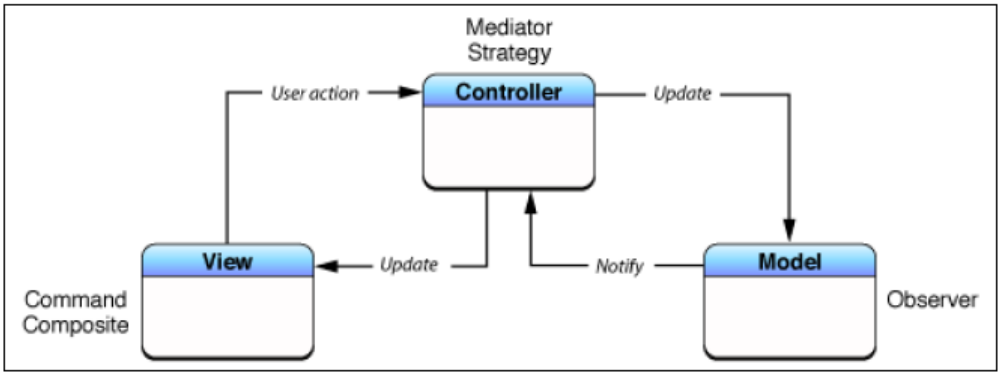
Desenvolvido na década de 70 pelo cientista da computação norueguês Trygve Mikkjel Heyerdahl Reenskaug o MVC surgiu como uma forma melhorada para construção de interfaces gráficas, mas tomou grande amplitude e passou a ser utilizado, de uma forma geral, na arquitetura de sistemas complexos.

O MVC que separa a lógica de negócios no *model*, a apresentação na *view* e a interação entre eles no *controller*, também se apresenta como uma boa escolha para a construção de aplicações web interativas. Isto devido ao fato de neste tipo de aplicação haverem grandes quantidades de interações de diversos tipos de usuários e buscas e exibições de dados.

A justificativa da escolha deste padrão se dá pelo fato de ser um padrão arquitetônico que se adéqua de forma concisa a soluções na web. Isso, graças à separação de camadas conforme a sua funcionalidade.

A essência do padrão MVC, é o relacionamento entre os componentes *Model*, *View*, *Controller*, Pois cada um possui uma funcionalidade especifica. Como a separação de componentes é a característica fundamental do padrão, essa divisão desacopla o acesso a dados da lógica do negócio e também da maneira na qual a informação é exibida para os clientes finais. Essa separação que torna os sistemas desenvolvidos com este padrão mais flexíveis, manuteníveis e ainda permitindo o reuso dos componentes.

Figura 1 - Diagrama de funcionamento do padrão MVC



Fonte: Fábrica de Software - SENAC/MS[1]

Desta maneira, o controlador ganha a responsabilidade de receber a notificação do modelo e transcrever que deve ser feita uma atualização na visão. Esta alteração torna a view menos acoplada ao model e é satisfatória para o cenário de desenvolvimento de aplicações para web.

2.4.1 Visão de Dados

As tecnologias que serão utilizadas neste sistema estão listadas a seguir:

2.4.1.1. Armazenamento de Dados:

Local: Microsoft SQL Server

Sistema gerenciador de banco de dados relacional desenvolvido pela Microsoft, utiliza para as consultas primárias as linguagens T-SQL e ANSI SQL. A versão escolhida para ser utilizada é a versão SQL Server 2016 SP1 Express Edition, uma versão gratuita de nível básico que possibilita a utilização dos principais recursos do SQL Server, permitindo criar aplicativos controlados por dados de até 10 GB de tamanho máximo de disco, podendo ser escalados entre edições à medida que for necessário um maior armazenamento de dados.

Remoto (nuvem): Banco de Dados SQL do Azure

O Banco de Dados SQL do Azure é um serviço de banco de dados relacional de nuvem da Microsoft com base no mecanismo do banco de dados relacional Microsoft SQL Server. O Banco de Dados SQL oferece um desempenho previsível em vários níveis de serviço, escalabilidade dinâmica, continuidade de negócios interna e proteção de dados. No nível básico no modelo de banco de dados único com máquina virtual dedicada o serviço oferece a partir de 2GB de armazenamento pelo valor de R$ 0,0252/hora. No nível básico em um ambiente compartilhado o serviço permite hospedar aplicativos WEB, móveis ou de API gratuitamente com 1GB de espaço em disco.

2.4.1.2. Back-end:

Framework: Microsoft ASP.NET MVC Framework

O ASP.NET MVC é um framework para aplicações WEB desenvolvido pela Microsoft que implementa o padrão de projeto MVC. É um framework open souce que fornece um ambiente robusto, leve e integrado para desenvolvimento junto ao IDE Microsoft Visual Studio.

Linguagem de programação: C#

[1] Disponível em: http://fabrica.ms.senac.br/2013/06/as-camadas-mvc/; Acesso em abril de 2017.

C# (ou C Sharp) é uma linguagem de programação interpretada, multi-paradigma, fortemente tipada, e, possuindo paradigmas de programação imperativa, funcional, declarativa, orientada a objetos e genérica, C# foi desenvolvida pela Microsoft como parte da plataforma .NET. A sua sintaxe orientada a objetos foi baseada no C++ mas inclui muitas influências de outras linguagens de programação, como Object Pascal e, principalmente, Java. O código fonte é compilado para Common Intermediate Language (CIL) que é interpretado pela máquina virtual Common Language Runtime (CLR). C# é uma das linguagens projetadas para funcionar na Common Language Infrastructure da plataforma .NET Framework.

2.4.1.3. Persistência de Dados:

Framework: ADO.NET Entity Framework

É uma das principais ferramentas de persistência presentes na plataforma .NET, sendo parte integrante do pacote de tecnologias ADO.NET. Proporciona soluções para minimizar o problema de impedância, abstraindo do desenvolvedor vários detalhes dos bancos de dados relacionais. Além disso, fornece uma série de recursos que aumentam muito a produtividade no desenvolvimento de aplicações persistentes.

2.4.1.5. Relatórios:

SAP Crystal Reports Developer Version for Microsoft Visual Studio

Extensão gratuita ao IDE Visual Studio que permite a criação de relatórios no modelo SAP Crystal Reports Design Tool com as ferramentas e componentes básicos disponíveis.

2.4.1.6. Front-end:

Framework: Twitter Bootstrap WEB Framework

Framework gratuito e open souce exclusivo para desenvolvimento WEB front-end, possui templates e componentes baseados em HTML5 e CSS3. Seus componentes são otimizados para o desenvolvimento de layouts responsivos de maneira prática resultando um aumento de produtividade.

Linguagem de programação: JavaScript

Linguagem de programação interpretada caracterizada por uma tipagem fraca e dinâmica. Em sua utilização em web front-end permite a construção de scripts e funções para serem interpretados em navegadores de maneira fácil e prática.

2.4.1.6. Testes Unitários/Integração:

Framework: Microsoft Visual Studio Unit Testing Framework

Suíte de ferramentas para testes unitários integrada ao IDE Visual Studio que possibilita testes unitários de aplicações, classes e métodos.

4. Referências

https://www.sap.com/brazil/product/analytics/crystal-visual-studio.html

https://msdn.microsoft.com/pt-br/library/bb399572(v=vs.100).aspx

https://msdn.microsoft.com/pt-BR/library/kx37x362.aspx

https://docs.microsoft.com/pt-br/azure/sql-database/sql-database-technical-overview

https://msdn.microsoft.com/pt-br/library/dd381412(v=vs.108).aspx

https://www.microsoft.com/pt-br/sql-server/sql-server-editions-express