Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

VICTOR HUGO ELIAS DE MATTOS

SISTEMA PARA VENDA DE PRODUTOS UTILIZANDO QR CODE COM ANDROID

Assis
2015

VICTOR HUGO ELIAS DE MATTOS

SISTEMA PARA VENDA DE PRODUTOS UTILIZANDO QR CODE COM ANDROID

Trabalho de Conclusão de Curso apresentado ao Curso de Ciência da Computação do Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação.

Orientador: Dr. Luiz Ricardo Begosso.

Área de Concentração: Informática.

Assis
2015

FICHA CATALOGRÁFICA

MATTOS, Victor Hugo Elias
Sistema para venda de produtos utilizando QR Code com Android/Victor Hugo Elias de Mattos. Fundação Educacional do Município de Assis – FEMA – Assis, 2015.
87p.

Orientador: Dr. Luiz Ricardo Begosso.
Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

1. Java.2.QR Code.3. Android.

CDD:001.6
Biblioteca da FEMA

SISTEMA PARA VENDA DE PRODUTOS UTILIZANDO QR CODE COM ANDROID

VICTOR HUGO ELIAS DE MATTOS

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Bacharelado em Ciência da Computação analisado pela seguinte comissão examinadora:

Orientador: Dr. Luiz Ricardo Begosso.

Analisador(1):

Assis
2015

DEDICATÓRIA

Dedico este trabalho trabalhos aos meus pais Elza e Donizeti, e a minha namorada Lais, por estarem sempre me apoiando e me motivando.

AGRADECIMENTOS

Primeiramente agradecer a Deus por ter me dado sabedoria e ter me guiado durante esses anos de faculdade, por ter me dado forças quando não tive mais e por me fortalecer cada vez mais para eu ir em busca dos meus sonhos.

Ao professor Dr. Luiz Ricardo Begosso, pela orientação e pelo constate estimulo transmitido durante o trabalho e por sempre tirar minhas dúvidas nas horas que eu mais precisei de ajuda.

Aos amigos que me motivaram ou me ajudaram e a todos que colaboraram direta ou indiretamente na execução deste trabalho.

Aos meus amigos do Projeto Rede Ciranda, por me ajudarem em alguns erros e na parte do desenvolvimento do trabalho.

Aos meus pais Elza e Donizeti, por sempre me ajudar nas horas que eu mais precisei de ânimo, e por me motivar a cada dia que se passa. Guerreiros que nunca me deixaram e nunca me esqueceram.

A minha namorada Lais, pelo apoio psicológico e por me motivar nos dias mais difíceis, e por me ajudar nas horas que eu mais precisei de ajuda.

A todos os professores do Curso de Ciência da Computação da FEMA, por sempre valorizarem os alunos que estudam na FEMA, sempre ajudarem nessa jornada no curso, e sempre mostrando os caminhos a seguir para se dar bem no mercado de trabalho.

RESUMO

Com o grande avanço das tecnologias, grandes empresas estão querendo reduzir custos, excluindo a mão de obra humana, para terem mais lucro. O uso destas novas tecnologias podem sair caro no começo, mas podem valer a pena por longo prazo, por tratar-se de um investimento alto no início, não um investimento mensal, diferente da mão de obra humana.

Com o surgimento de novas tecnologias flexíveis e a necessidade de conhecer novas tecnologias, surge a motivação para construir um aplicativo, na área comercial que atenda várias necessidades de grandes e pequenas empresas, que estão dispostas a ter um investimento para adquirir produtos com novas tecnologias.

Este aplicativo terá seu foco voltado a um sistema de compras de supermercado utilizando a plataforma Android com a tecnologia QR Code, controlado por um sistema via web, usando a linguagem de programação Java para web e para a aplicação Android, buscando uma maior contribuição para a área de tecnologia com essa nova tendência mundial.

Palavras Chave: Java; QR Code; Android.

ABSTRACT

With the breakthrough technologies, large companies are looking to reduce costs by excluding the human labor, to have more profit. The use of these new technologies can be expensive at first, but can be worth for long term, because it can be an investment only in the beginning, not a monthly investment, different from human labor.

With new flexible technologies and the need to meet new technologies, the motivation appears to build an application in the commercial area that meets various needs of large enterprises and small businesses, who are willing to have an investment to purchase products with new technologies.

This application will focus to a grocery shopping system using Android plataform with QR Code technology, controlled by a web-based system using the Java programming language for the web and for the Android application, seeking greater contribution to the technology with this new global trend.

Keyword: Java; QR Code; Android.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURA DE SIGLAS

QR	Quick Responde
iOS	iPhone OS
SO	Sistema Operacional
SDK	Software Development Kit
ISO	International Organization for Standardization
URL	Uniform Resource Locator
SMS	Short Message Service
API	Application Programming Interface
JVM	Java Virtual Machine
HTTP	Hypertext Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
EJB	Enterprise JavaBeans
J2SDK	Java 2 SDK Standard Edition
EDGE	Explicit Data Graph Execution
GSM	Global System for Mobile
GPS	Global Positioning System
ADTF	Analysist and Design Task Force
OMG	Object Management Group
RTF	Revision Task Force
FTF	Finalization Task Force
OOAD	Análise e design orientado a objetos
UML	Unified Modeling Language
SWT	Standard Widget Toolkit
CRUD	Create, Read, Update e Delete
RPC	Chamadas Remotas de Procedimento
XML	eXtensible Markup Language
Java EE	Java Enterprise Edition
OTI	Object Technology International Inc.
IBM	Internacional Business Machines
IDE	Integrated Development Environment

WSDL	Web Service Description Language
JSF	JavaServer Faces

SUMÁRIO

1. INTRODUÇÃO

Com o grande avanço da tecnologia, o número de celulares e dispositivos móveis vem tendo um crescimento gigantesco no mercado tecnológico. Com esse grande avanço tecnológico e com a falta de tempo sendo cada vez maiores para as pessoas, elas estão buscando a cada dia mais facilidade e praticidade com o uso das tecnologias disponíveis, e que não ocupe muito tempo e seus esforços sejam mínimos.

Um dos grandes avanços tecnológicos em dispositivos moveis e celulares foi que eles não são apenas restritos para ligações e mandar mensagem. O fato é que as pessoas possuem um super mini-computador e muitos não sabem que ele tem essa grande capacidade de processando e armazenamento. Nos dias de hoje, telefones celulares são bem menores que os antigos, com mais recursos que os de antigamente.

Mas não teve apenas a evolução do hardware, o grande responsável pelo hardware funcionar é o software. Os softwares também tiveram um grande avanço, devido aos novos hardwares desenvolvidos e também pelos Sistemas Operacionais desenvolvidos pela Google, o Android e pela Apple, o iOS.

A facilidade e praticidade de um dispositivo móvel ou celular realizar algum processo, o tornou a nova tendência para o mercado mundial. A grande característica que tornou os celulares uma tendência mundial, é a sua portabilidade, por ser um dispositivo pequeno e com um grande poder.

O Android é responsável por uma grande parte desse avanço, pelo simples fato de ser um sistema aberto, fazendo com que desenvolvedores, desenvolvam seus aplicativos e coloquem seus aplicativos para download para o usuário, alguns aplicativos pagos e outros gratuitos, mas não ficam restritos somente para o desenvolvedor do SO, colocar seu aplicativo para download.

O local onde os desenvolvedores do Android deixam seus aplicativos para download é na Google Play, onde possuem várias categorias, vários aplicativos para todos os gostos e para várias áreas, sendo para lazer, cuidar da saúde, raciocínio, lazer, etc.

O Android é o sistema operacional mais utilizado nos dias de hoje. O seu código aberto faz com que vários fabricantes de dispositivos móveis o utilizem como o seu sistema operacional, podendo tornar o Android cada vez mais forte no mercado

tecnológico, já que disputa com o iOS da Apple. Mas o Android tem uma vantagem, já que o Sistema Operacional iOS é somente utilizado nos produtos da Apple, enquanto o Android é utilizado por várias empresas.

1.1. OBJETIVO

Este trabalho tem como objetivo a implementação de um sistema de vendas no ambiente Android utilizando a tecnologia QR Code, para diminuição de gastos empresariais e a facilidade para o usuário fazer sua compra.

1.2. PÚBLICO ALVO

O público alvo deste trabalho são empresas das diversas áreas comerciais.

1.3. JUSTIFICATIVA

Com o avanço da ciência, novas tecnologias estão sendo desenvolvidas nas mais diversas áreas de aplicação. Esse trabalho se justifica pela tentativa de reduzir os custos operacionais da empresa. Acredita-se que a tecnologia QR-Code pode oferecer um diferencial para a empresa que a utiliza, gerando agilidade nos negócios.

1.4. METODOLOGIA

Para o desenvolvimento deste trabalho serão realizadas pesquisas intensificadas sobre o funcionamento da área de venda no varejo. As ferramentas utilizadas nesse projeto serão: o SDK do Android para o desenvolvimento do aplicativo, a plataforma Eclipse, a linguagem Java para fazer a implementação do software principal, o Banco de Dados My SQL para fazer a conexão.

1.5. ESTRUTURA DO TRABALHO

O presente trabalho está dividido em cinco capítulos.
O primeiro capítulo, descreve a Introdução, que estabelece os objetivos, o público alvo, justificativa e a metodologia.
O segundo capítulo aborda a fundamentação teórica, apresentando as tecnologias usadas para o desenvolvimento.
O terceiro capítulo descreve a modelagem do software, os diagramas e os casos de uso.
O quarto capítulo descreve os detalhes do desenvolvimento do trabalho.
O quinto capítulo descreve a conclusão do trabalho.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. QR CODE

O QR Code foi criado pela Denso, uma das principais empresas do grupo Toyota, e foi aprovada como padrão internacional ISO (ISO/IEC18004) em junho de 2000.

Figura 1: Exemplo de QR Code

A Figura 1 mostra um exemplo de Qr Code. Este símbolo no começo foi usado para o controle da produção de peças de automóveis, mas com o passar do tempo foi se propagando para outros campos.

A sua praticidade e por ser de graça, já que a Denso disponibilizou a patente para domínio público, tornou-a um código “perfeito”, devido as opções que ela pode oferecer, sendo superior aos outros códigos de barra e pela praticidade que pode ser usada para várias coisas, não somente para referenciar um produto como o código de barra tradicional.

QR Code é um código de barras bidimensional que pode ser escaneado por alguns dispositivos com câmera, através de uma imagem. Esse código pode servir para atender diversos objetivos, como indicar alguma URL, um número telefônico, um contato, uma SMS, um texto, e outras coisas.

Geralmente, símbolos bidimensionais contêm maior quantidade de dados, quando comparado a símbolos lineares, portanto, requerem um maior tempo de processamento de dados e processo mais complexo. Mas o QR Code tem muita consideração por permitir a leitura de alta velocidade.

Segundo Pankiewicsz (2009), algumas características de símbolos bidimensionais é que ele possui um grande volume de dados, sendo 7.089 caracteres numéricos no máximo, Alfanumérico 4.296 no máximo, Binário (8 bits) 2.953 no máximo, Kanji/Kana (alfabeto japonês) 1.817 no máximo, possui a gravação de alta densidade, sendo aproximadamente 10 vezes mais elevadas do que nos símbolos lineares, e sem faltar à leitura de alta velocidade, com isso, o QR Code tem uma superioridade em termos de desempenho e funcionalidades.

A sua superioridade é tanta que ele é capaz de lidar com vários tipos de dados, tais como caracteres numéricos, alfabetos, sinais, caracteres kanji (Língua Japonesa), hiragana (Alfabeto fonético básico do Japonês), alfanuméricos, sinais e imagens.

O QR Code precisa de um aplicativo em dispositivo para ver o seu conteúdo, a sua resposta é rápida. O QR Code geralmente é usado para o uso do marketing, pois ele armazena URLs e direciona a vídeos, pode direcionar para sites eletrônicos de compras ou para outros sites onde tem detalhes sobre o que está comprando.

Esses códigos podem ser criados em sites gratuitamente, podendo-se escolher a opção do tipo que desejar, e quando for escaneado ela mostrará o seu conteúdo, e se for URL ela já levará o usuário para o site.

Lüders (2012) diz que a vantagem está no conforto para o administrador. Através de QR-Codes é possível representar uma grande quantidade de dados, de uma forma compacta. Geralmente os QR-Codes contêm informações como: Endereço de uma página na internet, cartão de visita online, acesso ao Wireless local, ou textos informativos. Escaneando este código, o administrador não precisa mais digitar longos endereços ou combinações numéricas em seu Smartphone.

De acordo com Machado (2012), com a popularização dos celulares e smartphones com acesso à internet, o uso do QR Code é cada vez mais promissor e já se configura como um importante aliado do mercado imobiliário.

2.2. JAVA

Segundo Deitel (2010), Java é uma linguagem de programação de computador que é divertida para iniciantes aprenderem e adequada para programadores experientes utilizarem na construção de sistemas de informações empresariais importantes.

Em 1991, a Sun Microsystems financiou um projeto de pesquisa corporativa interna

que resultou em uma linguagem baseada em C++, chamada de Oak em homenagem a uma árvore de carvalho vista por sua janela na Sun, seu criador foi James Gosling. Esse projeto visava criar tecnologias para aparelhos domésticos e televisão interativa. Depois de um tempo descobriu-se que já existia uma linguagem com este nome, foi quando a equipe da Sun visitou uma cafeteira local, onde o nome Java (cidade de origem do café importado) foi sugerido.

O projeto passou por dificuldades, por causa do mercado de dispositivos eletrônicos inteligentes destinados ao consumidor final não estava se desenvolvendo tão rapidamente quanto a Sun esperava. Foi quando em 1993 a Web explodiu e a Sun viu o potencial de utilizar o Java para adicionar conteúdo dinâmico, como interatividade e animações, às páginas Web.

Em uma conferência em 1995, a Sun anunciou o Java composto pela Java Virtual Machine e pela API, e foi inserido no Netscape, principal *browser* de acesso à Internet usado na época. A comunidade de negócios teve um enorme interesse no Java por causa da interatividade com a Web.

Segundo Mendes, a plataforma Java oferece aos programadores e analistas de sistemas um conjunto completo de classes para o desenvolvimento de sistemas web (Servlet, JavaServer Pages, JavaServer Faces), desktop (Swing, SWT) e batch (método main()). Com essas classes, o tempo para o desenvolvimento é reduzido e a qualidade do sistema fica muito melhor.

Algumas características a tornaram uma linguagem muito conhecida por ser uma linguagem simples, multithread, neutra de arquitetura, segura, inexistência de ponteiros, independência de plataforma, orientada a objetos, gerenciamento automático de alocação e deslocação de memória, interpretada e compilada, necessita de ambiente de execução, precisa da JVM e oferece alto desempenho.

A linguagem Java é derivada das linguagens C e C++, por importar diversas estruturas delas, como o comentário de uma única linha.

Ela é independente de plataforma, o desenvolvedor focará o seu esforço somente no código em si. A maioria das linguagens é preciso gerar uma versão para cada plataforma que deseja utilizar, exigindo alterações no código fonte.

Para criar uma aplicação ou um programa é preciso seguir três passos: Edição, Compilação e Interpretação.

A Edição refere-se à escrita do código fonte, a Compilação gera um código intermediário chamado Bytecode e na Interpretação a JVM analisa e executa cada instrução do Bytecode. A compilação ocorre somente uma vez e a interpretação ocorre a toda vez que o programa for executado. Na Figura 2, mostrará todas as etapas que um programa em Java passa antes de ser executado.

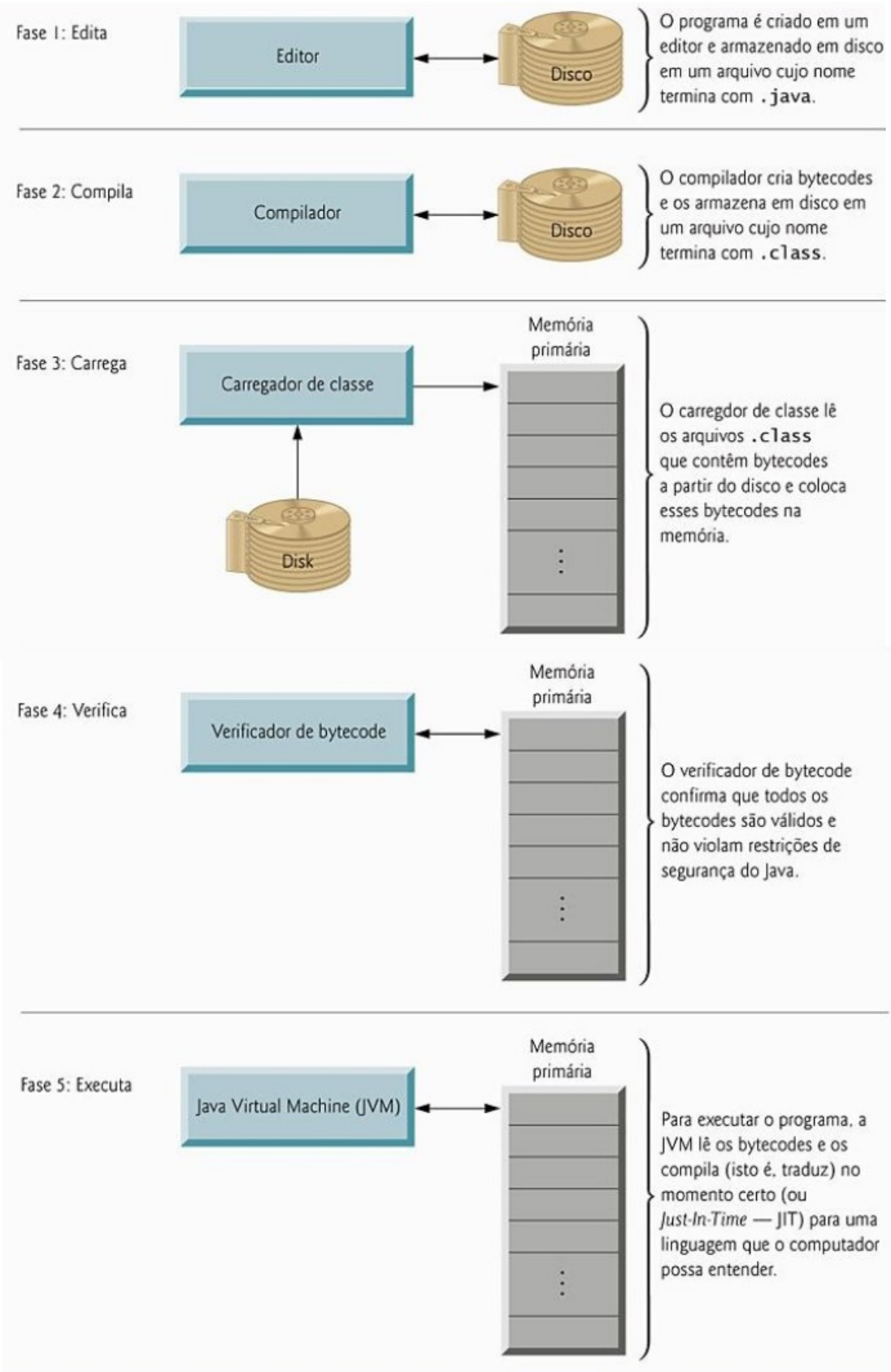


Figura 2: Ambiente típico de desenvolvimento Java (adaptado de Deitel; Deitel (2010))

A JVM é a responsável pela interpretação dos bytecodes, pela execução do código.
A JVM verifica se o código não viola a integridade e segurança da plataforma, interpretador e carrega os arquivos .class para a memória. Na Figura 3 mostra o processo de compilação e execução de um programa Java.

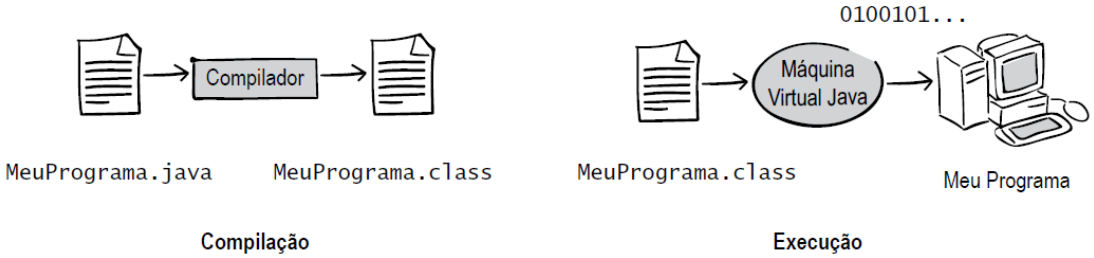


Figura 3: Etapas para compilação e execução de um programa Java (adaptado de Mendes (2009))

Quando se efetuamos requisições ao servidor, através de páginas Web, essas requisições são enviadas via protocolos HTTP ou HTTPS para containers, que podem ser tanto o web container que transforma código em servlets ou para container EJB.
O J2SDK é o conjunto de ferramentas para compilar, depurar, executar e documentar um programa escrito em Java.
Uma classe define o modelo de um objeto, ou seja, todas as características que o objeto contém foram definidas pela classe. As classes possuem métodos que realizam tarefas e algumas retornam informações no final da sua execução. As classes podem ter relacionamentos uma com as outras.
De acordo com Pacievitch em 2011:

> Em 2006 muitas partes do Java estavam sendo passadas para a licença de Software livre, e a maioria já estava disponível para o público gratuitamente, tudo Sob licença GNU. O Java foi uma revolução na interatividade, sua utilização aumenta a cada dia. Java é uma linguagem relativamente simples e dinâmica, permite criar programas e aplicações para a Web sem

depender de outra linguagem.

2.3. ANDROID SDK

De acordo com Queirós (2013), O AndroidTM é um sistema operativo para dispositivos móveis desenvolvido pela Open Handset Alliance, liderada pela Google. O mercado móvel apresentou em 2012 uma grande hegemonia dos dispositivos Android, com mais de metade da quota de mercado na venda de dispositivos móveis. A *Google Play*, loja online de aplicações Android, registou mais de 800 mil aplicações e mais de 30 biliões de downloads foram feitos até 2013 entre aplicações pagas e gratuitas.

Em Outubro de 2003, Andry Rubin, Rich Miner, Nick Sears e Chris White fundaram a Android Inc. em Pato Alto, Califórnia, EUA. Com a grande repercussão, a Google em agosto de 2005 resolveu adquirir esta empresa. Após dois anos, várias empresas viram o potencial que estaria prestes a surgiu, então, em 5 de novembro de 2007 foi criado o Consórcio Open Hadset Alliance, um grupo formado por mais de 30 empresas de tecnologia e Google com líder deste grupo, para acelerar o desenvolvimento de novas aplicações inovadoras. Depois de sete dias do consórcio formado, em 12 de novembro de 2007, o tão aguardado SDK foi liberado para os desenvolvedores, na qual era a primeira versão de muitas que estariam por vir. Em 22 de outubro de 2008, a empresa HTC foi à primeira empresa a lançar um celular com Android, o celular Dream, na qual sua versão era a 1.0. Depois desta versão, as atualizações começaram a ser constantes. Em 2009 foi atualizado o Kernel do Linux e a versão do Android foi atualizada pra 1.1., 1.5, e a partir da versão 1.5 começaram a ser batizados com nome de doces americanos.

Na Figura 4, é possível visualizar todas as empresas que fazer parte do Consórcio Open Hadset Alliance.

Figura 4: Empresas do Consórcio Open Hadset Alliance (adaptado de Garcia (2012))

Em 2010 houve apenas uma atualização do Kernel do Linux e uma do Android. Em 6 de dezembro o Android chegou a versão 2.3 que é a mais utilizada até o momento.

No ano de 2011 a comunidade Android teve um grande marco, a versão do Android 2.3 recebeu algumas mudanças e passou para a versão 2.3.3 e em 22 de fevereiro com alta nos Tablets, a versão do Android 3.0 foi destinada exclusivamente a eles, depois de 2 meses, a versão 3.1 foi lançada no mercado.

O principal lançamento do ano de 2011 foi o lançamento da versão do Android 4.0 onde tinha como proposta a integração e padronização da plataforma, tanto para Smartphones e Tablets, que eles pudessem usar a mesma versão, sem preocupações nenhuma. No dia 19 de outubro é liberado para as operadoras de telefonia e administradores a versão do Android 4.0.

A plataforma Android com certeza é uma das apostas mais audaciosas do Google, o que de fato inseriu a gigantesca empresa no ramo de desenvolvimento para dispositivos móveis. Também revolucionou ao anunciar a plataforma como sendo a primeira de caráter Open Source, por meio da OHA – Open Handset Alliance (Rabello).

A plataforma Android foi à primeira plataforma móvel completa, aberta e livre. A Figura 5 mostra os principais componentes da arquitetura do Android.

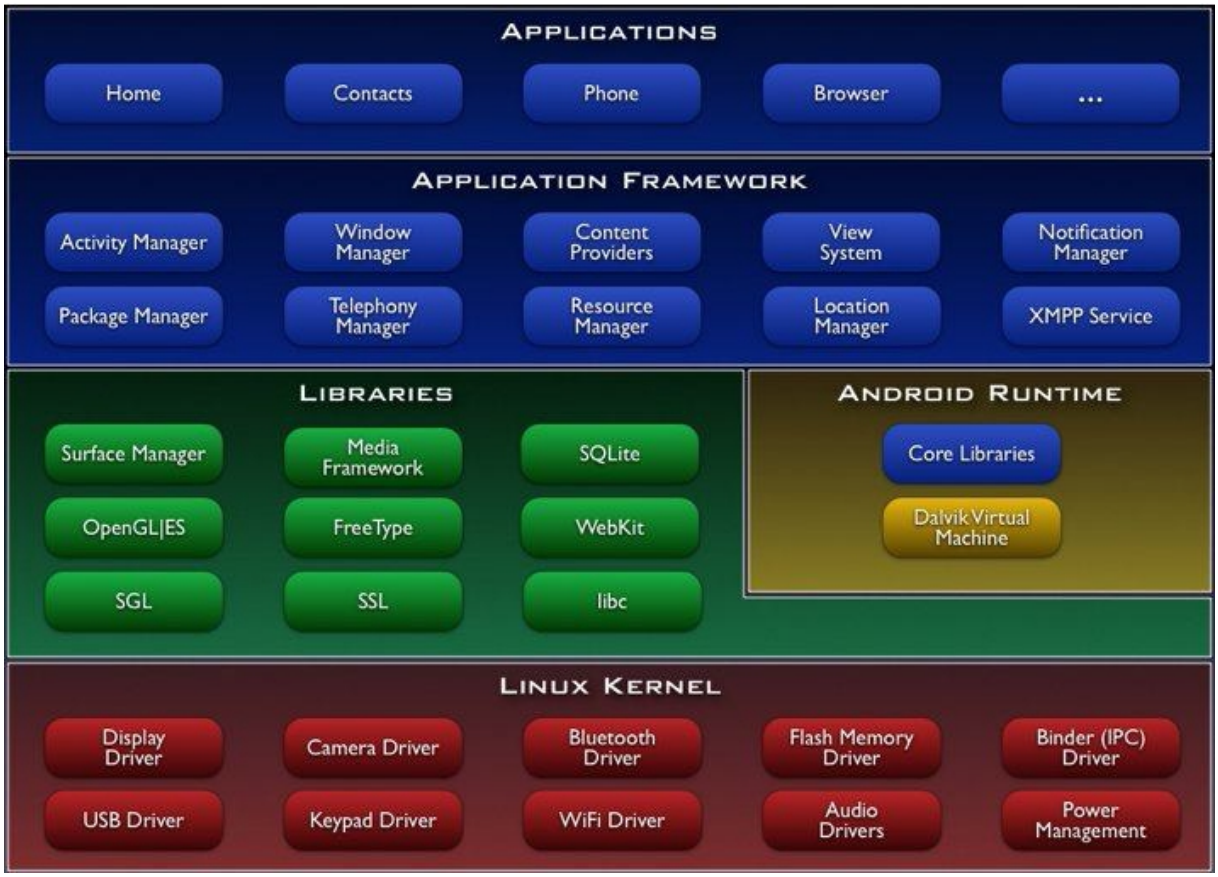


Figura 5: Camadas da arquitetura Android (adaptado de Rabello (2009))

A plataforma Android foi desenvolvida utilizando o sistema operacional Linux, então, as características do Linux foram incorporadas pela Android.

Algumas características suportadas pela plataforma Android: Framework de Aplicação, Máquina Virtual Dalvik, Navegador Web Integrado, Gráficos Otimizados, SQLite, Suporte para mídias, Telefonia GSM, Bluetooth, EDGE, 3G, WiFi, Câmera, GPS, entre outros.

Segundo Catapan (2009), a plataforma Android agrada também aos fabricantes de chips, pois adiciona novos recursos aos aparelhos, que aumenta a busca por processadores mais velozes, impulsionando o desenvolvimento e a comercialização de novos produtos. Como o sistema é código aberto, existe possibilidade de portá-lo para diferentes plataformas conforme necessário, assim como no caso do Linux, que

roda tanto em micros PC, quanto em *clusters* de servidores e *smartphones*.

A máquina virtual utilizado pelo Android é a Dalvik. Todas as aplicações Android rodam no contexto da sua instância de máquina virtual. A Dalvik foi desenvolvida para que os dispositivos possam suportar múltiplas máquinas virtuais eficientemente. A Dalvik executa arquivos na extensão .dex. Um .dex nada mais é do que uma espécie de bytecodes de Java otimizados para a Android.

A plataforma Android possui várias camadas em sua arquitetura: Applications, Application Framework, Libraries, Android Runtime, e Linux Kernel.

O Android por ser aberto, possibilita que os administradors utilizem softwares de vários desenvolvedores, sem depender da empresa fabricante, por isso tem o tornado cada vez mais popular.

De acordo com Garcia (2012), o Android está se tornando um sistema operacional Mobile de fácil manuseio tanto para os usuários, como para os programadores, e com isso, a cada dia surgem novos apps (programas), sendo que até Janeiro de 2012 havia no Play Store (repositório de aplicativos Android) mais de 400 mil aplicativos cadastrados.

2.4. UML

Segundo BOOCH; RUMBAUGH; JACOBSON (2005), pode-se dizer que um caso de uso é um “documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo”.

A UML é um trabalho e ideias de várias pessoas. A criação da UML se iniciou oficialmente em outubro de 1994, quando Rumbaugh juntou-se a Booch na Rational. O foco deste projeto no começo era a unificação dos métodos Booch e OMT. Um rascunho da versão 0.8 do Método Unificado foi lançado em outubro de 1995. Na mesma época, Jacobson se juntou à Rational e o escopo do projeto UML foi expandido com a finalidade de incorporar o OOSE. Os esforços resultaram no lançamento da versão 0.9 da UML em junho de 1996. Em 1996 solicitaram algumas opiniões das comunidades de engenharia de software, neste período, muitas empresas de software consideravam a UML um recurso estratégico para seus negócios. Foi estabelecido um consórcio da UML que desejava dedicar recursos com propósito de trabalhar a favor de uma definição mais forte e completa para as

empresas. Várias empresas contribuíram para a versão 1.0, entre elas estão a HP, IBM, Microsoft, Oracle, Rational, entre outras. A versão 1.0 foi uma linguagem de modelagem bem definida, poderosa e que poderia ser aplicada em vários problemas. Mary Loomis em janeiro de 1997 tentou convencer a OMG a tornar a UML 1.0 como a linguagem padrão de modelagem.

Em 1997, o grupo se expandiu e todos os participantes e colaboradores da proposta inicial foram incluídos ao OMG. Um grupo de tarefas liderado por Cris Kobryn da MCI Systemhouse e administrado por Ed Eykholt da Rational foi criado, com a intenção de formalizar a especificação da UML e íntegra-lá a outros esforços de padronização, e a versão escolhida para padronização foi à versão 1.1 da UML, que foi oferecida à OMG em julho de 1997. Em setembro do mesmo ano, a versão foi aceita pela ADTF e pelo Marachitere Board do OMG, e em seguida, ela foi submetida a votos dos membros da OMG. A OMG adotou a UML 1.1 em 14 de setembro de 1997.

A RTF assumiu a manutenção da UML e lançou várias versões, com um novo grupo e maiores parceiros eles produziram a versão UML 2.0, que foi revista por um ano pela FTF, coordenada por Branc Selic da IBM, e está versão foi adotada pela OMG no início de 2005.

A UML é um esquema de representação gráfica mais utilizada para modelar sistemas orientados a objetos. Ela proporciona preparações de planos de arquitetura de projetos de sistema, incluindo aspectos conceituais, como processos de negócios e funções do sistema, além de banco de dados e componentes de software, entre outras coisas. A UML é independente de plataforma.

O diagrama de casos de uso tem como objetivo, auxiliar a comunicação entre os analistas de sistemas e o cliente. A sua função é descrever um cenário que vai mostrar a ação que cada usuário vai pode fazer no sistema. Ele é representado por atores, casos de uso e relacionamento de caso de uso com os atores.

O ator é representado por um boneco e um rótulo com o nome do ator. Esse ator pode ser um usuário humano ou pode ser outro sistema computacional.

O caso de uso é representado por uma elipse e um rótulo com o nome do caso de uso. O caso define o que o usuário pode fazer no sistema. Na Figura 6 é possível visualizar como é composto o diagrama de Caso de Uso.

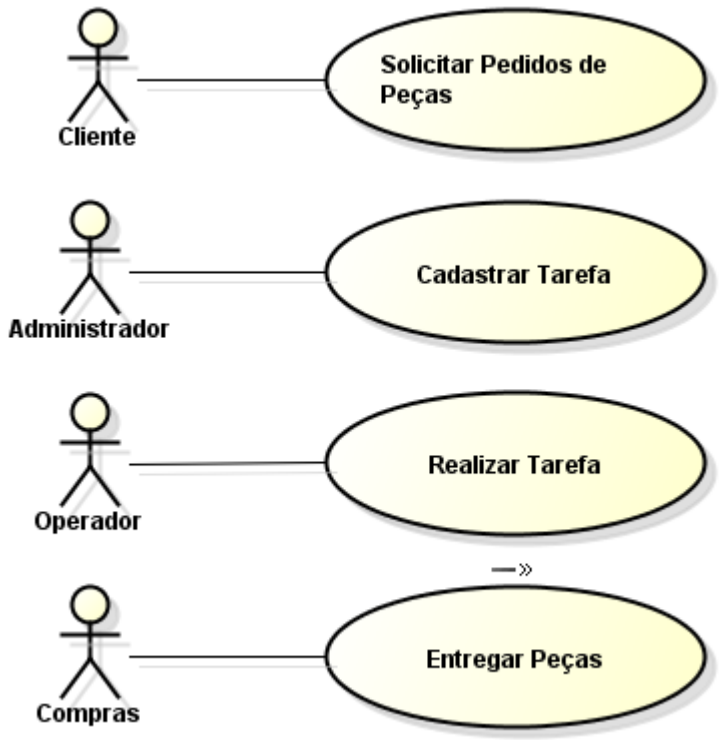


Figura 6: Exemplo do diagrama do Caso de Uso

De acordo com Macoratti (2004), na UML, o modelo de casos de uso consiste de diagramas de casos de uso que mostram os atores, os casos de uso e seus relacionamentos. Os elementos gráficos que representam atores, casos de uso e sistemas.

Um recurso que chama a atenção dessa ferramenta é a sua flexibilidade. Ela é extensível e é independente de qualquer processo OOAD particular.

“Na UML, o pacote tem uma aplicação mais ampla do que a empregada na linguagem Java, tendo em vista que um pacote pode conter outras visões do sistema como diagramas de casos de uso, componentes etc. Do ponto de vista estrutural, um pacote pode conter os classificadores e seus relacionamentos, diagramas de classe e outros pacotes” (Veronese; Correa; Werner; Felipe; Netto, 2002, p.348).

Na UML uma associação é representada por uma linha ligando as classes que se relacionam. As linhas de associação n-ária se interceptam em um losango nomeado. Uma classe abstrata na UML é representada com seu nome em itálico.

Conforme Deitel (2010):

> A UML 2 tornou-se a linguagem de modelagem gráfica preferida para projetar sistemas orientados a objetos. Utilizamos diagramas de atividade UML para demonstrar o fluxo de controle em cada umas das instruções de controle Java e utilizamos diagramas de classe UML para representar visualmente classes e seus relacionamentos de herança.

2.5. WEB SERVICE

De acordo com Dantas (2007), *Web Services* são componentes que permitem às aplicações enviar e receber dados em formato XML. Cada aplicação pode ter sua própria “linguagem”, que é traduzida para uma linguagem universal, o formato XML.
O *Web Service* é usado para disponibilizar um serviço interativo na *web,* para outras aplicações terem acesso. A melhor forma para a comunicação entre as aplicações, é fazer a comunicação através do protocolo HTTP.

2.6. ECLIPSE

O Eclipse é uma IDE para desenvolvimento Java, mas pode suportar outras linguagens a partir de *plugins*. Foi desenvolvido em Java e é *open source*.
De acordo com Quaresma (2011), o Eclipse é uma plataforma *open source* desenvolvida para construir ambientes integrados que podem ser usados para desenvolvimento de aplicações de diversos tipos, como por exemplo, web sites, programas C++, Java e J2EE. Foi criada pela OTI e IBM e é uma ferramenta genérica e com arquitetura aberta e extensível baseada em *plugins*.

2.7. SOAP

SOAP é um protocolo baseado em XML para troca de informações em um ambiente distribuído. O SOAP é usado para troca de mensagens entre aplicações na rede. Essas aplicações ou *Web Services* possuem interface de acesso simples.
Dantas (2007) define SOAP como um protocolo projetado para invocar aplicações remotas através de RPC ou trocas de mensagens, em um ambiente independente

de plataforma e linguagem de programação. SOAP é um padrão normalmente aceito para utilizar-se com Web Services. Desta forma, pretende-se garantir a interoperabilidade e intercomunicação entre diferentes sistemas, através da utilização de uma linguagem XML e um mecanismo de transporte HTTP padrão.

2.8. APACHE TOMCAT

O *Apache Tomcat* foi desenvolvido pela empresa Apache Software Foundation e é distribuído como *software* livre.

É um servidor *web* que implementa as tecnologias Java *Servlet* e *JavaServer Pages.* Tem capacidade de atuar como servidor *web* ou pode funcionar integrado a um servidor web dedicado. Como servidor web, provê um servidor *web* HTTP puramente em Java. Este servidor inclui ferramentas para configuração e gerenciamento, que podem ser editados em arquivos de configuração XML.

2.9. MYSQL

O MySQL é um sistema de gerenciamento de banco de dados, que utiliza a linguagem SQL. Este banco de dados é um dos mais populares do mundo, entre os usuários estão a NASA, o Banco Bradesco, HP, Nokia, Sony, entre outras grandes organizações e empresas. O MySQL é um software proprietário, mas para uso sem fins lucrativos, não precisa-se comprar a sua licença.

2.10. AXIS2

De acordo com o site Wikipédia (2014), o Apache Axis é um *framework* de código aberto, baseado na linguagem Java e no padrão XML, utilizado para construção de *web services* no padrão SOAP. Através do Axis os desenvolvedores podem criar aplicações distribuídas. O projeto Apache Axis é suportado pela *Apache Software Foundation.* O Axis gera um arquivo WSDL automaticamente que contém a definição da interface dos *web services*.

2.11. ZXING

Eduardo (2012), Zxing (Zebra *Crossing*) é um software *open-source* que implementa em Java uma biblioteca de processamento de código de barras em vários formatos

1D/2D, com portas para outras linguagens. O foco é sobre como usar a câmera embutida nos dispositivos para ler e decodificar códigos de barras no aparelho, sem se comunicar com um servidor. O projeto Zxing pode ser usado para codificar ou decodificar códigos de barras em *desktops* e servidores.

2.12. JAVA EE

Java EE é uma plataforma ampla usada para reduzir significativamente o custo e a complexidade do desenvolvimento, implantação e gerenciamento de aplicações de várias camadas centradas no servidor, pois, usa um conjunto de tecnologias coordenadas. Foi desenvolvida sobre a plataforma Java SE e oferece um conjunto de APIs para desenvolvimento e execução de aplicações portáteis, robustas, escaláveis, confiáveis e seguras no lado do servidor.

2.13. PRIMEFACES

De acordo com Games (2012), *PrimeFaces* é uma biblioteca de componentes de código aberto para o JSF 2.0 com mais de 100 componentes, permitindo criar interfaces ricas para aplicações web de forma simplificada e eficiente. O PrimeFaces é um *framework* da empresa *Prime Teknoloki* que oferece componentes ricos. Esses componentes foram construídos para trabalhar com AJAX por *default*. O *PrimeFaces* permite a aplicação de temas com o objetivo de mudar a aparência dos componentes de forma simples.

2.14. XML

Pereira (2009), XML é uma linguagem de marcação recomendada pela *World Wide Web Consortium* para a criação de documentos com dados organizados hierarquicamente, tais como textos, banco de dados ou desenhos vetoriais. O XML traz uma sintaxe básica que pode ser utilizada para compartilhar informações entre diferentes computadores e aplicações. Quando combinado com outros padrões, torna possível definir o conteúdo de um documento separadamente de seu formato, tornando simples para reutilizar o código em outras aplicações para diferentes propósitos.

2.15. IREPORT

A ferramenta Ireport, é uma ferramenta de designer usada para relatórios, que utilizam a biblioteca Java para relatórios a JasperReports. Com esta ferramenta é possível fazer layouts sofisticados, contendo gráficos, imagens, sub-relatórios, tabelas de referência cruzada e muito mais. Os dados podem ser acessados via JDBC, TableModels, JavaBeans, XML, Hibernate e CSV. Os relatórios podem ser na extensão PDF, RTF, XML, XLS, CSV, HTML, XHTML, textos, DOCX ou OpenOffice.

3. DOCUMENTAÇÃO

3.1. LEVANTAMENTO DE REQUISITOS

- Cadastro de clientes;

(nome, CPF, RG, rua, número, bairro, cidade, estado, telefone, celular, e-mail)

- Cadastro de produtos;

(descrição, marca, departamento, referência, quantidade, valor custo, valor venda, fornecedor)

- Cadastro de filial;

(nome, rua, número, bairro, cidade, estado, telefone, e-mail, fax)

- Cadastro de fornecedores;

(nome, CNPJ, rua, número, bairro, cidade, estado, telefone, e-mail, fax

- Cadastro de funcionários;

(nome, CPF, RG, rua, número, bairro, cidade, estado, telefone, celular, e-mail, salário)

- Realizar vendas;

(dia, horário, cliente, valor total)

- Saída de produtos;

(produto, dia, observação)

- Relatório de clientes;
- Relatório dos clientes que mais compram;
- Relatório de produtos;
- Relatório de vendas;
- Relatório de saída dos produtos;

3.2. DIAGRAMA DE ENTIDADE E RELACIONAMENTO (DER)

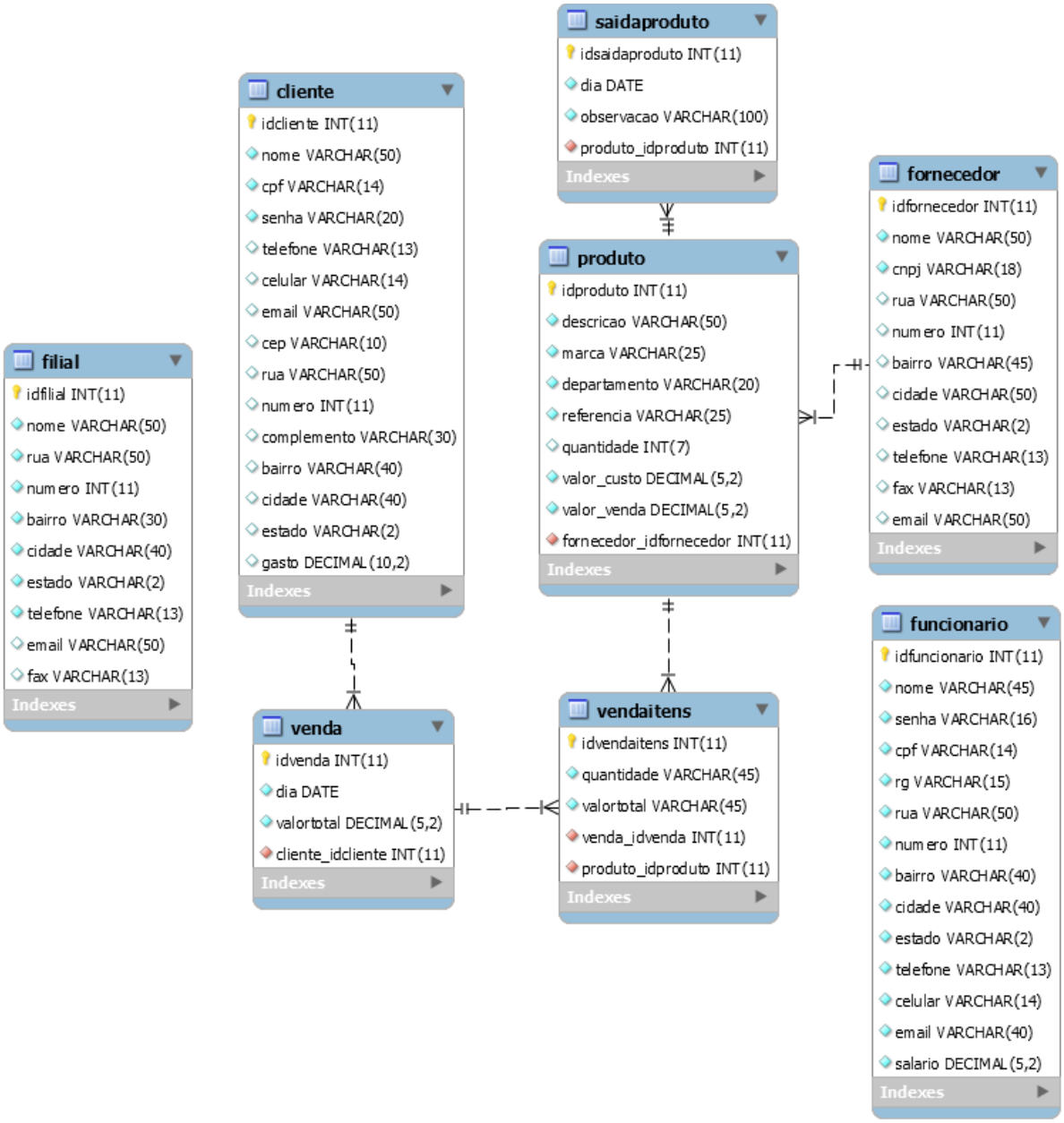


Figura 7: Diagrama de Entidade e Relacionamento

3.3. DIAGRAMA DE CLASSE

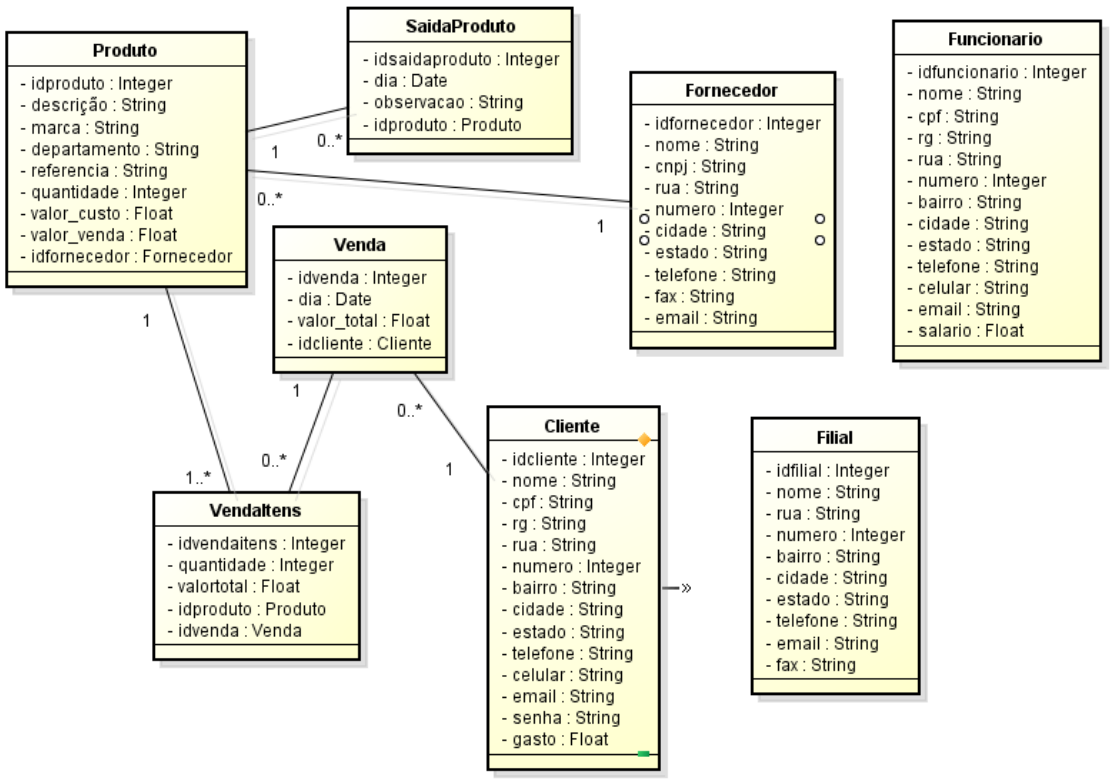


Figura 8: Diagrama de Classe

3.4. DIAGRAMA DE CASO DE USO

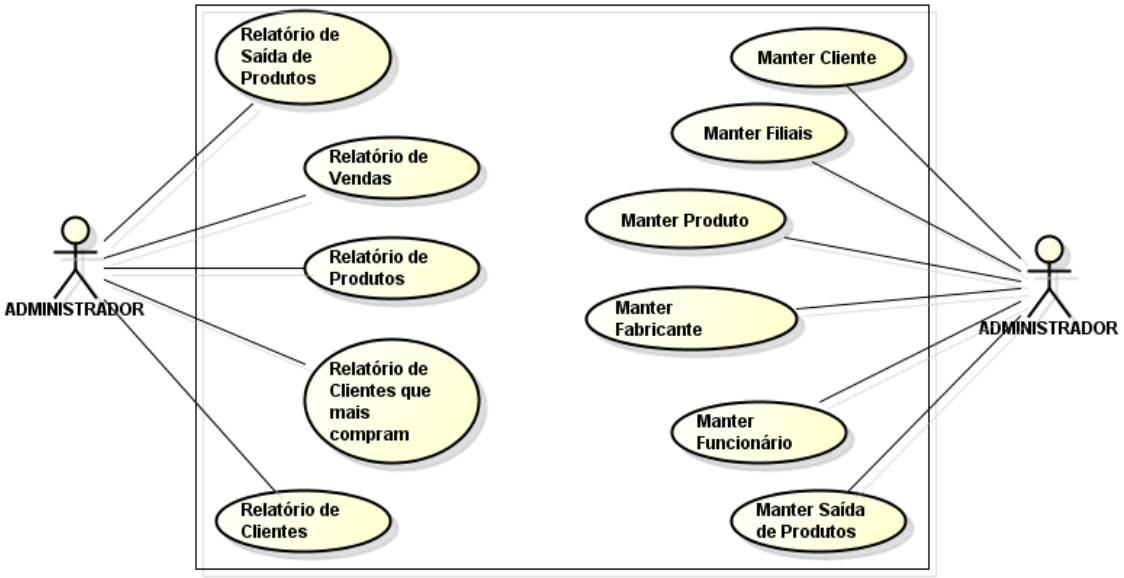


Figura 9: Diagrama de Caso de Uso do Administrador

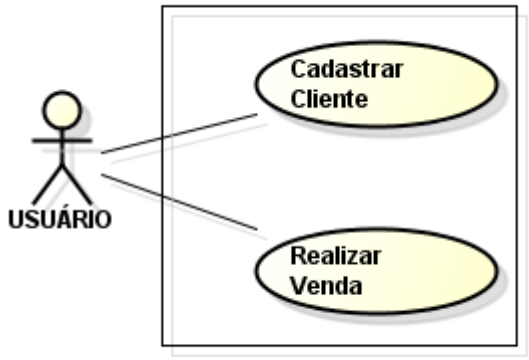


Figura 10: Diagrama de Caso de Uso do Usuário

3.5. CASOS DE USO

3.5.1. Manter Cliente

Figura 11: UC 01: Manter Cliente

3.5.1. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os dados, cadastrar, pesquisar, editar e excluir dados do cliente.

3.5.1.2. Atores

- Administrador.

3.5.1.3. Pré-condições

- Login do administrador efetuado com sucesso.

3.5.1.4. Evento inicial

- O administrador escolhe alguma opção no menu Cliente.

3.5.1.5. Fluxo principal

3.5.1.5.1. Cadastrar

a. Quando o administrador selecionar a opção Novo Cliente no menu Cliente, o sistema exibirá um formulário limpo com os campos: nome, CPF, RG, rua, número, bairro, cidade, estado, telefone, celular, e-mail.

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o cliente foi inserido com sucesso; (E2)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.1.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.5.1.1.b..

3.5.1.5.2. Pesquisar

a. Quando o administrador selecionar a opção Clientes no menu Cliente, o sistema exibe um formulário de clientes cadastrados;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os clientes de acordo com o código ou nome que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.1.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter cliente cadastrado

a. O sistema realiza a busca e verifica que não existem clientes cadastrados

com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.1.5.3. Editar

a. O sistema exibe uma lista de clientes cadastrados;

b. O administrador faz a busca na lista, escolhe o cadastro a ser alterado e efetua a edição do cliente; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o cliente foi atualizado;

d. O caso de uso é encerrado

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.1.5.3.b. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.1.5.3.b.

3.5.1.5.4. Excluir

a. O sistema exibe uma lista de clientes cadastrados;

b. O administrador faz a busca, escolhe o cliente a ser excluído e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do cliente, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o cadastro do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.1.5.4.b. do fluxo principal.

3.5.2. Manter Filial

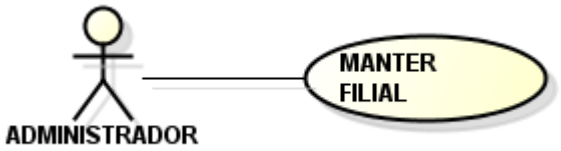


Figura 12: UC 02: Manter Filial

3.5.2.1. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os dados, cadastrar, pesquisar, editar e excluir dados das filiais.

3.5.2.2. Atores

- Administrador.

3.5.2.3. Pré-condições

- Login do administrador efetuado com sucesso.

3.5.2.4. Evento inicial

- O administrador escolhe alguma opção no menu Filiais.

3.5.2.5. Fluxo principal

3.5.2.5.1. Cadastrar

a. Quando o administrador selecionar a opção Nova Filial no menu Filial, o sistema exibirá um formulário limpo com os campos: nome, rua, número, bairro, cidade, estado, telefone, e-mail, fax;

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o filial foi inserido com sucesso;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.2.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.2.5.1.b..

3.5.2.5.2. Pesquisar

a. O sistema exibe uma lista com as filiais cadastradas;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os filiais de acordo com o código ou nome que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.2.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter filiais cadastrados

a. O sistema realiza a busca e verifica que não existem filiais cadastrados com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.2.5.3. Editar

a. O sistema exibe uma lista com as filiais cadastradas;

b. O administrador faz a busca na lista, escolhe o filial a ser alterado e efetua a edição da filial; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que a filial foi atualizado;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.2.5.3.a. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.2.5.3.b.

3.5.2.5.4. Excluir

a. O sistema exibe um formulário de filiais cadastrados;

b. O administrador faz a busca e escolhe a filial a ser excluído na lista e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do filial, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o filial do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.2.5.4.a. do fluxo principal.

3.5.3. Manter Produto

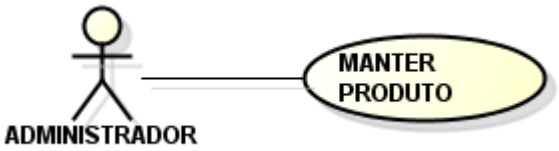


Figura 13: UC 03: Manter Produto

3.5.3.5. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os dados, cadastrar, pesquisar, editar e excluir dados dos produtos.

3.5.2.2. Atores

- Administrador.

3.5.3.3. Pré-condições

- Login do administrador efetuado com sucesso e fabricante cadastrado.

3.5.3.4. Evento inicial

- O administrador escolhe alguma opção no menu Produtos.

3.5.3.5. Fluxo principal

3.5.3.5.1. Cadastrar

a. Quando o administrador selecionar a opção Novo Produto no menu Produto, o sistema exibirá um formulário limpo com os campos: descrição, marca, departamento, referência, quantidade, valor custo, valor venda, fornecedor.

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi inserido com sucesso;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta o passo 3.5.3.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.3.5.1.b..

3.5.3.5.2. Pesquisar

a. O sistema exibe uma lista com os produtos cadastrados;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os produtos de acordo com o código ou nome que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.3.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter produtos cadastrados

a. O sistema realiza a busca e verifica que não existem produtos cadastrados com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.3.5.3. Editar

a. O sistema exibe uma lista de produtos cadastrados;

b. O administrador faz a busca na lista, escolhe o produto a ser alterado e efetua a edição do produto; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi atualizado;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.3.5.3.b. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.3.5.3.b.

3.5.3.5.4. Excluir

a. O sistema exibe uma lista dos produtos cadastrados;

b. O administrador faz a busca, escolhe o produto a ser excluído na lista e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do produto, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o produto do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.3.5.4.a. do fluxo principal.

3.5.4. Manter Fabricante

Figura 14: UC 04: Manter Fabricante

3.5.4.1. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os dados, cadastrar, pesquisar, editar e excluir dados dos fabricantes.

3.5.4.2. Atores

- Administrador.

3.5.4.3. Pré-condições

- Login do administrador efetuado com sucesso.

3.5.4.4. Evento inicial

- O administrador escolhe alguma opção no menu Fabricantes.

3.5.4.5. Fluxo principal

3.5.4.5.1. Cadastrar

a. Quando o administrador selecionar a opção Novo Fabricante no menu Fabricantes, o sistema exibirá um formulário limpo com os campos: nome, CNPJ, rua, número, bairro, cidade, estado, telefone, e-mail, fax.

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o fabricante foi inserido com sucesso;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.4.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.4.5.1.b.

3.5.4.5.2. Pesquisar

a. O sistema exibe uma lista de fabricantes cadastrados;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os fabricantes de acordo com o código ou descrição que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.4.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter fabricantes cadastrados

a. O sistema realiza a busca e verifica que não existem fabricantes cadastrados com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.4.5.3. Editar

a. O sistema exibe uma lista de fabricantes cadastrados;

b. O administrador faz a busca na lista, escolhe o fabricante a ser alterado e efetua a edição do fabricante; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi atualizado;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.4.5.3.a. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.4.5.3.b..

3.5.4.5.4. Excluir

a. O sistema exibe uma lista de fabricantes cadastrados;

b. O administrador faz a busca, escolhe o fabricante a ser excluído na lista e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do fabricante, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o fabricante do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.4.5.4.a. do fluxo principal.

3.5.5. Manter Funcionário

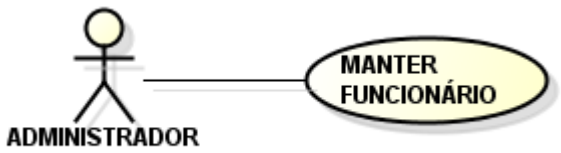


Figura 15: UC 05: Manter Funcionário

3.5.5.1. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os dados, cadastrar, pesquisar, editar e excluir dados dos funcionários.

3.5.5.2. Atores

- Administrador.

3.5.5.3. Pré-condições

- Login do administrador efetuado com sucesso.

3.5.5.4. Evento inicial

- O administrador escolhe alguma opção no menu Fabricantes.

3.5.5.5. Fluxo principal

3.5.5.5.1. Cadastrar

a. Quando o administrador selecionar a opção Novo Funcionário no menu Funcionários, o sistema exibirá um formulário limpo com os campos: nome, CPF, RG, rua, número, bairro, cidade, estado, telefone, celular, e-mail, salário;

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o funcionário foi inserido com sucesso;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.5.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.5.5.1.b..

3.5.5.5.2. Pesquisar

a. O sistema exibe uma lista de funcionários cadastrados;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os funcionários de acordo com o código ou nome que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.5.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter funcionários cadastrados

a. O sistema realiza a busca e verifica que não existem funcionários cadastrados com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.5.5.3. Editar

a. O sistema exibe uma lista de funcionários cadastrados;

b. O administrador faz a busca na lista, escolhe o funcionário a ser alterado e efetua a edição do funcionário; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi atualizado;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.5.5.3.b. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.5.5.3.b.

3.5.5.5.4. Excluir

a. O sistema exibe uma lista de funcionários cadastrados;

b. O administrador faz a busca, escolhe o funcionário a ser excluído na lista e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do

funcionário, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o funcionário do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.5.5.4.a. do fluxo principal.

3.5.6. Realizar Venda

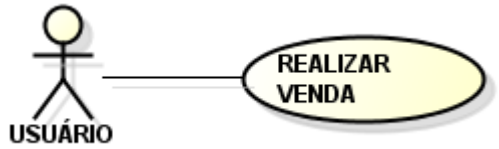


Figura 16: UC 06: Realizar Venda

3.5.6.1. Finalidade / Objetivo

- Possibilitar que o usuário faça suas compras.

3.5.6.2. Atores

- Usuário.

3.5.6.3. Pré-condições

- Login do usuário.

3.5.6.4. Evento inicial

- O usuário escolhe no menu a opção Novo Compra.

3.5.6.5. Fluxo principal

3.5.6.5.1. Adicionar produtos

a. O sistema exibirá uma lista com os produtos que já foram adicionados e através da leitura do QR Code e o usuário forneceu a quantidade que precisará;

b. O usuário fornece as informações solicitadas; (A1) (E1)

c. O sistema atualiza a tela de compras;

Fluxos Alternativos

A1 – Cancela a operação

a. O usuário cancela a operação;

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando ao usuário a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.6.5.1.a..

3.5.6.5.2. Remover

a. O sistema exibe uma lista com os produtos no carrinho que através da opção remover ao selecionar o produto do carrinho; (A1)

b. O sistema atualiza a tela de compras;

c. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.6.5.2.a. do fluxo principal.

3.5.6.5.3. Finalizar compra

a. O sistema exibe um formulário com o dia, valor total, e solicitando que o usuário preenche os campos para finalizar a venda;(A1) (E1)

b. O sistema atualiza o banco de dados e exibe uma mensagem que a venda foi realizada com sucesso;

c. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O usuário cancela a operação;

b. O sistema retorna ao passo 3.5.6.5.3.a. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando ao usuário a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.6.5.3.a..

3.5.7. Manter Saída de Produtos

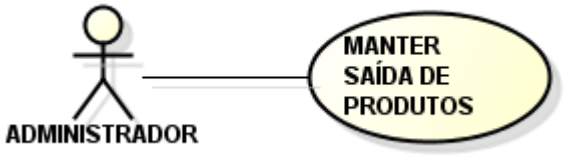


Figura 17: UC 07: Manter Saída de Produtos

3.5.7.1. Finalidade / Objetivo

- Possibilitar que o administrador do sistema possa ver os produtos que já foram lançados para trocar, adicionar mais, editar e excluir o produto lançado errado.

3.5.7.2. Atores

- Administrador.

3.5.7.3. Pré-condições

- Login do administrador efetuado com sucesso.

3.5.7.4. Evento inicial

- O administrador escolhe alguma opção no menu Saída de produtos.

3.5.7.5. Fluxo principal

3.5.7.5.1. Adicionar

a. Quando o administrador selecionar a opção Adicionar Produto no menu Saída de Produtos, o sistema exibirá um formulário limpo com os campos: produto, dia, observação;

b. O administrador fornece as informações solicitadas e confirma a operação; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi inserido com sucesso;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.7.5.1.a..

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que os campos não foram preenchidos de forma correta e exibe uma mensagem alertando o administrador a preencher corretamente os campos;

b. O sistema volta ao passo 3.5.7.5.1.b..

3.5.7.5.2. Pesquisar

a. O sistema exibe uma lista dos produtos lançados;

b. O administrador escreve no campo filtro para solicitar dado que precisa; (A1)

c. O sistema filtra os produtos de acordo com o código ou nome que contenham os dados que estão sendo informados a cada letra digitada; (E1)

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.7.5.2.a. do fluxo principal.

Fluxos de Exceção

E1 – Não ter funcionários cadastrados

a. O sistema realiza a busca e verifica que não existe produto lançado com os dados informados e apresenta a lista vazia com uma mensagem;

3.5.7.5.3. Editar

a. O sistema exibe uma lista de produtos adicionados;

b. O administrador faz a busca na lista, escolhe o produto adicionado a ser alterado e efetua a edição do produto adicionado; (A1) (E1)

c. O sistema atualiza o banco de dados e exibe uma mensagem que o produto foi atualizado;

d. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.7.5.3.b. do fluxo principal.

Fluxos de Exceção

E1 – Campo obrigatório não preenchido corretamente

a. O sistema identifica que algum campo não foi preenchido de acordo com o padrão e exibe uma mensagem alertando o administrador a preencher corretamente o campo;

b. O sistema volta ao passo 3.5.7.5.3.b.

3.5.7.5.4. Excluir

a. O sistema exibe uma lista de funcionários cadastrados;

b. O administrador faz a busca, escolhe o funcionário a ser excluído na lista e confirma a exclusão;

c. O sistema exibi uma mensagem para o administrador com os dados do funcionário, pedindo a confirmação da exclusão; (A1)

d. O administrador confirma a exclusão;

e. O sistema exclui o funcionário do banco de dados;

f. O caso de uso é encerrado.

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema retorna ao passo 3.5.7.5.4.a. do fluxo principal.

3.5.8. Relatório de Cliente

Figura 18: UC 08: Relatório de Cliente

3.5.8.1. Finalidade / Objetivo

- Possibilitar que o administrador tenha melhor visualização dos clientes.

3.5.8.2. Atores

- Administrador.

3.5.8.3. Pré-condições

- Login do administrador efetuado com sucesso e cadastros de cliente feito.

3.5.8.4. Evento inicial

- O administrador escolhe no menu Relatórios a opção Relatório de Clientes.

3.5.8.5. Fluxo principal

3.5.8.5.1. Download do Relatório

a. O sistema exibe uma lista com os dados de todos os clientes;

b. O administrador solicita o download do relatório; (A1)

c. O sistema aceita a solicitação do administrador;

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.8.5.1.a..

3.5.9. Relatório de Clientes que mais compram

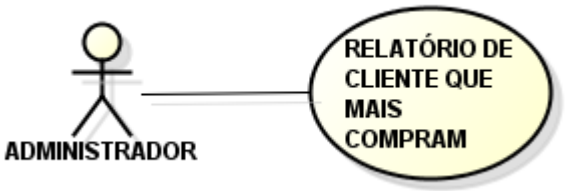


Figura 19: UC 09: Relatório de Clientes que mais compram

3.5.9.1. Finalidade / Objetivo

- Possibilitar que o administrador tenha melhor visualização dos clientes com o total de gastos dele.

3.5.9.2. Atores

- Administrador.

3.5.9.3. Pré-condições

- Login do administrador efetuado com sucesso e vendas realizadas.

3.5.9.4. Evento inicial

- O administrador escolhe no menu Relatórios a opção Relatório de Clientes que mais compram.

3.5.9.5. Fluxo principal

3.5.9.5.1. Download do Relatório

a. O sistema exibe uma lista com os dados de todos os clientes com seus gastos;

b. O administrador solicita o download do relatório; (A1)

c. O sistema aceita a solicitação do administrador;

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.7.5.1.a..

3.5.10. Relatório de Produtos

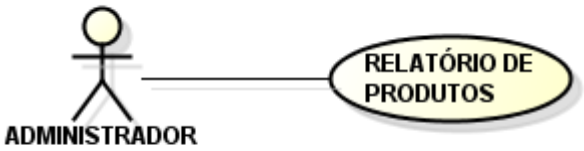


Figura 20: UC 10: Relatório de Produtos

3.5.10.1. Finalidade / Objetivo

- Possibilitar que o administrador tenha melhor visualização dos produtos.

3.5.10.2. Atores

- Administrador.

3.5.10.3. Pré-condições

- Login do administrador efetuado com sucesso e cadastro de produto feito.

3.5.10.4. Evento inicial

- O administrador escolhe no menu Relatórios a opção Relatório de Produtos.

3.5.10.5. Fluxo principal

3.5.10.5.1. Download do Relatório

a. O sistema exibe uma lista com os dados de todos os produtos;

b. O administrador solicita o download do relatório; (A1)

c. O sistema aceita a solicitação do administrador;

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.8.5.1.a..

3.5.11. Relatório de Venda

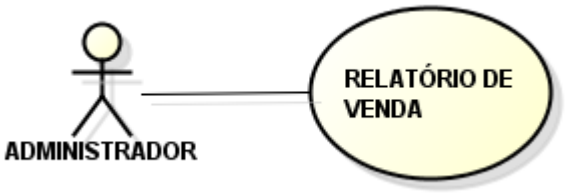


Figura 21: UC 11: Relatório de Venda

3.5.11. Finalidade / Objetivo

- Possibilitar que o administrador tenha melhor visualização das vendas.

3.5.11.2. Atores

- Administrador.

3.5.11.3. Pré-condições

- Login do administrador efetuado com sucesso e vendas realizadas.

3.5.11.4. Evento inicial

- O administrador escolhe no menu Relatórios a opção Relatório de Vendas.

3.5.11.5. Fluxo principal

3.5.11.5.1. Download do Relatório

a. O sistema exibe uma lista com os dados de todas as vendas;

b. O administrador solicita o download do relatório; (A1)

c. O sistema aceita a solicitação do administrador;

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.11.5.1.a..

3.5.12. Relatório de Saída de Produto

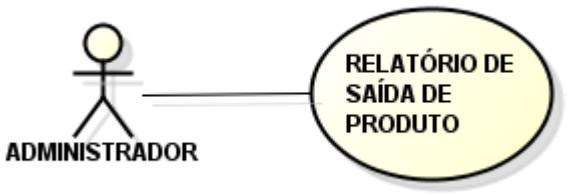


Figura 22: UC 12: Relatório de Saída de Produto

3.5.12.1. Finalidade / Objetivo

- Possibilitar que o administrador tenha melhor visualização da saída de produtos.

3.5.12.2. Atores

- Administrador.

3.5.12.3. Pré-condições

- Login do administrador efetuado com sucesso e produtos adicionados na saída de produtos.

3.5.12.4. Evento inicial

- O administrador escolhe no menu Relatórios a opção Relatório de Saída de Produtos.

3.5.12.5. Fluxo principal

3.5.12.5.1. Download do Relatório

a. O sistema exibe uma lista com os dados de todos os clientes;

b. O administrador solicita o download do relatório; (A1)

c. O sistema aceita a solicitação do administrador;

Fluxos Alternativos

A1 – Cancela a operação

a. O administrador cancela a operação;

b. O sistema volta ao passo 3.5.12.5.1.a..

4. DESENVOLVIMENTO

O trabalho foi separado em três partes: o projeto do Sistema Web para gerenciamento, o projeto do WebService para fazer a comunicação do aplicativo com o Banco de Dados externo e o projeto do aplicativo Android. Na Figura 23 é possível visualizar os três projetos.

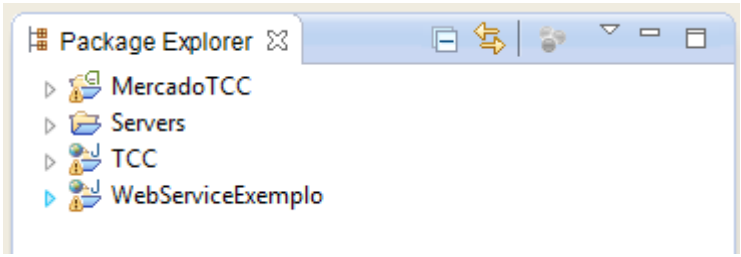


Figura 23: Projetos

4.1. PROJETO DO SISTEMA WEB

O projeto do Sistema Web foi desenvolvido para gerenciar tudo em geral. Fazer cadastros, atualizar algumas informações, salvar novas informações, entre outros procedimentos.

4.1.1. Estrutura

O projeto do Sistema Web tem a seguinte estrutura, conforme a Figura 24.

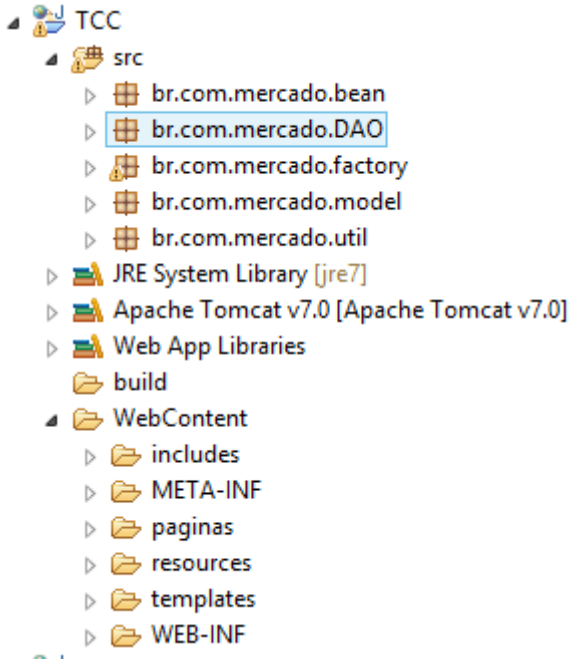


Figura 24: Estrutura do Projeto Web

O pacote *br.com.mercado.bean* possui os métodos para que a página web se comunique com a aplicação e com o banco de dados.

O pacote *br.com.mercado.DAO* possui classes que tem o CRUD.

O pacote *br.com.mercado.factory* possui uma classe que faz a conexão da aplicação com o banco de dados.

O pacote *br.com.mercado.model* possui as classes e seus atributos que vão ser utilizados na aplicação.

O pacote *br.com.mercado.util* possui algumas classes que serão usadas para algumas coisas, como por exemplo: mensagem de erro e conversão de datas para salvar no banco de dados.

A pasta *WebContent* possui todas as páginas e configurações suas respectivas configurações.

4.1.2. WebContent

Para ter acesso ao sistema, é preciso que seja um funcionário. Caso não seja um funcionário ou não tenha os dados corretos, não terá acesso ao sistema ou caso tente entrar em algum *link* que leve a alguma ação dentro do sistema e não estiver feito o login, ele redirecionará para a tela de *login,* que é mostrada na Figura 25.

Figura 25: Página de Login

O usuário seria o CPF do funcionário e a senha a que ele estabeleceu na hora do seu cadastro no sistema.

Feito o *login*, automaticamente será direcionado para a página inicial do sistema. A Figura 26 mostra a página inicial do sistema.

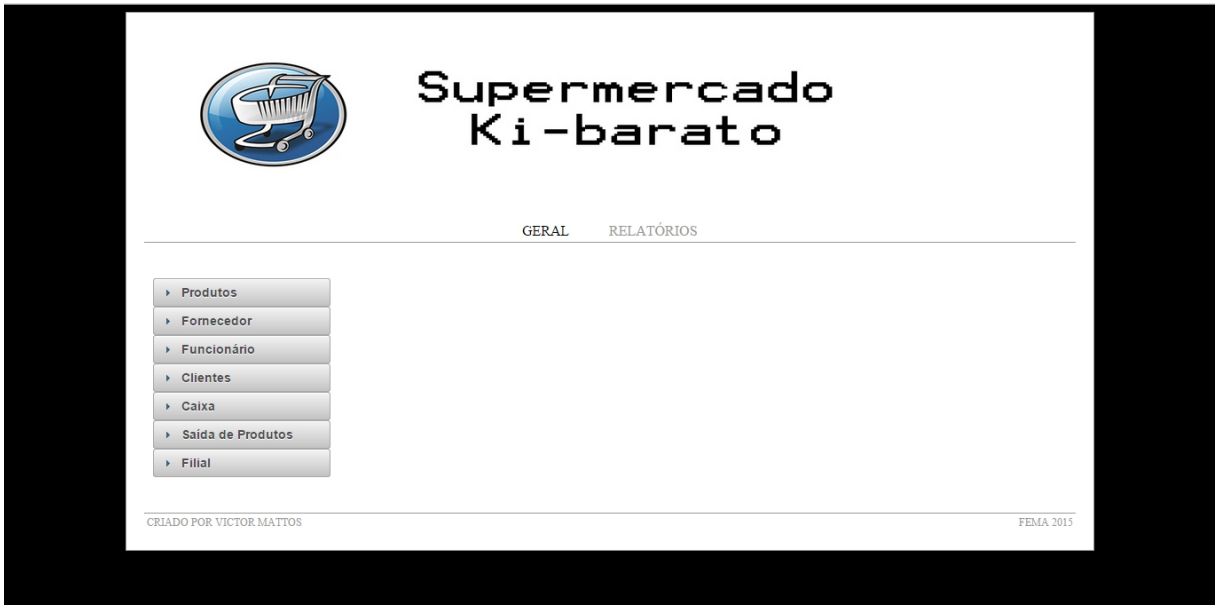


Figura 26: Página Inicial

O sistema foi dividido em duas partes: no gerenciamento e em relatórios.

No gerenciamento é possível fazer o cadastro de uma nova informação, atualizar, remover e listar várias informações. Enquanto que na parte de relatórios, será somente o download do relatório de alguma informação.

4.1.2.1. Template

O *Template* desenvolvido foi usado em todas as páginas do projeto, menos na página inicial. A Figura 27 mostra o código utilizado para o desenvolvimento do *Template* e na Figura 28 mostra um exemplo de CSS utilizado no trabalho.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:p="http://primefaces.org/ui"
    xmlns:ui="http://java.sun.com/jsf/facelets">

<h:head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Sistema Mercado</title>
    <h:outputStylesheet library="css" name="style.css" />
</h:head>
<h:body>
    <p:growl id="msgGlobal" life="6000" />
    <div id="all">
        <h:graphicImage library="images" name="banner.jpg" width="1050"
            height="200" />
        <div id="header">
            <table id="menu">
                <ui:insert name="menu" />
            </table>
        </div>
        <div id="meio">
            <div id="text">
                <ui:insert name="menuladoe" />
            </div>
            <div id="conteudo">
                <ui:insert name="menuladod" />
            </div>
        </div>
        <div id="foot">
            <ui:insert name="rodape" />
        </div>
    </div>
</h:body>
</html>
```

Figura 27: Template

```
#all {
    width: 1050px;
    margin-left: auto;
    margin-right: auto;
    font-family: futura;
    background-color: white;
    padding: 20px;
    border: 1px solid #999999;
}
```

Figura 28: Exemplo de CSS

A formatação do *Template* foi feita toda no CSS, alinhando os componentes em seus lugares. Cada *div* representa uma formatação que será feita. O *insert* representa que algum componente vai ser implementado para ocupar o seu espaço, conforme mostra a Figura 29.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:p="http://primefaces.org/ui"
    template="/templates/modeloPrincipal.xhtml">


    <ui:define name="menu">
        <ui:include src="/includes/menuGeral.xhtml" />
    </ui:define>

    <ui:define name="menuladoe">
        <ui:include src="/includes/itensGeral.xhtml" />
    </ui:define>

    <ui:define name="conteudo">
    </ui:define>

    <ui:define name="rodape">
        <ui:include src="/includes/rodape.xhtml" />
    </ui:define>

</ui:composition>
```

Figura 29: Exemplo do insert

O *Template* oferece uma grande vantagem. Quando precisa-se alterar algo que é utilizado em todas as páginas, como por exemplo o menu, é preciso alterar somente o menu que foi feito fora da página e não alterar página por página.

4.1.2.2. Páginas

As páginas foram desenvolvidas no mesmo modelo. Campos obrigatórios possuem * indicando que ele é obrigatório e caso esteja algo errado ele acusará erro. Na Figura 23 mostra a página de cadastro de produto.

Figura 30: Página de cadastro do produto

No cadastro do produto, todos os campos são obrigatório, caso tente gravar sem preencher nada, será gerado um erro e em uma mensagem no canto superior da tela será especificado o erro. É possível visualizar esse procedimento na Figura 31.

Figura 31: Erro gerado

Além de gerar a mensagem de erro, também é possível notar que ele marca o

campo que está o erro.

Nos campos obrigatórios, foi desenvolvido um laço, que diz que um pertence ao outro.

```
<p:outputLabel for="txtDescProdNovo" value="Descrição:" />
<p:inputText id="txtDescProdNovo"
    value="#{MBProduto.produto.descricao}" maxlength="50" size="30"
    required="true" requiredMessage="O Campo Descrição é Obrigatório." />
```

Figura 32: Laço dos campos

É possível visualizar na Figura 32, na propriedade *value* não possui um * dizendo que o campo é obrigatório, e sim um laço feito entre os dois campos, feito pelo *for* do *outputLabel* e pelo *id* do *inputText.*

Para fazer alterações no produto, ver mais detalhes sobre o produto ou excluir algum produto, é preciso listá-lo.

Figura 33: Lista de produtos.

A lista está ordenada pela descrição do produto, é possível somente visualizar qual o seu código, qual a sua descrição e quem foi o fornecedor, conforme mostra a Figura

33.

Esta lista pode ser ordenada pelo código ou pela descrição e também pode filtrar pelo código ou pela descrição do produto.

Para ordenar ele na ordem crescente ou decrescente, basta clicar no Código ou na Descrição, que ordenará primeiro na ordem crescente e depois na ordem decrescente.

Para filtrar algum item, digite o código que precisa filtrar no campo em baixo do Código. Caso queira filtrar pelo campo Descrição, basta escrever no campo em da Descrição.

É possível realizar três ações: ver informações detalhadas do produto, atualizar o produto ou remover ele do sistema.

Para visualizar as informações detalhadas do produto, basta clicar no mais e ele abrirá uma nova janela e abrirá uma tela, conforme a Figura 34.

Figura 34: Informações detalhadas do produto.

Com a janela aberta, não é possível manipular nada, apenas ver as suas informações e cancelar a operação.

Para editar as informações do produto, basta clicar no lápis e ele abrirá uma nova janela, conforme a Figura 35.

Figura 35: Editar informações do produto.

Com a janela aberta, não é possível manipular nada atrás. Alguns dados não podem ser alterados, como o código e a sua quantidade, mas as outras informações podem ser alteradas. Para salvar as informações que foram alteradas, basta apertar o botão Editar, caso contrário, queira cancelar a operação, basta clicar no botão Cancelar. Para excluir o produto, basta clicar na lixeira e ele abrirá uma nova janela, conforme a Figura 36.

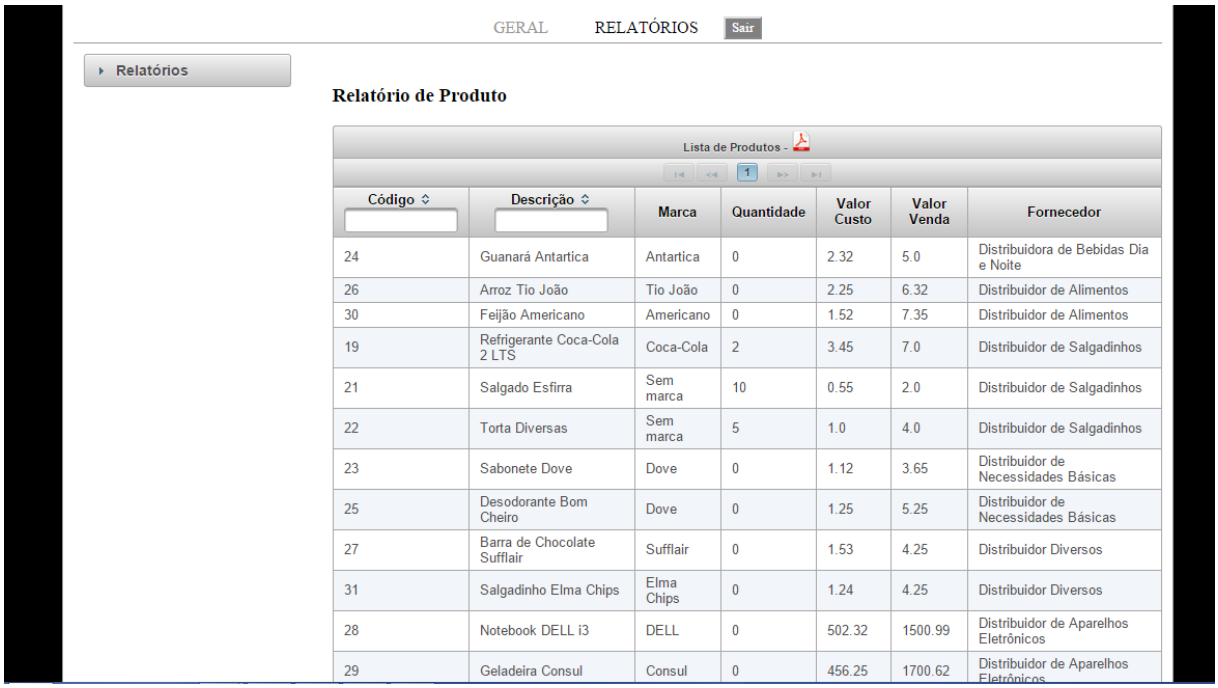
Figura 36: Exclusão do produto.

Com a janela aberta, não é possível manipular nada atrás. Essa janela é gerado para o que o usuário veja outras informações e tenha certeza que está excluindo o produto certo. Caso seja o produto certo para exclusão, basta clicar no botão Excluir, caso contrário, para cancelar a operação basta clicar no botão Cancelar.

Para gerar um relatório, é preciso ir na área onde estão os relatórios. Para ir na área de relatório, basta clicar no menu Relatório do lado do menu Geral, como mostra na Figura 37.

Figura 37: Página inicial dos relatórios.

Em seguida, aparecerá a página inicial do menu relatório, que será igual à Figura 38. Agora para gerar um relatório, é preciso primeiro selecionar qual o tipo de item que precisa gerar o relatório.



Código	Descrição	Marca	Quantidade	Valor Custo	Valor Venda	Fornecedor
24	Guanará Antartica	Antartica	0	2.32	5.0	Distribuidora de Bebidas Dia e Noite
26	Arroz Tio João	Tio João	0	2.25	6.32	Distribuidor de Alimentos
30	Feijão Americano	Americano	0	1.52	7.35	Distribuidor de Alimentos
19	Refrigerante Coca-Cola 2 LTS	Coca-Cola	2	3.45	7.0	Distribuidor de Salgadinhos
21	Salgado Esfirra	Sem marca	10	0.55	2.0	Distribuidor de Salgadinhos
22	Torta Diversas	Sem marca	5	1.0	4.0	Distribuidor de Salgadinhos
23	Sabonete Dove	Dove	0	1.12	3.65	Distribuidor de Necessidades Básicas
25	Desodorante Bom Cheiro	Dove	0	1.25	5.25	Distribuidor de Necessidades Básicas
27	Barra de Chocolate Sufflair	Sufflair	0	1.53	4.25	Distribuidor Diversos
31	Salgadinho Elma Chips	Elma Chips	0	1.24	4.25	Distribuidor Diversos
28	Notebook DELL i3	DELL	0	502.32	1500.99	Distribuidor de Aparelhos Eletrônicos
29	Geladeira Consul	Consul	0	456.25	1700.62	Distribuidor de Aparelhos Eletrônicos

Figura 38: Relatório de produto.

É possível visualizar que possui diferentes tipos de relatórios que podem ser geral no menu esquerdo.

O exemplo da Figura 38 mostra alguns dados antes da confirmação para gerar o relatório. Caso esteja tudo certo, basta pressionar o botão embaixo da lista dos produtos, para que seja gerado e relatório e faça o download do arquivo em formato *pdf*. Para ter certeza que o cursor está encima do botão, é somente analisar a cor do botão, caso esteja cinza o cursor não está encima do botão, mas caso esteja preto, o cursor está encima do botão.

4.2. PROJETO DO WEB SERVICE

O projeto do *Web Service* foi necessário ser desenvolvido, pois, o Android não faz comunicação com banco de dados externo, ele somente faz a comunicação com o banco de dados interno. Então, com a necessidade do aplicativo fazer a comunicação com o banco de dados MySQL, esse projeto foi desenvolvido.

4.2.1. Estrutura

O projeto do Web Service tem a seguinte estrutura, conforme a Figura 39.

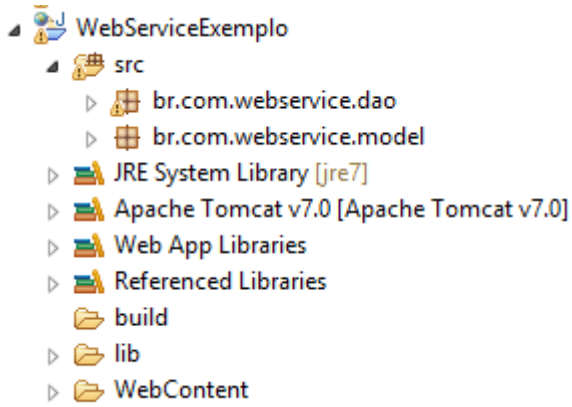


Figura 39: Estrutura do projeto Web Service

O pacote *br.com.webservice.dao* possui uma classe para a conexão da aplicação com o banco de dados e as outras classes que possuem o CRUD.

O pacote *br.com.webservice.model* possui as classes que o aplicativo Android vai utilizar.

4.2.2. Classes Web Service

O projeto utilizou a ferramenta Axis2 para que as classes gerem um WSDL.

A extração do WSDL pode ser feita usando o localhost ou com o IP da máquina que está o servidor. Na Figura 40 mostra o WSDL gerado usando o localhost e na Figura 41 mostra o WSDL gerado com o IP da máquina.

Figura 40: Exemplo do WSDL gerado da classe ClienteDAO com localhost

Figura 41: Exemplo do WSDL gerado da classe ClienteDAO com o IP da máquina

O WSDL é o mesmo, a única coisa que mudou foi o tipo de acesso ao WSDL. Como o servidor está hospedado em máquina pessoal, precisa-se mudar o IP quando for trocar de rede. Na Figura 42 mostra qual o exemplo de como obter o IP da máquina.

Figura 42: Exemplo de IP da rede local

A comunicação Web Service com o aplicativo Android é feita através do WSDL que foi gerado. Mas para que possa ter essa comunicação, precisa-se de um protocolo de comunicação, e o protocolo usado para fazer a comunicação, foi o protocolo SOAP.

Para fazer testes e ver se o SOAP reconhece o WSDL gerado pela ferramenta Axis2, basta salvar o WSDL e abrir um novo projeto na ferramenta SoapUI. Depois de feitos esses passos, ela mostrará todos os métodos que foram feitas na classe, conforme mostra a Figura 43.

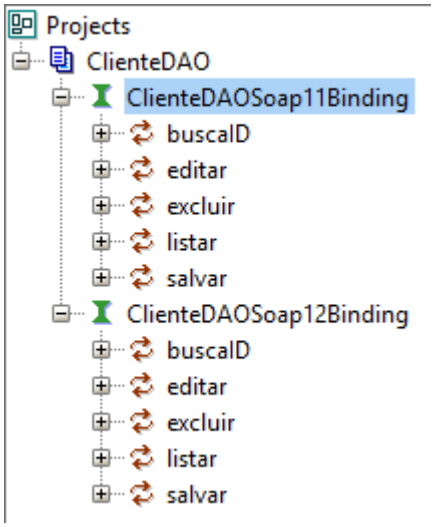


Figura 43: Métodos da classe ClienteDAO

Para visualizar e ver se os métodos funcionam corretamente, basta abrir um método e preencher os campos de acordo com o método solicitado. Na Figura 44 mostra a solicitação do método listar.

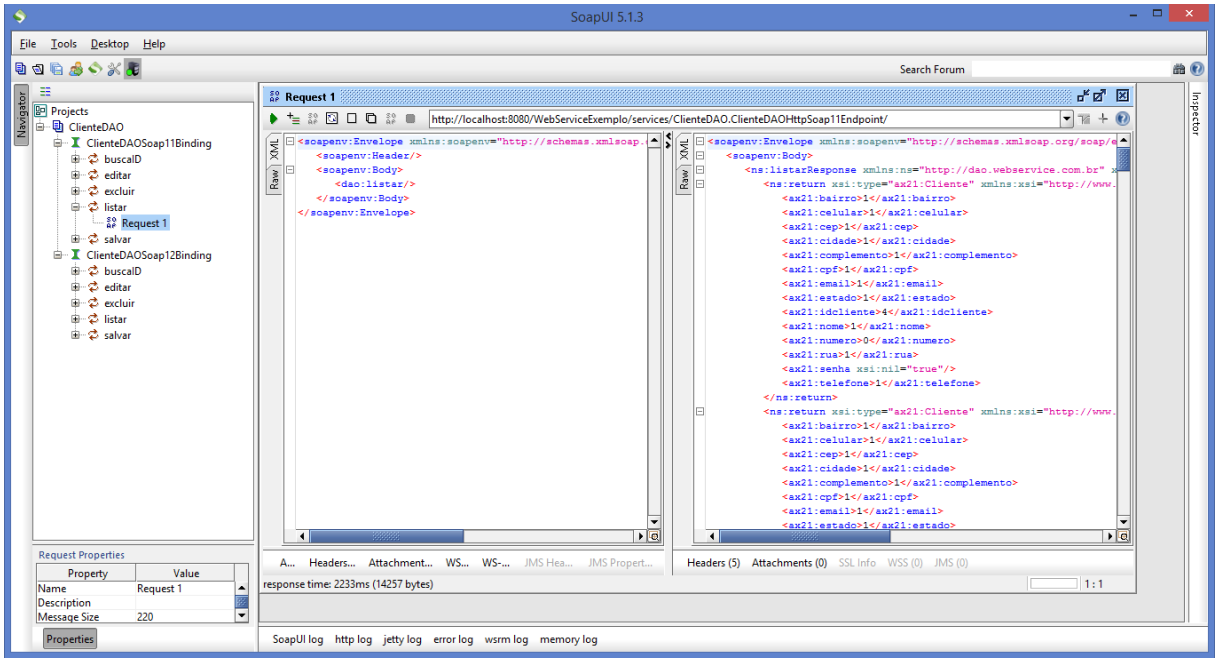


Figura 44: Exemplo de solicitação na classe ClienteDAO

O retorno da requisição solicitada será um XML.

4.3. PROJETO DO APLICATIVO ANDROID

O aplicativo Android foi desenvolvido para que o usuário realize a venda. O usuário que não possui cadastro no sistema, ele pode fazer o cadastro dos seus dados no sistema pelo aplicativo.

4.3.5. Estrutura

O aplicativo Android tem a seguinte estrutura, conforme a Figura 45.

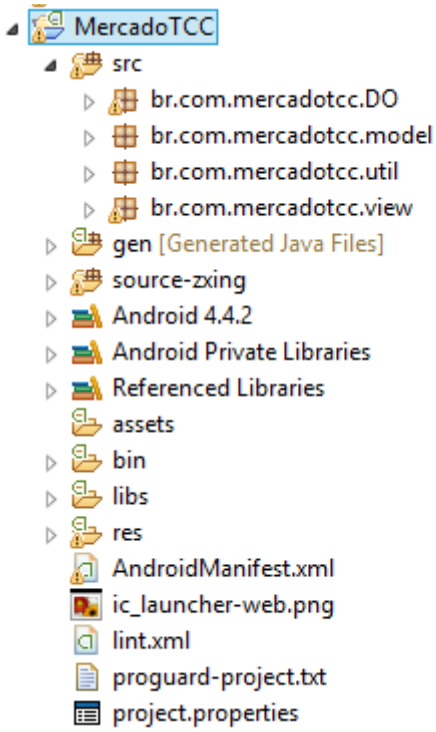


Figura 45: Estrutura do aplicativo Android

O pacote *br.com.mercadotcc.DO* possui as classes com seus métodos para fazer a comunicação com o banco de dados.

O pacote *br.com.mercadotcc.model* possui as classes com seus atributos que o aplicativo vai utilizar.

O pacote *br.com.mercadotcc.util* possui uma classe que vai validar se o campo é obrigatório e se ele está em branco.

O pacote *br.com.mercadotcc.view* possui as classes que vão implementar as telas utilizadas no sistema.

A pasta *gen* possui todas as propriedades da aplicação.

A pasta *source-zxing* é a pasta onde está localizada toda a aplicação do projeto *Zxing*.

A pasta *lib* é a pasta onde está localizada todas as *jars* utilizada no aplicativo.

A pasta res é a pasta onde está localizada as telas, imagens, *strings*, entre outras coisas.

O arquivo *AndroidManifest* possui todas as configurações do projeto.

4.3.2. Comunicação Externa

Como o Android não faz comunicação com banco de dados externo, foi preciso o desenvolvimento de um Web Service para fazer essa comunicação. O protocolo utilizado para fazer a comunicação da aplicação com o banco de dados externo, foi o protocolo SOAP.

Diferente de aplicações que já fazer comunicação com o banco de dados, o SOAP ele faz comunicação com o banco de dados através do protocolo HTTP que primeiro estabelece uma comunicação com o banco de dados, e depois de estabelecido esta conexão, ele envia uma mensagem solicitando alguma requisição, a resposta pode conter erros ou pode ter sucesso. Na Figura 46 mostra a comunicação.

```
1  package br.com.mercadotcc.DO;
2
3  import org.ksoap2.SoapEnvelope;
8
9  public class LoginDO {
10     private static final String URL = "http://192.168.0.103:8080/WebServiceExemplo/services/LoginDAO?wsdl";
11     private static final String NAMESPACE = "http://dao.webservice.com.br";
12
13     private static final String BUSCAR = "validate";
14
15     public boolean validate(String cpf, String senha) {
16         SoapObject buscar = new SoapObject(NAMESPACE, BUSCAR);
17         buscar.addProperty("cpf", cpf);
18         buscar.addProperty("senha", senha);
19
20         SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(
21                 SoapEnvelope.VER11);
22
23         envelope.setOutputSoapObject(buscar);
24
25         envelope.implicitTypes = true;
26
27         HttpTransportSE http = new HttpTransportSE(URL);
28         try {
29             http.call("urn:" + BUSCAR, envelope);
30             SoapPrimitive resposta = (SoapPrimitive) envelope.getResponse();
31             return Boolean.parseBoolean(resposta.toString());
32
33         } catch (Exception e) {
34             e.printStackTrace();
35             return false;
36         }
37     }
38 }
39
```

Figura 46: Requisição da classe LoginDO.

É possível analisar na Figura 33, que ela usa objetos e manda pacotes de dados para que tenha a comunicação e aguarda uma resposta de verdadeiro ou falso.

4.3.3. Telas

A tela inicial da aplicação Android possui os campos login, que é o CPF do cliente, e a senha que foi estabelecida no momento do seu cadastro. Caso o cliente não

possua cadastro, o aplicativo possui um *link* que vai direcionar o cliente para uma outra tela, para que o mesmo possa fazer seu cadastro no sistema. Além disso, a aplicação possui três campos, um para ter acesso ao sistema depois de preenchido o *login* e a senha, outro para limpar os campos de *login* e senha e outro para sair da aplicação. A Figura 47 mostra a tela inicial do aplicativo no Android.

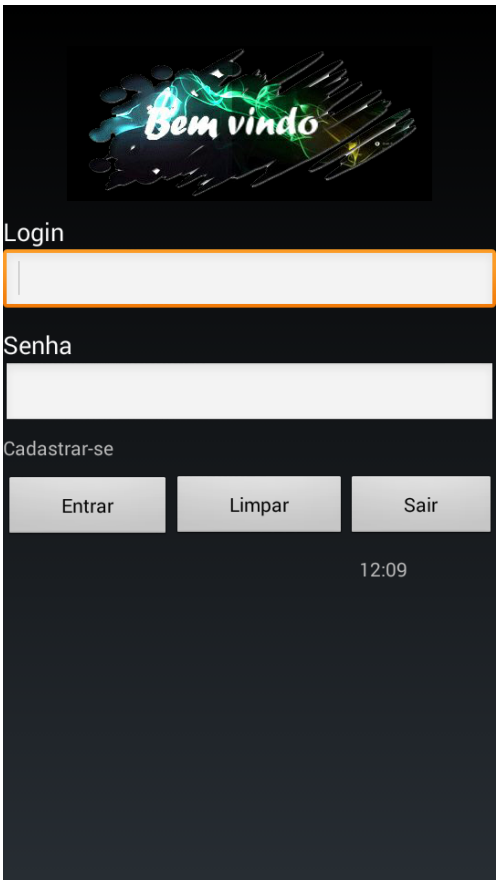


Figura 47: Tela inicial do aplicativo.

Pode-se notar na Figura 47 que possui um item chamado Cadastrar-se. Esse item levará o cliente para outra tela para que possa fazer seu cadastro no sistema e ter acesso a ele.

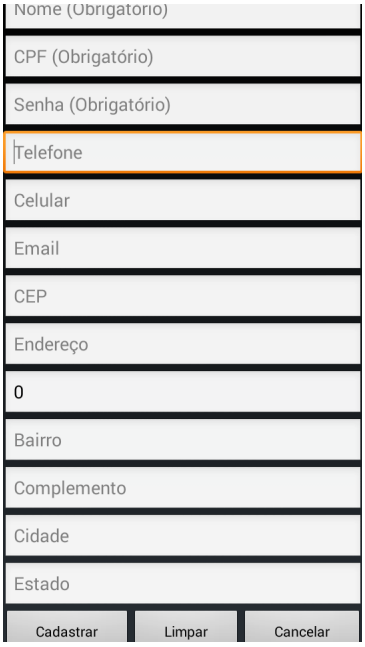


Figura 48: Tela de cadastro do cliente.

Conforme a Figura 48, o cadastro do cliente, possui somente três campos obrigatórios, o restante dos campos são opcionais. Caso o cliente na hora do cadastro não preencha esses campos obrigatórios, será mostrado um erro que os campos estão em branco e com isso, não foi possível realizar o cadastro. Na Figura 49 mostra o erro gerado.

Figura 49: Erro de campos em branco.

Na tela inicial do aplicativo, caso tente entrar sem preencher os campos, o aplicativo retornará um erro que os campos estão em branco. Na Figura 50 mostra o erro gerado.

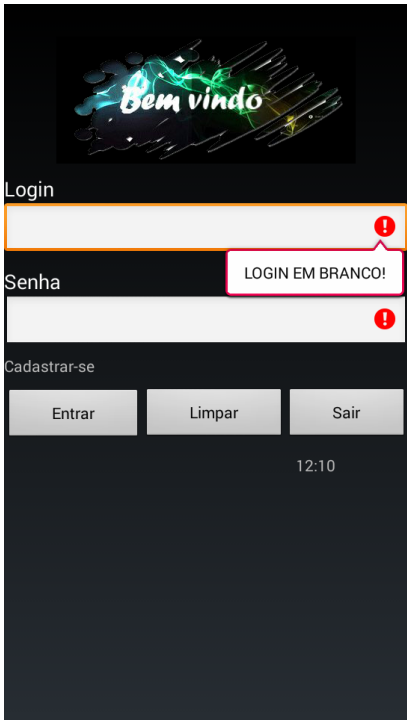


Figura 50: Campos em branco na tela de login.

Caso tente acessar o sistema, sem cadastro ou sem informações validas, o aplicativo retornará um erro informando que os dados estão errados. Na Figura 51 mostra o erro gerado.

Figura 51: Dados incorretos na tela inicial.

Quando tiver preenchido os campos corretamente, ele redirecionará a uma tela que é possível realizar três ações: realizar uma nova compra, consultar a lista de produtos cadastros ou sair do sistema. A Figura 52 apresenta as três opções que o usuário tem acesso.

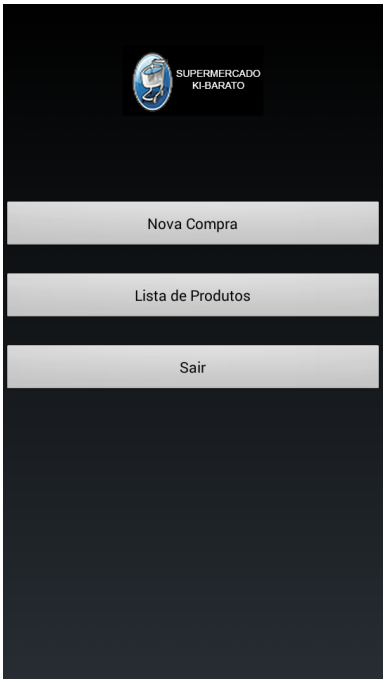


Figura 52: Opções dentro da aplicação.

Selecionando a primeira opção, ele abrirá uma nova tela, que mostrará todos os produtos adicionados no seu carrinho, como é uma nova compra, a lista aparecerá vazio, conforme a Figura 53.

Figura 53: Tela de nova compra.

O botão de novo produto quando pressionado, levará o cliente para outra tela para que o usuário possa *scannear* o código do produto e adicionar o produto na lista de compra, conforme a Figura 54.

Figura 54: Câmera ativada para scannear o código.

Para que o código seja escaneado corretamente, e preciso que o cliente centralize o código no meio da câmera, e depois de *scanneado*, o produto será automaticamente adicionado na lista de compras, conforme a Figura 55.

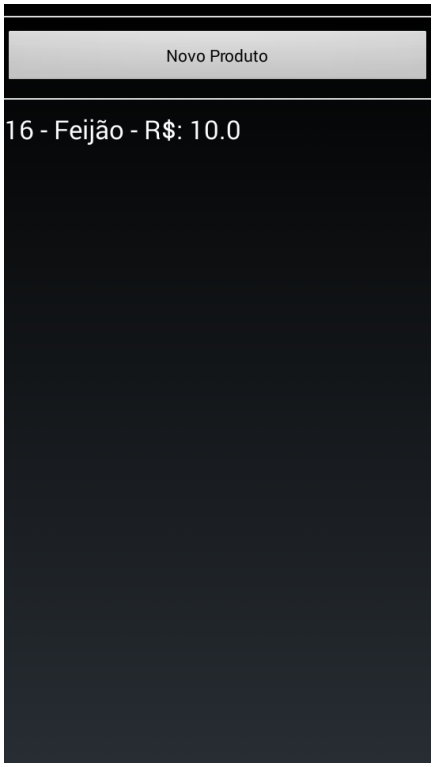


Figura 55: Produto na lista de compra.

A lista de produtos da compra mostrará os dados do produto, como o seu código, a sua descrição e qual o seu valor. Na hora de finalizar a compra, terá uma opção que levará o usuário para outra tela, mostrando o valor total e para ele informar o seu cartão para finalizar a compra.

Outra opção no menu inicial, é para consultar os produtos cadastrados no sistema. Essa opção serve para consultar os produtos antes de realizar as compras, conforme mostra a Figura 56.

Figura 56: Lista de produtos cadastrados no sistema.

Depois de consultar os produtos, basta pressionar o botão voltar para voltar no menu inicial.

Para sair do sistema, basta pressionar o botão Sair do menu inicial.

5. CONCLUSÃO

Com o desenvolvimento do sistema Web para gerenciamento e do aplicativo Android, o empresário terá menos gasto com funcionários e equipamentos, já que o aplicativo é instalado diretamente do celular do cliente. Outro grande funcionalidade do aplicativo é que o cliente pode fazer seu cadastro diretamente pelo aplicativo, evitando grandes esperas em filas. Além disso, o aplicativo Android também pode servir para consultar o valor dos produtos, já que alguns produtos não tem o valor na etiqueta. Como algumas coisas ficam difíceis de serem gerenciadas pelo celular, como cadastros e relatórios, foi importante o desenvolvimento da aplicação Web para fazer o gerenciamento.

Este trabalhou possibilitou grande conhecimento no seu desenvolvimento, por ser o primeiro aplicativo Android com Web Service e QR Code. A tecnologia QR Code mostrou que um código de barras é amplos, podendo armazenar vários tipos de dados e mostrou como os códigos barras não são padrões e cada um tem o seu propósito. Por ser meu primeiro aplicativo Android e com o Web Service, tive muita dificuldade, pois, o desenvolvimento comparado ao desenvolvimento Web é diferente. Nunca tinha desenvolvido algo com Web Service e está foi minha maior dificuldade, porque não saber como o Web Service funciona.

Por não ser um software encomendado por alguma empresa, pode-se dizer que estás aplicações não atenderam a todos os requisitos de várias empresas. Caso fosse para alguma empresa, um levantamento de requisitos seria realizado para saber quais as necessidades que as aplicações precisavam atender.

O esperado com o desenvolvimento deste trabalho é a obtenção de conhecimento com o conhecimento de novas ferramentas, como a tecnologia QR Code que pode ser usada de várias maneiras. Outro objetivo é que possa ajudar em outras pesquisas, pois foi utilizado diversas ferramentas no desenvolvimento deste trabalho.

Acredita-se que este software possa ganhar força no futuro, e diminuir custos de grandes empresas, já que não se fará necessário a contratação de alguns funcionários e o custo mensal será bem menor.

REFERÊNCIAS

BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML.** Segunda Edição. Editora Campus/Elsevier.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML – Guia do administrador,** Segunda Edição. Addison-Wesley. Elsevier Editora Ltda, 2006.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **Unified Modeling Language User Guide**, Segunda Edição, Addison-Wesley Object Technology Series, 2005.

CATAPAN, Danilo Rodrigues. **Desenvolvimento de aplicativo para Android SDK.** 2009, 50p. Trabalho de Conclusão de Curso(Ciência da Computação) - Fundação Educacional do Município de Assis – FEMA/Instituto Municipal de Ensino Superior de Assis – IMESA.

DANTAS, Daniel Chaves Toscano. **Simple Object Access Protocol (SOAP).** Disponível em <http://www.gta.ufrj.br/grad/07_2/daniel/>. Acesso em: 06 ago. 2015.

DEITEL, Paul; DEITEL, Harvey. **Java – Como Programar,** Oitava Edição. Edson Furmankiewicz, São Paulo: Pearson Prentice Hall, 2010.

Desenvolvimento para Android Tutorial Para Iniciantes Introdução. Disponível em <http://euprogramando.com.br/desenvolvimento-android-tutorial-iniciantes/>. Acesso em: 16 mar. 2015.

Eduardo. Lendo código de barras na sua aplicação para Android. Disponível em <http://www.oslunaticos.com.br/2012/05/lendo-codigo-de-barras-na-sua-aplicacao-para-android/#more-1845>. Acesso em: 06 ago. 2015.

GAMES, William. **Visão Geral sobre PrimeFaces.** Disponível em <https://williamgamers.wordpress.com/2012/06/04/visao-geral-sobre-primefaces/>. Acesso em: 07 ago. 2015.

GARCIA, Eduardo Giroto. ***COMPAREOFFER*: APLICATIVO ANDROID E SOFTWARE WEB PARA OBTENÇÃO E COMPARAÇÃO DE PREÇOS PARA PRODUTOS DE SUPERMERCADOS.** 2012. 91p. Trabalho de Conclusão de Curso(Ciência da Computação) -Fundação Educacional do Município de Assis – FEMA/Instituto Municipal de Ensino Superior de Assis - IMESA.

LUDERS, Lasse. **Dicas profissionais para o uso de QR-Codes: Entrevista com Lasse Lüders.** Disponível em <http://br.jimdo.com/2012/03/08/dicas-profissionais-

para-o-uso-de-qr-codes-entrevista-com-lasse-l%C3%BCders/>. Acesso em: 20 out. 2014.

MACHADO, Guilherme. **QR-Code 5 dicas para usar a ferramenta com sucesso no mercado imobiliário**. Disponível em <http://www.redimob.com.br/post/e4a65972-7f6e-4b38-8701-79187d535f4d/qr-code-5-dicas-para-usar-a-ferramenta-com-sucesso-no-mercado-imobiliario>. Acesso em: 20 out. 2014.

MACORATTI, José Carlos. **Modelando Sistemas em UML - Casos de Uso.** Disponível em <http://www.macoratti.net/net_uml2.htm>. Acesso em: 16 mar. 2015.

MENDES, Douglas Rocha. **Programação Java com Ênfase em Orientação a Objetos.** Novatec.

NETBEANS. **Trilha do Aprendizado do Java EE e Java Web**. Disponível em <https://netbeans.org/kb/trails/java-ee_pt_BR.html>. Acesso em: 06 ago. 2015.

PACIEVITCH, Yuri. **História do Java**. Disponível em <http://www.infoescola.com/informatica/historia-do-java>. Acesso em: 21 jun. 2015.

PANKIEWICSZ, Igor. **O que são os QR-Codes?.** Disponível em <http://www.tecmundo.com.br/imagem/1995-o-que-sao-os-qr-codes-.htm>. Acesso em: 20 out. 2014.

PEREIRA, Ana Paula. **O que é XML?.** Disponível em <http://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm>. Acesso em: 07 ago. 2015.

QUARESMA, Camila Gondim. **DESENVOLVIMENTO DE APLICATIVO WEB MARKET SYSTEM.** 2011. 56p. Trabalho de Conclusão de Curso(Ciência da Computação) - Fundação Educacional do Município de Assis – FEMA/Instituto Municipal de Ensino Superior de Assis – IMESA.

QUEIRÓS, Ricardo. **Android - Introdução ao Desenvolvimento de Aplicações.** FCA – Editora de Informática, Lda.

RABELLO, Ramon Ribeiro. **Android: um novo paradigma ao desenvolvimento móvel.**

SOON, Tan Jin. **QR Code.**

SOUZA, Lucas Henrique Ronqui de. **Desenvolvimento de Aplicativo Educativo para Plataforma Android.** 2012. 110p. Trabalho de Conclusão de Curso(Ciência da

Computação) - Fundação Educacional do Município de Assis – FEMA/Instituto Municipal de Ensino Superior de Assis – IMESA.

VERONESE, Gustavo; CORREA, Alexandre; WERNER, Cláudia; NETTO, Felipe Jezini. **ARES: Uma Ferramenta de Engenharia Reversa Java-UML.** Simpósio Brasileiro de Engenharia de Software, 16, 2002. Gramado, Brasil.

WIKIPÉDIA. **Código QR.** Disponível em <http://pt.wikipedia.org/wiki/C%C3%B3digo_QR>. Acesso em: 20 out. 2014.

WIKIPÉDIA. **Apache Tomcat.** Disponível em <https://pt.wikipedia.org/wiki/Apache_Tomcat>. Acesso em: 6 ago. 2015.

WIKIPÉDIA. **Apache Axis.** Disponível em <https://pt.wikipedia.org/wiki/Apache_Axis>. Acesso em: 7 ago. 2015.