RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

PARKING: Sistema web para gerenciamento de estacionamentos automotivos.

Antônio Luiz Scarpe
Graduando em Sistemas de Informação – Uni-FACEF
aluizscarpe@gmail.com

Igor Pessoni Silva
Graduando em Sistemas de Informação – Uni-FACEF
igorpessoni2014@outlook.com

Carlos Eduardo de França Roland
Mestre em Desenvolvimento Regional – Uni-FACEF
roland@facef.br

Resumo

O objetivo desse trabalho é apresentar o processo de análise, projeto e implementação de um sistema *web* para Gestão de Estacionamentos Automotivos, visando melhoria na eficiência e eficácia no dia a dia de trabalho nesses estabelecimentos. Para isso foram feitos levantamentos dos processos de trabalho, buscando documentar as rotinas para projetá-las e implementá-las em uma aplicação *web* utilizando *frameworks* Java e o Sistema Gerenciador de Bancos de Dados PostgreSQL. A implementação de um produto mínimo viável (MVP do termo inglês Minimum Viable Product) se mostrou factível com as decisões de projeto tanto da metodologia de desenvolvimento quanto das ferramentas utilizadas. Com o resultado alcançado, apresenta-se uma alternativa para melhoria da qualidade e agilidade operacional de estacionamentos automotivos, oferecendo uma alternativa para a gestão dessa classe de negócio em constante crescimento. A partir da reflexão proporcionada pelo processo de desenvolvimento, pode-se vislumbrar novas e inovadoras funcionalidades que podem potencializar o valor entregue aos clientes do produto.

Palavras-chave: Estacionamento Automotivo. Frameworks. Java. PostgreSQL. Processos Gerenciais. Sistema Web.

Abstract

The objective of this paper is to present the process of analysis, design and implementation of a web system for Automotive Parking Management, aiming at improving efficiency and effectiveness in the daily work of these establishments. For this, surveys of work processes were made, seeking to document the routines to design them and implement them in a web application using Java frameworks and the PostgreSQL Database Manager System. The implementation of a Minimum Viable Product (MVP) proved feasible with the design decisions of both the development methodology and the tools used. With the result achieved, we present an alternative for improving the quality and operational agility of automotive parking, offering an alternative for the management of this constantly growing business class. From the reflection provided by the development process, one can glimpse new and innovative features that can enhance the value delivered to customers of the product.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Keywords: *Automotive Parking. Frameworks. Java. Management processes. PostgreSQL. Web system.*

1 Introdução

Uma das principais respostas ao aumento global do número de motoristas urbanos, foi o surgimento dos primeiros estacionamentos durante as décadas de 1920 e 1930. As áreas destinadas ao repouso de veículos automotores, de propulsão humana ou animal, facilitaram a rotina dos condutores, tornando-se essências na atual sociedade.

Com o passar das décadas, a constante renovação e a ampliação da frota de veículos tornaram-se um problema tanto para os gestores de estacionamentos, que encontravam cada vez mais dificuldade para controlar a entrada e saída dos veículos de seus estabelecimentos, quanto para os próprios clientes que tinham problemas para encontrar as melhores vagas no ato do estacionamento.

Nesse contexto, o presente artigo possui como objetivo geral o desenvolvimento de uma aplicação *web* integrada a um sistema microcontrolado para gestão e controle de entrada e saída de veículos em um estacionamento. Com o atingimento do objetivo geral, projeta-se o aumento significativo na eficácia dos procedimentos do gestor do estacionamento e uma melhora na identificação das vagas disponíveis no estabelecimento.

Este trabalho buscou, durante seu desenrolar, cumprir os seguintes objetivos específicos: criação e apresentação da documentação do artefato de *software*; produzir o código fonte da aplicação *web*; demonstrar a construção do protótipo com dispositivos microcontrolados; simular o funcionamento do *software* em situações reais; e por fim apresentação dos resultados alcançados com a finalização do protótipo funcional.

Os procedimentos metodológicos adotados para o desenvolvimento foram: revisão bibliográfica de tópicos referentes ao tema, com estudos realizados em livros e artigos científicos; levantamento dos requisitos e funcionalidades com potenciais clientes; construção dos diagramas de BPMN, de Classe, de Sequência e de Casos de Uso; desenvolvimento da aplicação de controle de entrada e saída de veículos utilizando a linguagem Java, com os *frameworks* JSF, *Hibernate* e *Spring* em conjunto com o gerenciador de banco de dados PostgreSQL.

O presente trabalho é apresentado em sessões abordando os seguintes tópicos: referencial teórico contendo a conceituação e caracterização dos componentes que foram utilizados no desenvolvimento do protótipo funcional; uma seção destinada à análise de viabilidade da proposta com foco empreendedor; outra contendo a solução proposta; então a apresentação dos resultados obtidos; e por fim as considerações finais sobre a realização do projeto.

2 Referencial Teórico

Essa seção apresenta o referencial teórico que embasa os elementos do tema, da questão problema e da hipótese de solução da pesquisa, e as ferramentas e componentes utilizados durante o desenvolvimento do projeto. Destacam-se a

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

explicação do conceito de estacionamento privado bem como notas sobre o uso da linguagem de programação Java, os frameworks JSF, *Hibernate* e *Spring,* e do sistema gerenciador de banco de dados PostgreSQL. Finalizando, é mencionado o uso da arquitetura MVC para o desenvolvimento da solução proposta.

2.1 Estacionamentos Privados

Pode-se definir o termo estacionamento como a designação das áreas destinadas para repouso de veículos automotores de propulsão humana ou propulsão animal. Eles surgiram a partir da década de 1920, quando os automóveis passaram a ocupar cada vez mais espaços nos perímetros urbanos e foi necessário adotar uma estratégia para otimizar essa ocupação (MOBILIDADE URBANA E ESTACIONAMENTO, 2017).

Com o crescente número de vagas nos condomínios, escritórios e shoppings, utilizar o automóvel para se deslocar para outros pontos da cidade tornou-se mais fácil e conveniente para os motoristas e não demorou muito para empreendedores identificarem a oportunidade e passar a tarifarem suas vagas, fazendo, assim, com que se surgissem os primeiros estacionamentos privados.

Portanto, um estacionamento privado pode ser definido como um negócio local em que o proprietário de um espaço cobra de um proprietário de um veículo para que este possa manter seu patrimônio seguro e protegido por um determinado período.

O objetivo desse trabalho é criar uma solução para os problemas corriqueiros dos estacionamentos privados, como controle de clientes conveniados, controle financeiro, controle de entrada e saída de veículos, e monitoramento, em tempo real, das vagas disponíveis no estabelecimento.

2.2 Linguagem Java.

Para a implementação do *front-end* e das regras de negócio da solução proposta por este projeto, optou-se pela utilização da linguagem de programação Java. Java trata-se de uma tecnologia de programação orientada a objetos, que possui uma diferença em relação às linguagens modernas, pois é compilada para um *bytecode* que é um código intermediário que, por sua vez, é interpretado por uma máquina virtual chamada Java Virtual Machine.

A linguagem foi desenvolvida na primeira metade da década de 1990 em um dos laboratórios da empresa Sun Microsystems, mantendo um objetivo inicial de ser mais simples e eficiente que linguagens predecessoras utilizadas na época. Seu primeiro alvo de aplicação era alcançar as produtoras de *software* para produtos eletrônicos de consumo, principalmente eletrodomésticos. Porém, o verdadeiro sucesso comercial viria apenas após a popularização da rede *internet*, que forçou os colaboradores da Sun Microsystems a adaptar o código para ser utilizado em microcomputadores (INDRUSIAK, 1996).

A Java Virtual Machine, também conhecida como JVM, funciona como um interpretador de código, carregando e executando os aplicativos Java, convertendo *bytecodes* em códigos executáveis pelos processadores das máquinas.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Graças à sua forma de atuação, é possível obter maior portabilidade de código, pois a execução do Java se relaciona diretamente com a JVM e não com o sistema operacional que gerencia os recursos do equipamento.

Entre as linguagens que suportam os princípios da Programação Orientada a Objetos (POO), Java destaca-se entre as mais utilizadas. Os principais aspectos que a diferenciam das demais são a sua portabilidade, desempenho e performance e a capacidade de reutilização dos *softwares* desenvolvidos (ALMEIDA, 2005).

Em geral, a linguagem Java torna-se significativamente apropriada para o desenvolvimento deste projeto, pois trata-se de uma tecnologia de programação adequada para o desenvolvimento de aplicações complexas em diversos nichos como aplicações para internet, ou mesmo sistemas em arquitetura *stand-alone* (INDRUSIAK, 1996).

2.3 PostgreSQL

Visando manter uma infraestrutura segura e estável para armazenar os dados que serão processadas pela a aplicação da solução proposta, optou-se pela utilização do Sistema Gerenciador de Banco de Dados (SGBD) *open source* PostgreSQL.

O PostgreSQL é um SGBD Relacional distribuído de forma gratuita (licença *open source*), que pode ser utilizado para armazenar dados de aplicações como ERPs ou qualquer outra solução de *software* para diversas áreas de negócios existentes, bem como administrar o acesso às informações resultantes do processamento destes dados (MILANI, 2008).

Sua história, começa no ano de 1986 durante a execução do projeto POSTGRES da Universidade Berkeley, na Califórnia (EUA). Coordenada pelo professor Michael Stonebreaker, uma equipe foi designada para criar o modelo e as regras de um novo sistema de armazenamento de dados. Os integrantes deste grupo foram apoiados por diversos órgãos para execução do projeto, dentre eles são mencionados o Army Research Office (ARO) e National Science Foundation (NSF). Houve diversas publicações de versões com correções de bugs entre 1987 e 1993, porém foi em 1994 que ocorreu a maior mudança com a incorporação da linguagem SQL. Esta funcionalidade foi implementada por dois desenvolvedores chamados Andrew Yu e Jolly Chen, e na mesma época o programa foi compatibilizado para o padrão ANSI C, o que na prática tornou-o portável para mais de uma plataforma. Dessa forma o então Postgres95 ficou popular entre os SGBDs do mercado (MILANI, 2008).

Segundo o autor, no ano de 1996 novas melhorias surgiram, e o nome Postgre95 já estava desatualizado. Novamente a denominação da ferramenta foi mudada, desta vez para o nome utilizado hoje: PostgreSQL.

Este SGBD possuí excelente reputação de confiabilidade, integridade de dados e conformidade a padrões, sendo simples de manipular. Também possui mecanismos que o tornam seguro, oferecendo suporte a diversos tipos de dados sofisticados como JSON, XML, objetos geométricos, hierarquias, *tags* e matrizes. Também possui várias funcionalidades complexas, como por exemplo: Controle de

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

concorrência multiversionado; recuperação em um ponto no tempo; replicação assíncrona; transações agrupadas; cópias de segurança quente; planejador de consultas; suporte conjunto de caracteres internacionais; sensibilidade a caixa além de ser altamente escalável (CARVALHO, 2017).

Todas as características que foram mostradas nessa sessão, bem como o fato de se tratar de um SGBD gratuito e de fácil manipulação, qualificam o PostgreSQL para suporte a este projeto.

2.4 Framework Hibernate

O *software* que foi desenvolvido para implementação da solução da questão problema deste projeto consiste em uma aplicação Java. Por conta dessa característica, foi utilizado o *framework* Hibernate para a implementação da persistência dos dados com o SGBD.

O Hibernate é um *framework* que possui o propósito de facilitar os ciclos de administração, produção e suporte do banco de dados base da infraestrutura de um sistema computadorizado. Pode-se determinar que se trata de um instrumento que visa, a longo prazo, a diminuição de custo de manutenção do ciclo de vida de um sistema (AVILA, 2013).

Este *framework* possui várias vantagens como, por exemplo, o fato de ser *open source* (licença LGPL); a funcionalidade de transformar as classes feitas no Java em tabelas de dados, bem como os tipos de dados; melhoria da portabilidade; reutilização e flexibilidade de código; utilização de recursos nativos; dentre outras (RODRIGUES, VIEIRA, et al., 2010).

2.5 Framework JavaServer Faces

Para a construção das interfaces do sistema, visando facilitar a interação do usuário com o *software*, foi utilizado o *framework* JavaServer Faces. Essa tecnologia estabelece padrões para o desenvolvimento de interfaces com o usuário no lado do servidor (*back-end*) (JAVASERVER FACES TECHNOLOGY, 2019).

Trata-se de uma estrutura baseada padrão Model-View-Controller para aplicações *web* (WEB MVC), que se concentra em simplificar a criação de interfaces para o usuário em sistemas baseados em Java WEB, além de facilitar a reutilização de componentes de *user interface* (TUTORIAL JSF 2,0, 2010).

2.6 Spring Framework

O Spring é um *framework* Java produzido com o propósito de auxiliar o desenvolvimento de aplicações, explorando, para isso, os conceitos de Inversão de Controle e Injeção de Dependências. Dessa forma, os projetos que desejam adotá-lo dispõem de uma tecnologia que fornece recursos necessários à grande parte das aplicações e também, módulos para persistência de dados, integração, segurança, testes e desenvolvimento *web* (DEVMEDIA, 2019).

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

2.7 Estrutura Model-View-Controller (MVC)

Para se obter uma melhor organização e disposição do código a ser escrito para aplicações *web*, utilizada-se a estrutura MVC.

A estrutura MVC, cuja sigla corresponde aos termos *Model*, *View* e *Controller*, trata-se de um padrão de *design* de *software* que pode ser definido como um padrão cuja estrutura desejada existe para facilitar o gerenciamento da representação dos dados coletados e armazenados nos sistemas e *softwares* desenvolvidos (REENSKAUG e COPLIEN, 2009).

As três camadas do MVC tem as características:

> [...] de baixo acoplamento e alta coesão, sendo eles: Modelo (Model) – mantém o estado da aplicação, é mais que uma classe para armazenar dados, nele devem estar todas as regras de negócios e também as comunicações com o banco de dados se necessário; Visão (View) – especifica exatamente como o modelo deve ser apresentado. É a interface do usuário. A visão é dinâmica se adequando a qualquer modificação do modelo; e Controlador (Controller) – traduz as interações do usuário com a visão, mapeando-as para tarefas que o modelo irá realizar (LUCIANO e ALVES, 2011, p. 106).

3 Empreendendo a Solução Proposta

Esta seção apresenta e define os pontos referentes à visão empreendedora da solução proposta por este projeto. No decorrer da seção, serão descritos os principais conceitos da área e evidenciados os princípios das Startups Enxutas e do Modelo de Negócios CANVAS. Por fim, é apresentado o CANVAS desenvolvido para esse projeto.

3.1 Conceitos de Empreendedorismo

Elaborado em 1945 pelo economista e cientista político Joseph Alois Schumpter, o conceito de empreendedorismo pode ser definido como algo a ser projetado por pessoas versáteis com um conjunto de características como por exemplo, habilidades técnicas para gerar e organizar recursos financeiros, e gerenciar operações internas de uma empresa (ZUGMAN e CASTOR, 2009).

Uma definição diferente para o conceito de empreendedorismo pode ser considerada como um método de formar algo distinto, com valor, através da dedicação de tempo e energia, assumindo todos os riscos decorrentes dessas ações, como os financeiros, psicológicos e sociais, e posteriormente receber satisfação econômica e social (HISRICH, PETERS E SHEPHERD, 2014).

Assim é possível afirmar que, apesar de pontuais diferenças nos discursos dos pensadores que idealizam as características das principais concepções do empreendedorismo, empreender trata-se de uma disposição de idealizar, coordenar e realizar projetos, serviços e negócios.

3.2 Startup Enxuta

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Conforme observa Ries (2012), *startup* trata-se de um princípio de empreendedorismo que prioriza o recebimento de *feedback* imediato, contínuo e constante, seja ele quantitativo ou qualitativo sobre uma proposta de negócio. Constitui-se de um ciclo de aprendizado resumido em construir, medir e aprender.

Outra definição dada ao conceito de *startup* é destacada por Nardes e Miranda (2014), conforme mostra o trecho a seguir:

> O conceito evoluiu e, atualmente, especialistas, investidores e empreendedores adotam a ideia de que startup é basicamente um empreendimento que enfrenta um ambiente de extrema incerteza, isto é, trata-se de um grupo de pessoas buscando empreender em mercados onde as variáveis são pouco conhecidas (NARDES e MIRANDA, 2014, p. 254).

O principal objetivo desta metodologia de gestão de negócios é minimizar todo o ciclo de desenvolvimento do protótipo de produto da empresa envolvida, incentivando conhecer diversas ideias de forma mais rápida até que se alcance a melhor solução possível, sendo constantemente exposto às incertezas do mercado.

Então pode-se considerar que *startup* é uma instituição humana que foi criada para oferecer novidades nos campos de produtos e serviços em condições incertas (RIES, 2012).

3.3 Modelo de Negócios Canvas

Apesar de ser um termo que existe há no mínimo 50 anos, o Modelo de Negócios sofreu, por muito tempo, dificuldades em suas implementações, devido ao fato de não existir um glossário comum e nem uma ferramenta que pudessem ser utilizados pelos profissionais que atuavam como empreendedores para descrever seus negócios.

Esse cenário começou a mudar a partir do ano de 2010, quando foi publicado o livro Business Model Generation, que promoveu o conjunto de elementos organizados e estruturados visualmente que viria a se tornar a principal metodologia para descrever uma *startup*.

Osterwalder e Pigneur (2011) definem:

> Acreditamos que um Modelo de Negócios pode ser melhor descrito com nove componentes básicos, que mostram a lógica de como uma organização pretende gerar valor. Os nove componentes cobrem as quatro áreas principais de um negócio: clientes, oferta, infraestrutura e viabilidade financeira (OSTERWALDER; PIGNEUR, 2011, p.15).

Os nove componentes são: Oferta de Valor, Segmentos de Cliente, Canais, Relacionamentos com Cliente, Fontes de Receita, Recursos Chave, Atividades Chave, Parcerias Chave, e Estrutura de Custos (OSTERWALDER; PIGNEUR, 2011).

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

O CANVAS é uma ferramenta de fácil manuseio e identificação, que promove a discussão dos principais aspectos do modelo de negócio, visando a evolução do conceito do negócio e por conta dessas características, será aplicado no projeto.

3.4 Estudo de Caso – Parking Gestão de Estacionamentos

Nesta sessão será apresentado o Modelo de Negócios CANVAS desenvolvido para esse projeto, bem como a interpretação de cada componente mostrado.

3.4.1. Quadro CANVAS

Buscando conceituar a solução proposta para esse projeto, foi definido o CANVAS do produto que é apresentado na Figura 1. Nele são descritos os nove componentes que caracterizam o modelo de negócio do sistema desenvolvido.

Figura 1 - Modelo Canvas

Fonte: os autores

3.4.2. Interpretação do CANVAS

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

O projeto consiste em um Sistema Gerenciador de Estacionamentos Automotivos, utilizando dispositivos microcontrolados para a facilitação da identificação das vagas disponíveis, e a implementação das regras de negócio operacionais de processos com clientes.

O Segmento de Clientes almejado pelo produto consiste em um mercado de massa, representado pelos proprietários de estacionamentos automotivos privados. Os clientes possuem necessidades e problemas similares que pretende-se atender com a solução.

Para atingir o objetivo de conquistar os clientes, a Proposta de Valor consiste em um sistema *web* de gerenciamento dos processos e regras de negócio do estacionamento e o controle de entrada/saída de veículos com o monitoramento em tempo real de vagas disponíveis. Com este produto o Cliente resolve problemas de gestão do negócio, bem como, facilita e organiza os dados de seus clientes e controla e executa o planejamento financeiro do empreendimento, o que contribui para que o estacionamento melhore seu desempenho operacional com redução de custos e maiores facilidades para seus clientes.

Os Canais de Comunicação inicialmente serão operacionalizados de forma direta. Haverá portais de comunicações através da internet, buscando apresentar os principais pontos fortes do sistema e os benefícios de seu uso no ambiente de negócio. O interessado poderá solicitar uma apresentação do funcionamento do produto e, tendo interesse, poderá realizar a compra da solução de forma *online* ou através da Equipe de Venda. Após a venda haverá um prazo para a implantação do sistema no local de operação do Estacionamento, que será previamente agendado. Feita a implantação, são disponibilizadas as formas de contato com a equipe de Suporte através de um Help Desk.

O Relacionamento com os Clientes será através de assistência pessoal por um Representante do negócio em contato com os interessados e utilizadores do produto para auxiliar nos testes, vendas e suporte da solução; além de comunidades criadas em um ambiente de integração dos usuários do produto, para *feedback*, troca de experiências, de conhecimentos, dentre outras.

A Receita provém de renda recorrente, oriunda de assinatura da utilização do sistema com periodicidade mensal, referente a valores de licença de uso e suporte técnico ao sistema. Poderão também ocorrer eventuais transações de receita resultantes de pagamentos únicos como implantação do sistema, treinamento dos usuários, e customizações.

Os principais Recursos utilizados para entrega da Proposta de Valor serão os recursos físicos, que considera o espaço para o desenvolvimento e adequação dos dispositivos microcontrolados ao sistema *web*, e recursos humanos para atendimento aos clientes, desenvolvimento e manutenção do sistema, entre outros.

As Atividades Chave do Modelo de Negócio consistem no desenvolvimento da aplicação *web*, passando pela implantação do sistema e, posteriormente, no serviço de suporte com a plataforma de Help Desk.

Como plataformas utilizadas para o desenvolvimento e a manutenção do *software*, são utilizadas soluções gratuitas do mercado de ferramentas de

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

desenvolvimento, tais como a linguagem JAVA e seus *frameworks* para *front-end* e *back-end* como a IDE NetBeans e o SGBD PostgreSQL.

Inicialmente, a gestão do negócio será direcionada pela gestão de custos, concentrando-se em minimizar custos sempre que possível. Os custos fixos considerados no estudo são os salários das pessoas ligadas ao projeto, e os custos variáveis são os relacionados à manutenção dos equipamentos e ambientes de operação das ferramentas de desenvolvimento.

4 Desenvolvimento

Nesta seção são apresentadas as etapas realizadas pelo processo de desenvolvimento da solução proposta. Em destaque, pode-se observar a modelagem BPMN, bem como os Diagramas de Atividades e de Banco de Dados. Também serão demonstrados detalhes da codificação.

4.1 Notação BPMN

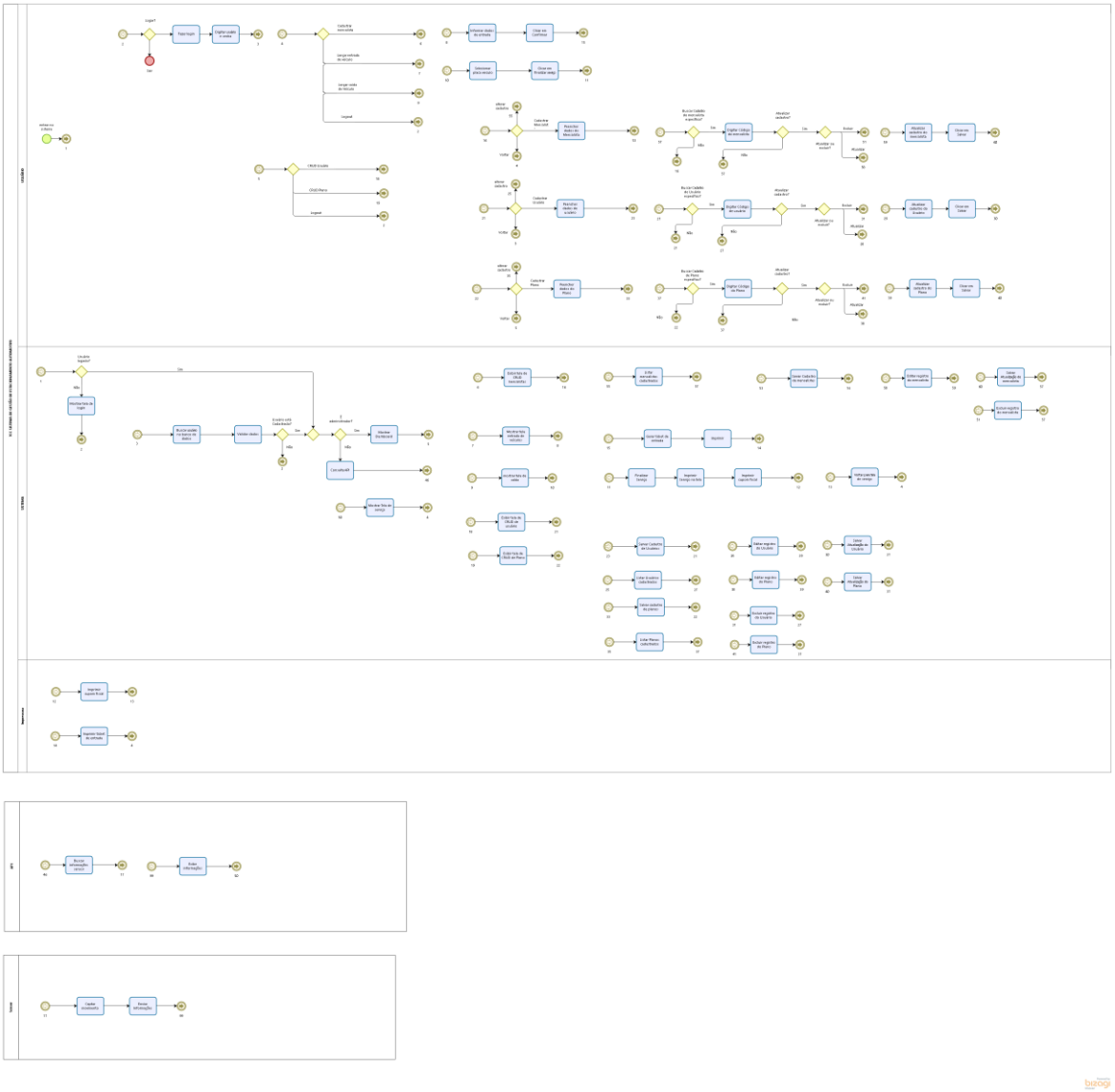
A técnica de modelagem de processos refere-se à representação gráfica das ações de negócio de uma empresa. Essa concepção é importante, pois possibilita a compreensão correta do funcionamento dos processos organizacionais, destacando a geração de valor para os clientes. A melhor maneira de facilitar a modelagem e negócios é com a Notação e Modelagem de Processos de Negócio (BPMN do termo em inglês Business Process Model and Notation (ALMEIDA, 2017).

Estse padrão possui um conjunto de símbolos e regras que permite modelar diferentes fluxos de processos, com vários níveis de detalhamento. A BPMN surgiu a partir do esforço coletivo entre várias empresas de utilizar ferramentas de modelagem, uniformizando a maneira de se modelar processos, buscando maximizar a compatibilidade entre sistemas de informação e facilitar a comunicação entre *stakeholders* (ALMEIDA, 2017).

O diagrama BPMN realizado para descrever os fluxos e processos da solução proposta para o problema dos estacionamentos automotivos, é apresentado na Figura 2.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Figura 2 - BPMN



Fonte: os autores

4.2 Diagrama de Classes

Buscando representar as classes que compõem o sistema *web* a ser desenvolvido como solução proposta, será utilizado por esse projeto como recurso o Diagrama de Classes da UML, representado na Figura 3.

Procurando formalizar a apresentação de Diagrama de Classes, um bom conceito de classe foi estabelecido por Ventura (2018):

> No contexto de produção software, podemos entender que uma Classe é uma abstração de um objeto da vida real (vida real que será tratada via software), que agrupa dados (atributos) e procedimentos (operações) relacionados ao seu contexto.

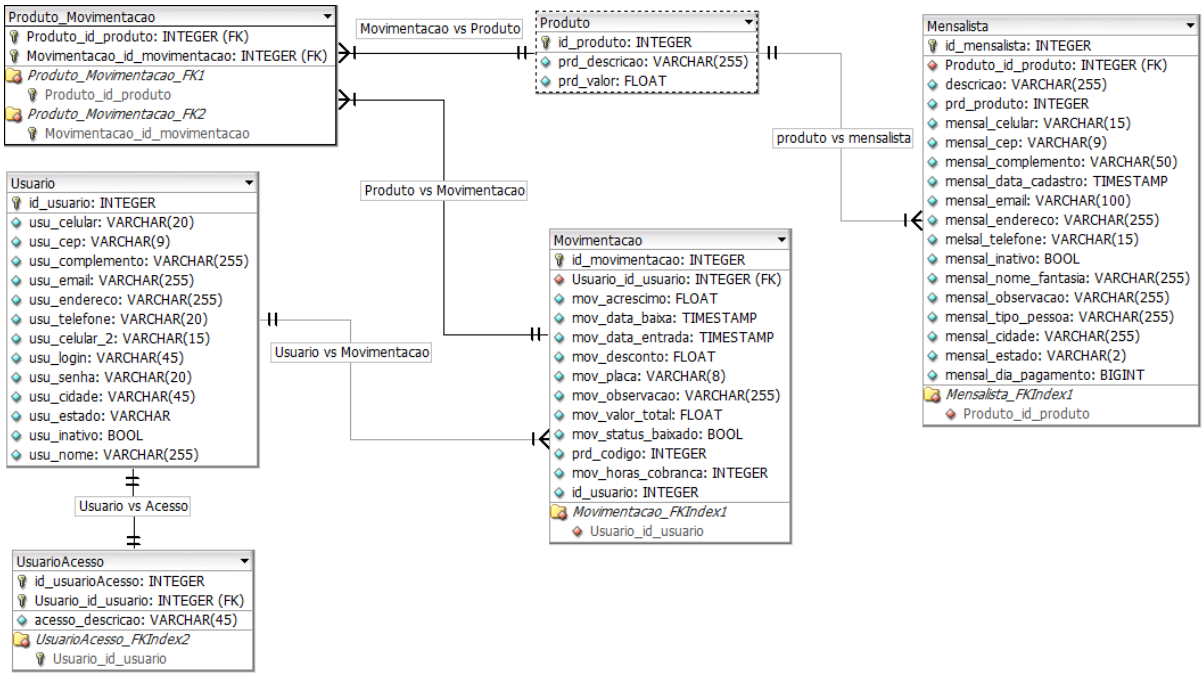
Em resumo, pode-se estabelecer que o Diagrama de Classes trata de um conceito empregado na área de Engenharia de Software, usado para representar

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

a estrutura de uma solução (sistema), apresentando suas classes, atributos, operações e as relações entre os objetos (SIGNIFICADOS, 2018).

Este tipo de representação é útil na construção de *softwares*, pois indica todas as classes que o sistema necessita, além de servir como base para a produção de outros diagramas que auxiliam na definição do escopo do projeto.

Figura 3 - Diagrama de Classes



Fonte: os autores

4.3 Diagrama de Sequência

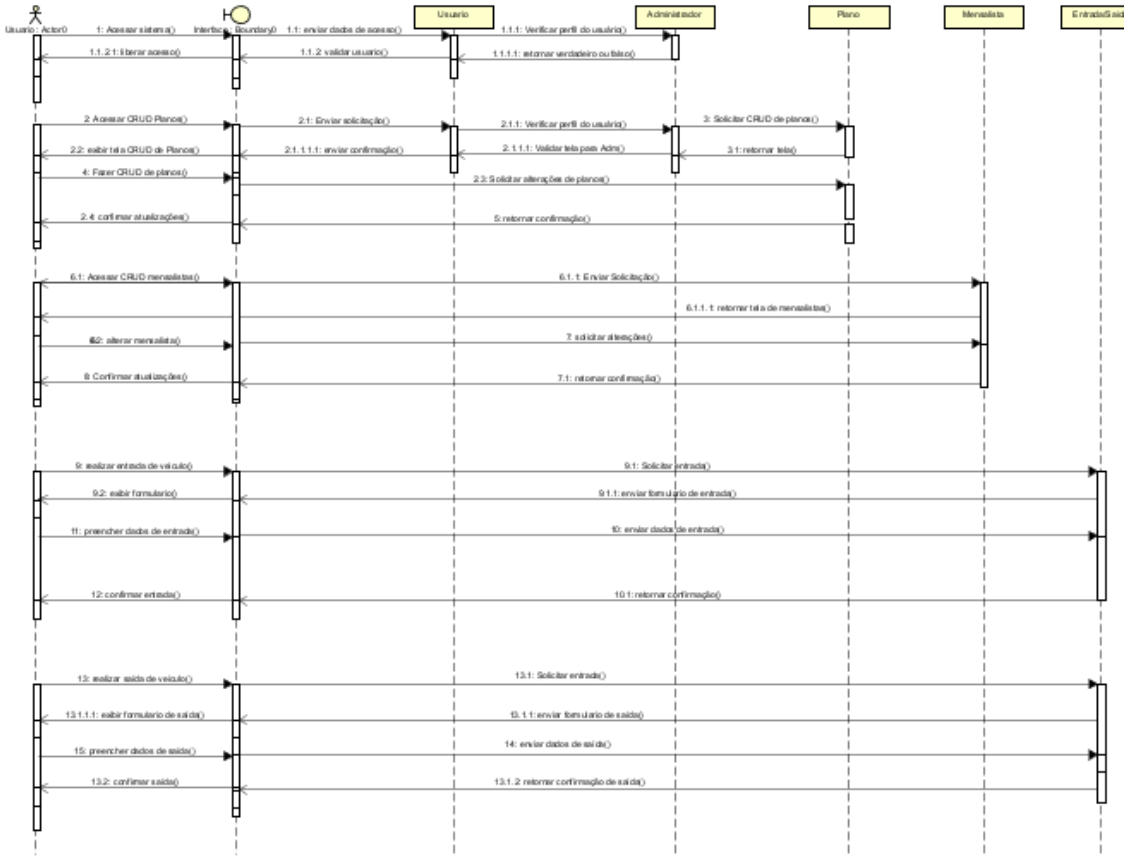
O Diagrama de Sequência (DS) é uma ferramenta da UML, que possui o objetivo de representar o desenrolar, no tempo, de processos e de mensagens passadas entre objetos em um *software* de computador.

O DS é um instrumento para apresentar e ilustrar como as funções de um sistema se relacionam, buscando atingir um objetivo. Auxilia os desenvolvedores e engenheiros de *software* a delimitar o escopo das funcionalidades, proporcionando a compreensão da finalidade da solução (VENTURA, 2018).

O Diagrama de Sequência que foi desenvolvido para esta aplicação, encontra-se representado pela Figura 4.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Figura 4 - Diagrama de sequência



Fonte: os autores

4.4 Diagrama de Casos de Uso

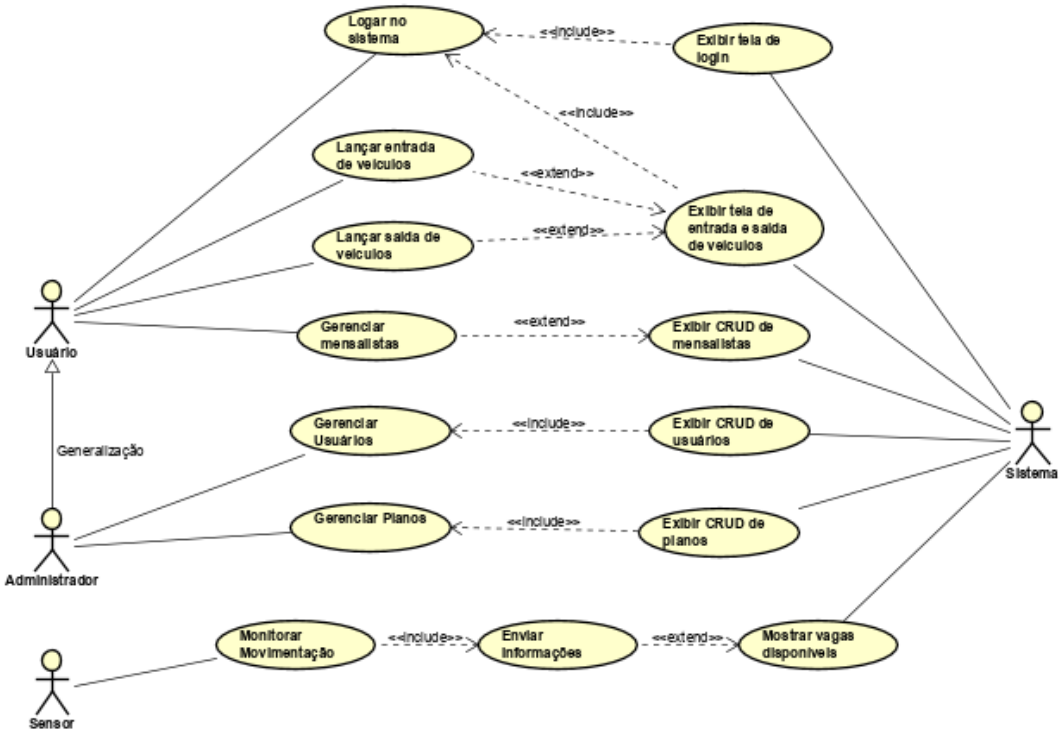
O Diagrama de Casos de Uso é o diagrama mais geral e informal da UML, utilizado na fase de levantamento e análise de requisitos para representar as principais funcionalidades (usos) do sistema/*software* (SILVA, PANSANATO e FABRI, 2010).

A técnica utilizada para construir um Diagrama de Casos de Uso, compreende identificar os atores (usuários ou outros sistemas) e os serviços que o sistema fornecerá aos atores, conhecidos como Casos de Uso (SILVA, PANSANATO e FABRI, 2010).

Procurando representar mais claramente os atores e serviços que compõem o sistema *web* projetado como solução, foi produzido o Diagrama de Casos de Uso demonstrado na Figura 5.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Figura 5 - Diagrama de casos de uso



Fonte: os autores

4.5 Codificação

Será apresentada, durante essa sessão, alguns exemplos da codificação do sistema. Na figura 6 é exemplificado uma parte do menu do sistema onde é possível ver a utilização de alguns recursos de *FrontEnd* como: *PrimeFaces*, CSS, controle de acesso utilizando enumeradores e rotas para outras telas do sistema com o evento *onclick.*

Figura 6 – *Template* Principal

*templatePrincipal.xhtml

```
86
87          <sec:ifAnyGranted roles="USER, ADMIN">
88              <p:submenu label="Cadastros" id="subMenuCadastro">
89
90                  <sec:ifAnyGranted roles="USER, ADMIN">
91                      <p:menuitem value="Mensalista" id="menumensalista" style="color:#264059;"
92                          onclick="redirecionarPage('#{request.contextPath}','/cadastro/', 'find_mensalista')"
93                          ajax="true" />
94                  </sec:ifAnyGranted>
95
96                  <sec:ifAnyGranted roles="ADMIN">
97                      <p:menuitem value="Funcionário" id="menufuncionario" style="color:#264059;"
98                          onclick="redirecionarPage('#{request.contextPath}','/cadastro/', 'find_funcionario')"
99                          ajax="true" />
100                 </sec:ifAnyGranted>
101
102                 <sec:ifAnyGranted roles="ADMIN">
103                     <p:menuitem value="Produto" id="menuproduto" style="color:#264059;"
104                         onclick="redirecionarPage('#{request.contextPath}','/cadastro/', 'find_produto')"
105                         ajax="true" />
106                 </sec:ifAnyGranted>
107
```

Fonte: os autores

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

A seguir, na figura 7, será mostrado o mapeamento de alguns atributos da classe produto utilizando o *Framework Hibernate* para fazer a persistência com o banco de dados

Figura 7 - *Model Class* Produto

*Produto.java

```
1  package br.com.project.model.classes;
2
3  import java.io.Serializable;
21
22 @Audited
23 @Entity
24 @Table(name = "produto")
25 @SequenceGenerator(name = "produto_seq", sequenceName = "produto_seq", initialValue = 1, allocationSize = 1)
26 public class Produto implements Serializable {
27
28     private static final long serialVersionUID = 1L;
29
30     @IdentificaCampoPesquisa(descricaoCampo = "Código", campoConsulta = "prd_codigo")
31     @Id
32     @Column(name = "prd_codigo")
33     @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "produto_seq")
34     private Long prd_codigo;
35
36     @IdentificaCampoPesquisa(descricaoCampo = "Descrição", campoConsulta = "prd_descricao", principal = 1)
37     @Column(nullable = false)
38     private String prd_descricao;
```

Fonte: os autores

O trecho de código apresentado na Figura 8 abaixo apresenta dois métodos da classe Produto. O primeiro está sobrescrevendo o método *toString* com alguns atributos do produto. O outro é um método que retorna um *Json* com os principais atributos.

Figura 8 – Métodos da classe produto

```
@Override
public String toString() {
    return "Produto [prd_codigo=" + prd_codigo + ", prd_descricao="
            + prd_descricao + "]";
}

public JSONObject getJson() {
    Map<Object, Object> map = new HashMap<Object, Object>();
    map.put("prd_codigo", prd_codigo);
    map.put("prd_descricao", prd_descricao);
    return new JSONObject(map);
}
```

Fonte: os autores

O trecho representado na Figura 9 demonstra os métodos *getter* (método assessor) e o método *setter* (método modificador) para o controle de acesso e modificação do sistema.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Figura 9 – Métodos *Getter* e *Setter*

```
public Long getPrd_codigo() {
    return prd_codigo;
}

public void setPrd_codigo(Long prd_codigo) {
    this.prd_codigo = prd_codigo;
}

public String getPrd_descricao() {
    return prd_descricao;
}

public void setPrd_descricao(String prd_descricao) {
    this.prd_descricao = prd_descricao;
}
```

Fonte: os autores

Por último, o trecho de código demonstrado na Figura 10 exemplifica um arquivo Java script com exemplos de funções que auxiliaram o sistema como, por exemplo, a função addMascaraDecimalMonettaria que converte um número em valor monetário e a função validarSenhaLogin, que valida a senha e login do usuário

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Figura 10 - Funções

*scripty.js

```
98 function addMascaraDecimalMonetaria(id) {
99     var id = getValorElementPorId(id);
100    if (id != idundefined) {
101        jQuery(function($){
102            $("#"+id).maskMoney({precision:2, decimal:",", thousands:"."});
103        });
104    }
105
106 }
107 function validarSenhaLogin() {
108    j_username = document.getElementById('j_username').value;
109    j_password = document.getElementById('j_password').value;
110
111    if (j_username === null || j_username.trim() === '') {
112        alert("Informe o Login.");
113         $("#j_username").focus();
114        return false;
115    }
116    if (j_password === null || j_password.trim() === '') {
117        alert("Informe a Senha.");
118         $("#j_password").focus();
119        return false;
120    }
121
122    return true;
123    }
```

Fonte: os autores

5 Resultados

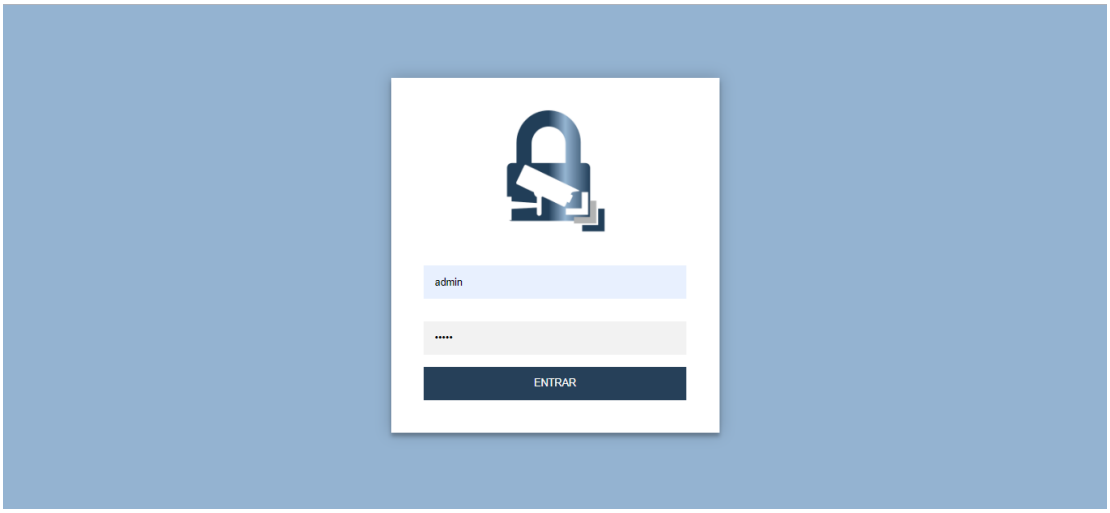
Nessa sessão são apresentados os resultados do desenvolvimento do sistema *web* projetado como solução para a melhoria dos processos e da produtividade em estacionamentos privados.

O *software* foi desenvolvido utilizando os *frameworks*: JavaServer Faces (JSF) + PrimeFaces para o *frontend*; Hibernate (para a persistência no banco de dados), e o Spring Framework para a injeção de dependência. Além disso foi utilizado o SGBD PostgreSQL e o sistema foi construído com base no padrão de arquitetura de *software* MVC.

A interface do sistema e suas funcionalidades são descritas a partir da Figura 11 que apresenta a tela de Login do sistema, por onde o usuário fará o acesso ao sistema.

Figura 11 - Tela de Login

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca



Fonte: os autores

Após inserir os dados, é validado se o usuário informado existe no banco de dados, caso não existir será exibida uma mensagem de erro. Caso contrário, o usuário tem acesso à tela inicial do sistema, conforme exibido na Figura 12, na qual é exibido um menu lateral contendo as opções de funcionalidades do sistema, bem como a identificação do usuário logado (canto superior direito).

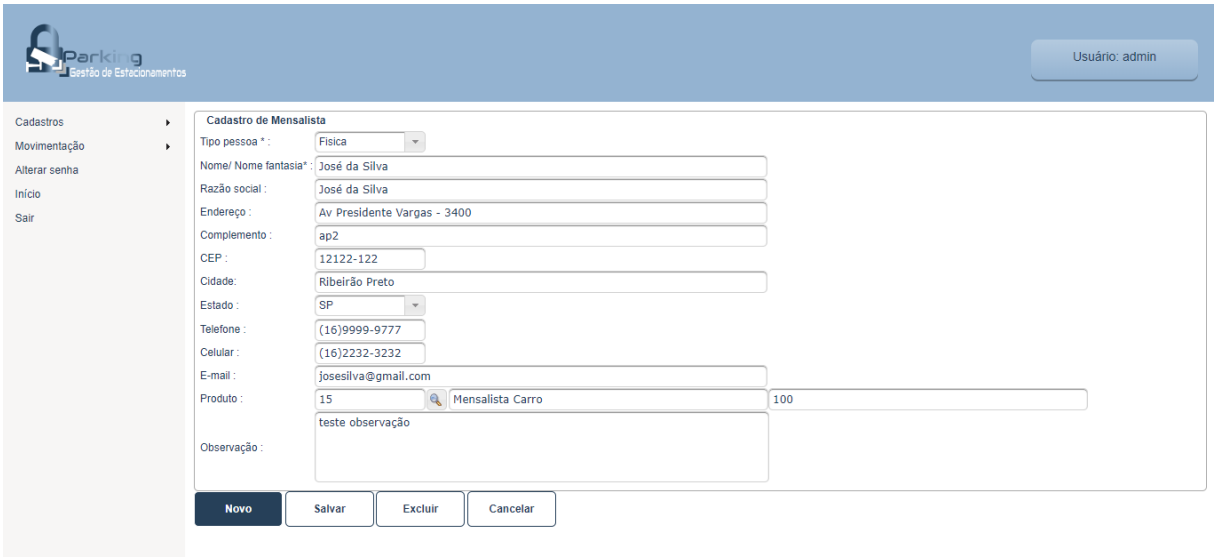
Figura 12 - Tela inicial

Fonte: os autores

O sistema, por padrão, possui uma seção de Cadastros onde é possível cadastrar Funcionários, Mensalistas e Produtos (tipos de veículo aceitos no estabelecimento), um exemplo deste tipo de rotina está demonstrada na Figura 13, com a tela de Cadastro de Mensalistas.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

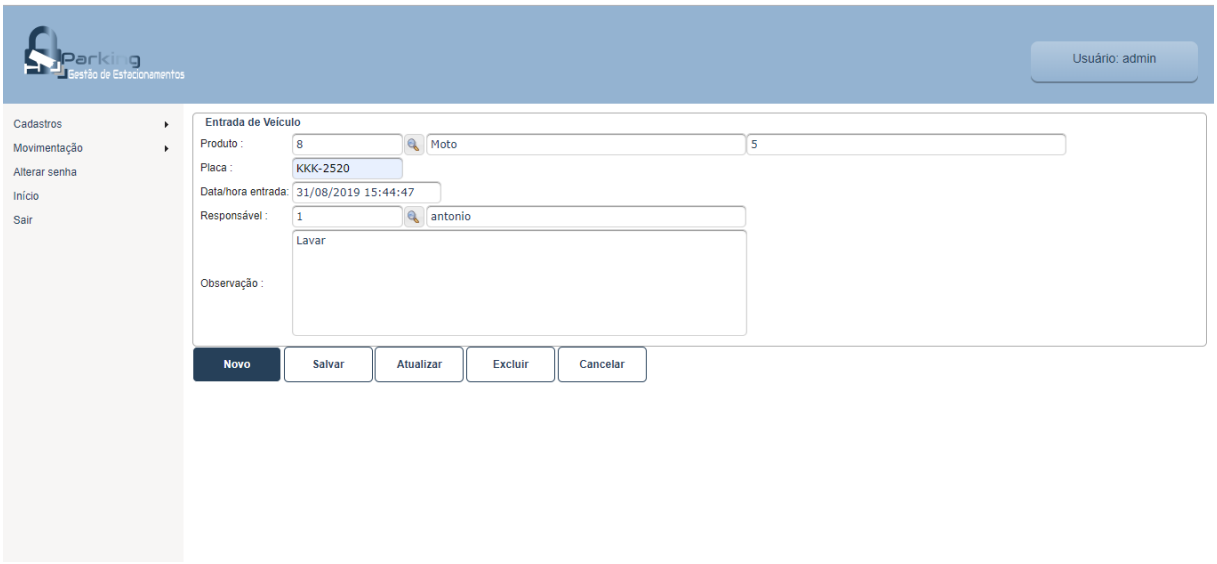
Figura 13 - Cadastro de mensalistas



Fonte: os autores

Outra seção do sistema é a de Movimentações, onde as regras de negócio e gestão são executadas. Nesta parte, o usuário pode informar Entradas e Saídas de veículos, conforme mostrado na Figura 14 (Entrada de Veículo) e na Figura 15 (Baixa de Entrada de Veículo), bem como consultar todas as movimentações de veículos na tela.

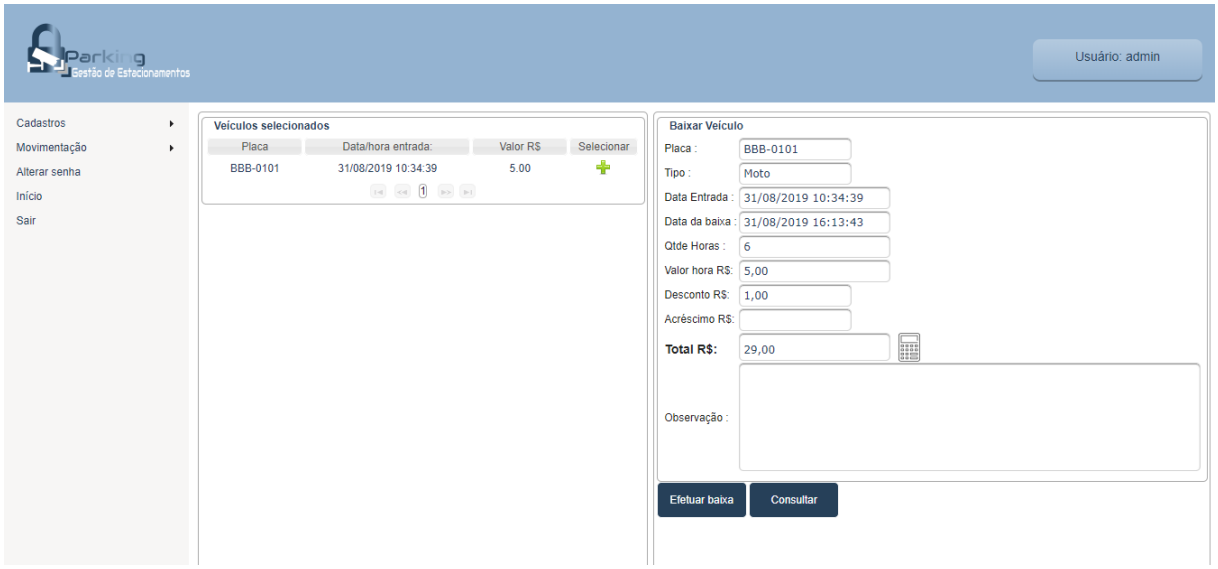
Figura 14 - Entrada de Veículo



Fonte: Os autores

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

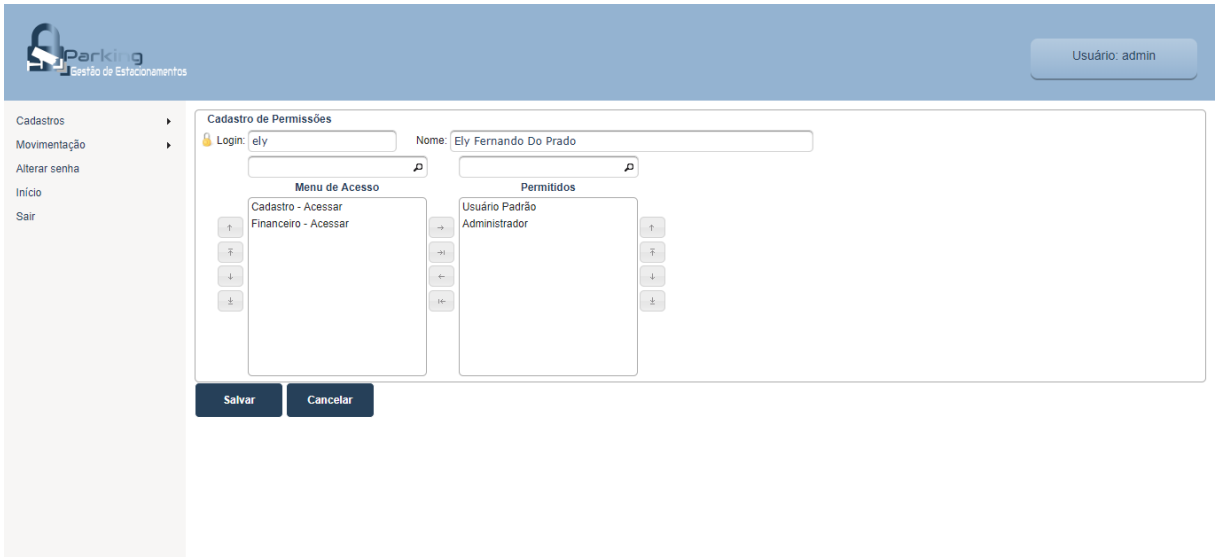
Figura 6 - Baixa de veículos



Fonte: os autores

O acesso a todas essas rotinas é controlado pelo usuário Administrador, que pode vincular os usuários que devem ter acesso a cada rotina. Isso é possível na tela de Controle de Permissões, conforme mostrado na Figura 16.

Figura 7 - Controle de Permissões



Fonte: os autores

6 Conclusão

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

O principal objetivo deste artigo, consiste em apresentar a análise, o projeto e a implementação de uma solução *web* que visa possibilitar o aumento da eficácia dos processos que ocorrem em um estacionamento automotivo. Para tanto foi necessária a integração de *frameworks* baseados na linguagem Java com o SGBD PostgreSQL para a criação de um sistema que implementa os módulos necessários para o correto gerenciamento do funcionamento de um estabelecimento, ou seja, contendo rotinas de cadastro de Clientes, de Produtos e Mensalistas, bem como os processos de movimentação de entrada e saída veículos do estacionamento, complementados com um controle de permissões por rotina para os usuários do *software*.

Durante o desenvolvimento, foram enfrentados desafios para a integração dos diferentes *frameworks*: JSF, Hibernate e Spring para a criação de um ambiente dinâmico para o usuário do sistema. Foram necessários estudos e testes de rotinas para se alcançar a solução mínima viável.

Finalizando, almeja-se implementar a solução proposta comercialmente no futuro e para isso, mais funcionalidades deverão ser incluídas no sistema. Destaca-se, dentre elas, o uso de dispositivos microcontrolados para controle de identificação de vagas disponíveis e ocupadas no estacionamento. Para isso, pretende-se utilizar a plataforma Arduino para a criação do sistema para detectar se a vaga encontra-se disponível ou não.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

Referências

ALMEIDA, Marcelo Lucas de. Elementos finitos paramétricos implementados em Java, 2005.

ALMEIDA, Vinicius Nóbile de. Euax Consulting. Site Euax Consulting, 2017. Disponivel em: <https://www.euax.com.br/2017/02/o-que-e-bpmn-business-process-model-and-notation/>. Acesso em: 24 ago. 2019.

AVILA, Walter Marlon Mamedes. Pra que serve o Hibernate? Fábrica de Software, 2013. Disponivel em: <http://fabrica.ms.senac.br/2013/06/pra-que-serve-o-hibernate/>. Acesso em: 01 set. 2019.

CARVALHO, Vinicius. PostgreSQL**:** Banco de dados para aplicações web modernas. [S.l.]: Casa do código, 2017.

DEVMEDIA. Como começar com SPRING. Devmedia, 2019. Disponivel em: <https://www.devmedia.com.br/exemplo/como-comecar-com-spring/73>. Acesso em: 01 set. 2019.

HISRICH, Robert; PETERS, Michael; SHEPHERD, Dean. Empreendedorismo. [S.l.]: AMGH, 2014.

INDRUSIAK, Leandro Soares. Linguagem Java. Grupo JavaRS JUG Rio Grande do Sul, 1996.

JAVASERVER Faces Technology. ORACLE, 2019. Disponivel em: <https://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html>. Acesso em: 01 set. 2019.

LUCIANO, Josué; ALVES, Joel Barberá Wallison. PADRÃO DE ARQUITETURA MVC: MODEL-VIEWCONTROLLER. Revista EPeQ Fafibe, v. I, n. 3, 2011.

MILANI, André. PostgreSQL - Guia do Programador. 1ª. ed. São Paulo: Novatec Editora, 2008.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

MOBILIDADE URBANA E ESTACIONAMENTO. INDIGO, 2017. Disponível em: <https://www.parkindigo.com.br/empresas/estacionamento-e-mobilidade-urbana/>. Acesso em: 01 nov. 2019.

NARDES, Felipe Bruno Souza; MIRANDA, Roberto Campos da Rocha. Lean Startup e Canvas: uma proposta de metodologia para startups. Revista Brasileira de Administração Científica, Aquidabã, v. V, n. 3, Julho 2014.

OSTERWALDER, Alexander; PIGNEUR, Yves. Business Model Generation - Inovação em Modelos de Negócios. 5º. ed. Rio de Janeiro: Alta Books, 2011.

REENSKAUG , Trygve; COPLIEN, James. The DCI Architecture: A New Vision of Object-Oriented Programming, Março 2009.

RIES, Eric. A startup enxuta**:** como os empreendedores atuais utilizam a inovação continua para criar empresas extremamente bem-sucedidas. São Paulo: Texto Editora Ltda, 2012.

RODRIGUES, Carlos S. et al. Hibernate. Departamento de Informática e Estatística Universidade Federal de Santa Catarina, 2010. Disponivel em: <http://www.inf.ufsc.br/~frank.siqueira/INE5612/Seminario2010.2/Hibernate.pdf>. Acesso em: 01 set. 2019.

SIGNIFICADOS , 2018. Disponivel em: <https://www.significados.com.br/diagrama-de-classes/>. Acesso em: 31 Agosto 2019.

SILVA, Christiane; PANSANATO, Luciano; FABRI, J.. Ensinando Diagramas UML para Estudantes Cegos, Cornélio Procópio, 2010.

TUTORIAL JSF 2,0. Mkyong.com, 2010. Disponivel em: <https://www.mkyong.com/tutorials/jsf-2-0-tutorials/>. Acesso em: 01 set. 2019.

VENTURA, Plínio. Entendendo o Diagrama de Classes da UML. Até o momento**.**, 2018. Disponivel em: <https://www.ateomomento.com.br/uml-diagrama-de-classes/>. Acesso em: 31 Agosto 2019.

VENTURA, Plínio. Entendendo o Diagrama de Sequência da UML. Até o momento**.**, 2018. Disponivel em: <https://www.ateomomento.com.br/diagrama-de-sequencia-uml/>. Acesso em: 31 Agosto 2019.

RESIGeT On-Line
Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica
Vol.10 Nro. 1 2019
Uni-FACEF Centro Universitário Municipal de Franca

ZUGMAN, Fabio; CASTOR, Belmiro Valverde Jobim. Dicionário de Termos de Estratégia Empresarial. [S.l.]: Editora Atlas, 2009.