Gerenciamento de Estoque e Vendas (GEV): sistema de informação para comércio de pequeno e médio porte integrando funcionalidades entre um *software desktop* e um aplicativo para dispositivo *Android*

Felipe Otávio Mendes de Oliveira[1], Michele Cristiani Barion[1], Daniela Marques[1]

[1] Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) – Campus Hortolândia – São Paulo – SP – Brasil

felipeotaviomendes@gmail.com, michele_barion@hotmail.com, ifsp.danielal@gmail.com

Abstract. *Through quantitative and qualitative surveys, it can be observed that sellers of clothing and shoe stores spend a lot of time in sales, while waiting in search of the product chosen by the customer, besides the possibility of not having it available in stock. Knowing that, this project consists in the creation of a system composed of two applications designed to manage stock information and product sales in small and medium-sized businesses. Among these applications, we will have a mobile software for consultation of the products requested in real time by the customers, also with reservation functionality and product sales, as well as a desktop application for product and sales management, as well as management reports for managers and sellers. As a final result, it aims at greater efficiency in the sales process and product queries, thus providing greater agility in sales.*

Resumo. Por meio de pesquisas quantitativa e qualitativa, observa-se que vendedores de lojas do segmento de vestuários e sapatos gastam muito tempo nas vendas, considerando a espera na busca do produto escolhido pelo cliente, além da possibilidade de não tê-lo disponível no estoque. Diante destes fatos, este projeto consiste na criação de um sistema composto por duas aplicações destinadas a administrar informações de estoque e vendas de produtos em comércios de pequeno e médio porte. Dentre essas aplicações, teremos um *software mobile* destinado a consultas dos produtos solicitados em tempo real pelos clientes, contando também com as funcionalidades de reserva e venda de produtos, e também uma aplicação *desktop* para gerenciamento de produtos e vendas, além de relatórios gerenciais para gerentes e vendedores. Como resultado final, objetiva-se uma maior eficiência no processo de vendas e consultas de produtos, proporcionando, assim, uma maior agilidade nas vendas.

1. Introdução

Com o grande número de produtos expostos em lojas de sapatos ou vestuários disponibilizados para venda, considerando diferentes tipos, tamanhos, marcas e/ou cores, vendedores precisam buscar as escolhas diante das diversas variáveis, visitando o estoque sempre que ocorrer uma nova solicitação pelo cliente. Observa-se, assim, que vendedores gastam muito tempo neste processo de busca e *feedback* diante da disponibilidade do pedido.

Segundo Walker (1991), há dez mandamentos do bom atendimento, sendo que dois deles estão associados à proposta deste trabalho:

1. Atendimento imediato: não deixar o cliente esperando e, consequentemente, agir com rapidez. “Por favor, aguarde que vou atendê-lo (a) em seguida...” é uma expressão a ser usada pedindo paciência ao cliente.

2. Criar e sugerir soluções: buscar com a equipe soluções criativas para melhorar o atendimento. “Lembre-se: você é o ouvido da empresa”.

Considerando aplicações que já existem para este fim, muitas são disponibilizadas gratuitamente para instalação ou disponibilizam parte das funções gratuitas e outros recursos a partir de planos que deverão ser pagos. Comparando a proposta deste trabalho com uma aplicação produzida, o sistema NEX (PROGRAMANEX) é uma ferramenta que objetiva a gestão comercial de uma loja, porém, se limita por não conter funcionalidades voltadas à otimização do processo de vendas.

Para ter um embasamento diante da visão de clientes e de vendedores, considerando a necessidade de se ter uma aplicação para agilizar o atendimento quanto à busca do produto escolhido, foi desenvolvida uma pesquisa de caráter quantitativa e qualitativa para levantamento de dados e, assim, justificar o desenvolvimento dessa proposta. Foi usada a ferramenta *Google Forms*, para registro das informações.

As figuras 1 e 2 mostram os resultados de dois questionários, tendo um a participação de 49 usuários na visão clientes que frequentam ou já frequentaram lojas físicas de segmento vestuário e/ou sapato, e o segundo com 6 usuários que trabalham ou já trabalharam como vendedores em comércios.

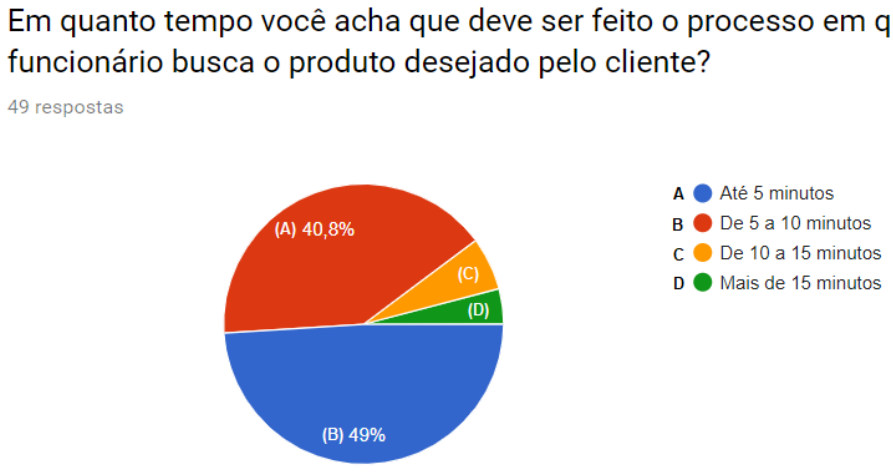

Figura 1. Visão cliente: tempo gasto na busca de produtos

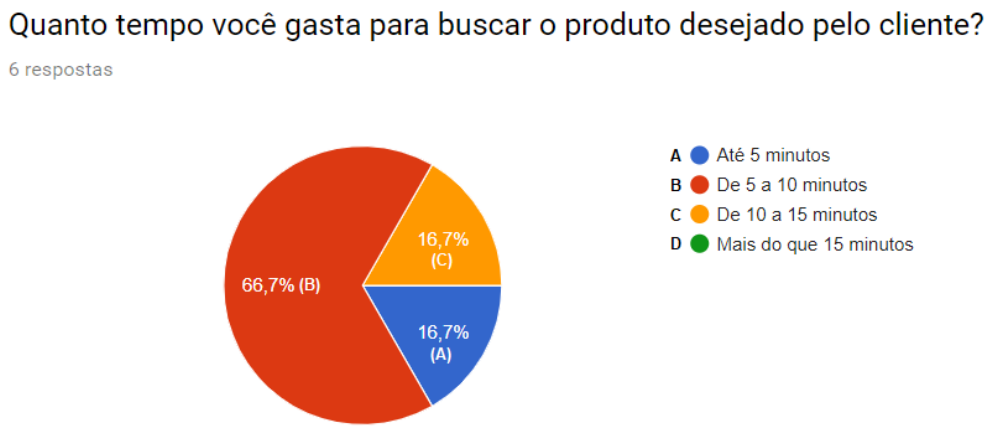

Figura 2. Visão vendedor: tempo gasto na busca de produtos

A figura 1 mostra que aproximadamente 50% dos clientes consumidores desejam uma resposta de seus pedidos até 5 minutos. Em contrapartida, a figura 2 mostra que 66,7% dos vendedores atendem de 5 a 10 minutos e ainda, pode ser considerado que muitas vezes não um determinado produto escolhido não seja encontrado, fazendo com que o vendedor retorne todo o processo com uma nova escolha por parte do cliente ou sugira outros produtos, mas sem a certeza que as propostas serão satisfatórias na visão do consumidor.

Diante destes fatos, este projeto consiste no desenvolvimento de uma aplicação *desktop* para gerenciamento de vendas e estoque e um módulo para dispositivo móvel a ser disponibilizado ao vendedor, no qual poderá fazer buscas no estoque juntamente com o cliente, seja a partir de opções solicitadas ou caso haja falta da escolha, poderá optar por sugestões que serão disponibilizadas a partir de produtos semelhantes. “Muitas ações para melhorar a qualidade no atendimento são simples e fáceis de aplicar, mas muitas organizações ainda deixam de lado” (BARBOSA, T.; TRIGO, A.; SANTANA, L., 2015). Sendo assim, como resultado final desta proposta, espera-se a otimização e a agilidade no processo de busca e venda de produtos.

Para abordagem deste trabalho, este será dividido em cinco seções, além da introdução, sendo: referencial teórico, que tem como objetivo fornecer aos leitores conhecimento sobre alguns conceitos e ferramentas utilizadas; metodologia, seguindo os métodos utilizados para fazer este trabalho; desenvolvimento, mostrando a estruturas das aplicações e as principais funcionalidades implementadas; por fim a conclusão, propostas para possíveis trabalhos futuros e as referências usadas conforme o desenvolvimento.

2. Referencial teórico

O modelo adotado para seguir os processos de desenvolvimento deste projeto foi o incremental, considerando uma construção por incrementos através dos requisitos definidos. Alguns conceitos e ferramentas utilizados para o desenvolvimento das aplicações e da documentação do projeto serão apresentados a seguir.

2.1. Modelo Incremental

> [...] desenvolver uma implementação inicial, expô-la aos comentários dos usuários e continuar por meio da criação de várias versões até que um sistema adequado seja desenvolvido. Atividades de especificação e validação são intercaladas, e não separadas, com rápido *feedback* entre todas as atividades (SOMMERVILLE, 2011).

Neste processo, o cliente é o responsável por analisar as partes realizadas do sistema, em que, as mesmas são denominadas um conjunto de funcionalidades da aplicação e cada uma destas parcelas tem sua prioridade para ser feita e entregue.

Conforme apresenta a figura 3, durante o desenvolvimento novas implementações poderão ser integradas posteriormente, após serem concluídas. Com os conjuntos destas funcionalidades desenvolvidas, é formado o sistema como um todo, com a possibilidade de ter futuras melhorias, até concluí-lo.

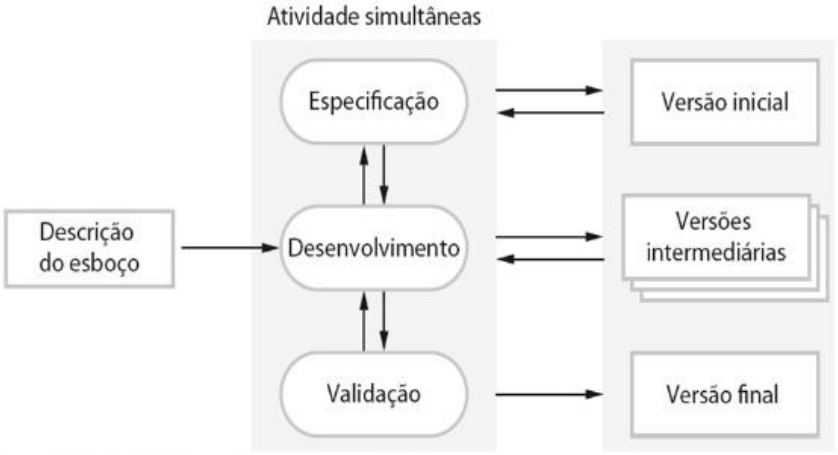

Figura 3. Desenvolvimento Incremental
Fonte: SOMMERVILLE, 2011

2.2. Elicitação de Requisitos

É um processo muito importante feito antes de iniciar o desenvolvimento de um *software*, que facilita o entendimento do problema em específico por via de um documento que define quais requisitos serão implementados. De acordo com o Pressman (2011), elicitação de requisitos é a combinação de elementos de resolução de problemas, elaboração, negociação e especificação. É feita uma abordagem colaborativa e em equipe, para que os processos de levantamento de requisitos sejam feitos, sendo eles, identificação do problema, elementos que propõem uma solução, negociação através de diferentes abordagens e um conjunto prévio dos requisitos atrelados à solução.

2.3. Requisitos funcionais e não funcionais

Os requisitos de um *software* são separados em funcionais e não funcionais. Segundo Sommerville (2011), requisitos funcionais são registros de tarefas que o sistema deve fornecer e como será seu comportamento diante de determinadas situações. Já os requisitos não funcionais

são restrições relacionadas às funções/serviços que a aplicação oferece. Ao fazer o levantamento de requisitos dos requisitos funcionais, podemos representá-los por via da Linguagem de Modelagem Unificada (*UML*) utilizando o diagrama de caso de uso.

> [...] Casos de usos é a parte mais importante da construção de um *software*, utilizando UML dentro do processo iterativo. Observamos que um caso de uso sempre é iniciado por um ator, e que se ele não conseguir identificar um ator estará errado. Um caso de uso pode necessitar da participação de vários atores; e é uma forma de abstração da complexidade. [...] Descreve o mundo como ele é ou como ele é esperado nos cenários principais. Os cenários alternativos descrevem alternativas para cada cenário principal do caso de uso (MEDEIROS, 2004).

2.4. StarUML

É uma ferramenta *open-source* que serve para o desenvolvimento de diagramas *UML* (*Unified Modeling Language*) e *MDA* (*Model Driven Architecture*). Além disso, ela fornece uma boa arquitetura de *plug-ins* possibilitando com que o programador possa fazer uma conversão do diagrama feito para uma linguagem de programação que a ferramenta oferece suporte (STARUML).

2.5. Eclipse

É uma *IDE* (*Integrated Development Environment*) para o desenvolvimento de aplicações Java, que também oferece suporte para o desenvolvimento em C++ e PHP. Esta ferramenta também proporciona ao usuário suporte de *plug-ins* relacionados a gráficos, relatórios, modelagem, testes e outros (ECLIPSE).

2.6. Java

> Ao escrever um programa de computador em uma linguagem orientada a objetos, você criará, em seu computador, um modelo de alguma parte do mundo. As partes das quais o modelo é construído são os objetos que aparecem no domínio do problema. Esses objetos devem ser representados no modelo de computador que estiver sendo criado (BARNES, 2009).

Segundo Deitel (2017), Java é uma linguagem de programação utilizada para desenvolver aplicativos corporativos de grande porte, fornecer aplicativos para dispositivos voltados ao consumo popular (por exemplo, telefones celulares, *smartphones*, televisão, *set-up* boxes etc.) e para muitos outros propósitos. Ainda ela também é uma linguagem chave para desenvolvimento de aplicativos *Android* adequados a *smartphones* e *tablets*.

2.7. *Android*

Com a chegada dos dispositivos móveis, surgiu a necessidade de se criar um sistema operacional que atendesse requisitos específicos. No ano de 2008 surge o *Android* como um projeto apoiado pela *Open Handset Alliance*, uma aliança de diversas empresas na área de *smartphones* e

tecnologia (OHA, 2008).

Segundo OHA (2008), o *Android* é um sistema operacional de código aberto e foi criado com o intuito de permitir que desenvolvedores inventem aplicativos móveis que atendam as necessidades e agradem os usuários, utilizando os recursos oferecidos pelo sistema junto ao dispositivo, criando e sofisticando mais recursos para aqueles que fazem o seu uso (OHA 2008).

2.8. *Android Studio*

Android Studio é uma *IDE* (*Integrated Development Environment*) baseada no IntelliJ IDEA, feita para o desenvolvimento de aplicativos *Android*. Esta ferramenta oferece diversos recursos para melhorar a produtividade na criação de aplicativos, podendo desenvolver as aplicações através das linguagens de programação Java ou Kotlin (AndroidStudio).

2.9. *SQL*

A *SQL* (*Structured Query Language*) é uma linguagem padrão universal usada por Sistemas de Gerenciamento de Banco de Dados (SGBD).

> [...] Ela oferece uma interface de linguagem declarativa de nível mais alto. Assim, o usuário apenas especifica qual deve ser o resultado, deixando para o SGBD a otimização real e as decisões sobre como executar a consulta (AMADEU, 2014).

2.10. Banco de dados no Modelo Relacional

> O modelo relacional representa o banco de dados como uma coleção de relações. Informalmente, cada relação é semelhante a uma tabela de valores ou até certo ponto, a um arquivo plano de registros. Este recebe esse nome porque cada registro tem uma estrutura linear ou plana (AMADEU, 2014).

Segundo Amadeu (2014), na terminologia formal do modelo relacional há os seguintes conceitos: tupla (uma linha), atributo (um cabeçalho da coluna), relação (representa a tabela) e domínio (descrevem os tipos de valores que aparecem em cada coluna).

3. Metodologia

Após a pesquisa quantitativa e qualitativa cujo resultado comprovou a validade da proposta, foi feito o levantamento dos requisitos.

Utilizou-se o modelo incremental durante todo o processo de desenvolvimento de *software*. Dentro do processo incremental, a aplicação se dividiu em iterações que foram incrementadas a cada novo ciclo. Esses ciclos consistiram na repetição dos processos de levantamento de requisitos, desenvolvimento do *software*, teste funcional da aplicação, que consiste em validar a aplicação com base nas especificações funcionais, e a integração da parte desenvolvida ao todo.

As aplicações propostas (ambientes *desktop* e *mobile*) foram desenvolvidas na linguagem de programação Java, sendo cada uma delas criadas por meio das *IDE's* destinadas para cada tipo de aplicação. O armazenamento dos dados foi feito através de uma hospedagem de banco de dados em um servidor *web* gratuito, administrado através da ferramenta phpMyAdmin em conjunto com a linguagem *SQL* (MYSQL).

4. Desenvolvimento

Esta seção retrata as fases do desenvolvimento do projeto como um todo, sendo elas: estrutura do sistema, levantamento e documentação dos requisitos, modelo de dados do sistema e implementação dos *softwares*.

4.1. Estrutura do sistema

Conforme apresenta a figura 4, a estrutura do sistema é dividida em uma aplicação para dispositivos móveis, uma aplicação *desktop* e um servidor *web*. Quanto as funcionalidades propostas:

a. A aplicação *mobile* é destinada as consultas e reservas de produtos.

b. A aplicação *desktop* tem a função de gerenciar os produtos de estoque e gerenciar os usuários do sistema. O servidor é responsável por armazenar o banco de dados *SQL* e prover acesso aos dados para as aplicações, sendo administrado através da ferramenta phpMyAdmin.

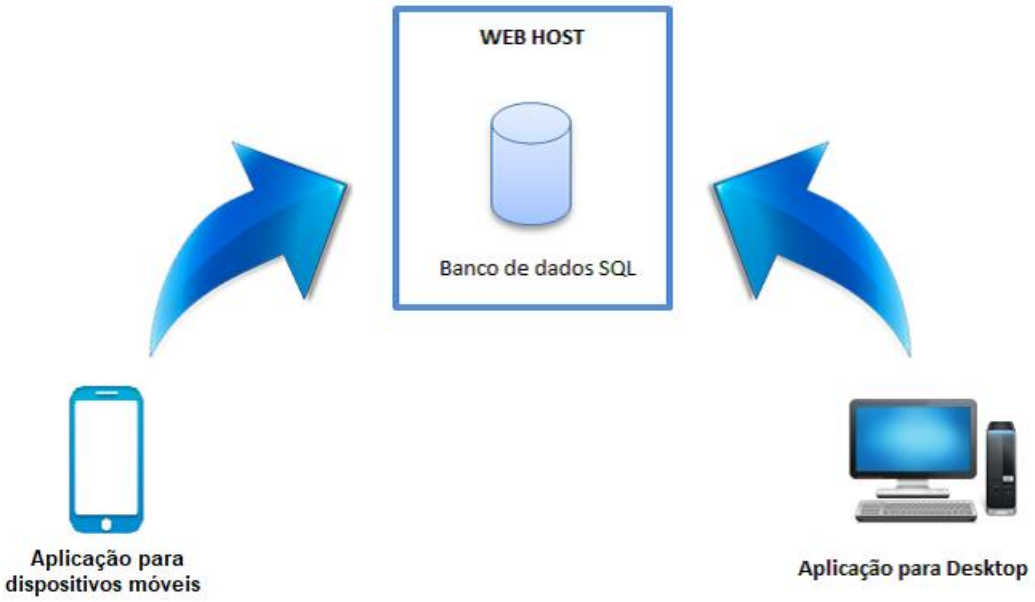

Figura 4. Estrutura do sistema

4.2. Levantamento de requisitos

O levantamento dos requisitos do sistema foi feito baseado em aplicações similares, além da necessidade em melhorar a eficiência do atendimento do vendedor com relação aos consumidores diante de suas escolhas na aquisição de produtos. Primeiro foram feitas listas especificando os requisitos funcionais e não funcionais das aplicações que compõem o sistema como um todo. As figuras 5 e 6 mostram os requisitos funcionais relativos aos *softwares* deste projeto.

RF01 – Cadastrar, excluir, editar e visualizar usuários 'vendedores' Funcionalidade exclusiva do usuário 'Administrador' do sistema. Fazer cadastro, exclusão, edição e visualização de usuários tipo 'Vendedor'.
RF02 – Cadastrar, excluir, editar e visualizar produtos Funcionalidade exclusiva do usuário 'Administrador' do sistema. Fazer cadastro, exclusão, edição e visualização de usuários tipo 'Vendedor'.
RF03 – Efetuar Login Funcionalidade realizada por todos usuários, para obter acesso ao sistema.
RF04 – Recuperar Senha Funcionalidade realizada pelos usuários, ao perder a senha, tem a possibilidade de recuperá-la pelo e-mail cadastrado.
RF05 – Editar, excluir e visualizar reservas de produtos Funcionalidade destinada ao usuário 'vendedor', para reservar produtos pedidos pelos clientes.
RF06 – Vender Produtos Funcionalidade destinada ao usuário 'vendedor', para efetuar a exclusão do produto vendido no estoque.
RF07 – Gerar Nota Funcionalidade destinada ao usuário 'vendedor', após a venda de produtos é gerado uma nota fiscal eletrônica.
RF08 – Cadastrar, excluir, editar e visualizar clientes Funcionalidade que pode ser realizada por todos os usuários, para fazer o cadastro, exclusão, edição e visualização dos clientes da loja.

Figura 5: Requisitos funcionais da aplicação *desktop*

RF01 – Pesquisar Produtos Funcionalidade destinada ao usuário para a pesquisa dos produtos requisitados por clientes.
RF02 – Reserva de Produtos Funcionalidade destinada ao usuário para a reserva de produtos requisitados por clientes.
RF03 – Efetuar Login Funcionalidade realizada por todos os usuários para obter acesso ao sistema.
RF04 – Recuperar Senha Funcionalidade realizada pelos usuários ao perder a senha, em que, os mesmos têm a possibilidade de recuperá-la pelo e-mail cadastrado.
RF05 – Cadastrar e Editar Clientes Funcionalidade que pode ser exercida por todos os usuários, que tem por seu objetivo cadastrar e modificar informações dos clientes do comércio.
RF06 – Alterar Informações do Usuário Funcionalidade que objetiva atualizar as informações pessoais do próprio usuário.

Figura 6: Requisitos funcionais da aplicação *mobile*

Quanto aos requisitos não funcionais (figura 7) foram definidos os mesmos às duas aplicações.

RNF01 – Criptografia na senha dos usuários. (A definir)
RNF02 – Disponibilidade do servidor (000webhost) para a comunicação das aplicações.
RNF03 – Eficiência na estrutura do *software*. (Padrão MVC)

Figura 7: Requisitos não funcionais

Após a especificação dos requisitos, e de acordo com as funcionalidades apresentadas na figura 6 e 7, foram feitos os diagramas de caso de uso de cada aplicação proposta, como apresentam as figuras 8 e 9.

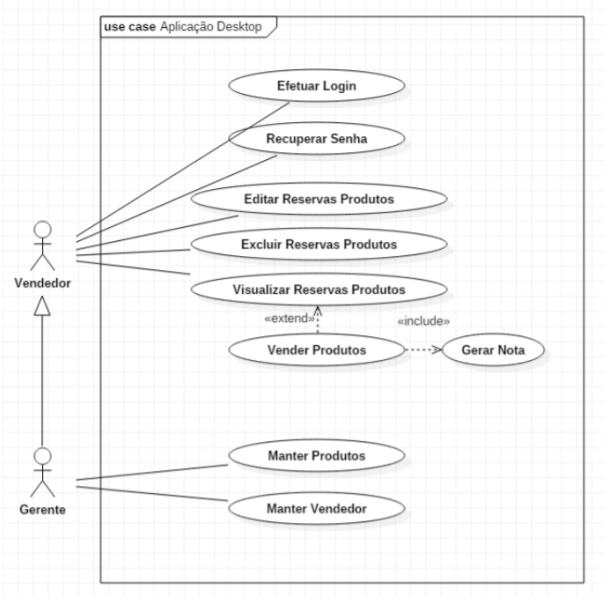

Figura 8: Diagrama de caso de uso da aplicação *desktop*

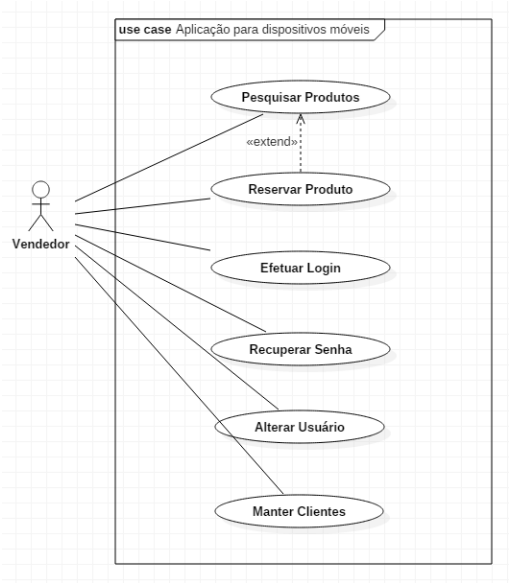

Figura 9: Diagrama de caso de uso da aplicação *mobile*

4.3. Implementação das funcionalidades propostas

Para a troca de dados simultânea entre as aplicações e o banco de dados foi utilizado o WEBHOST, um servidor de hospedagem gratuita. A figura 10 retrata o Modelo de Entidade e Relacionamento (MER) feito para representar a estrutura do banco de dados das aplicações.

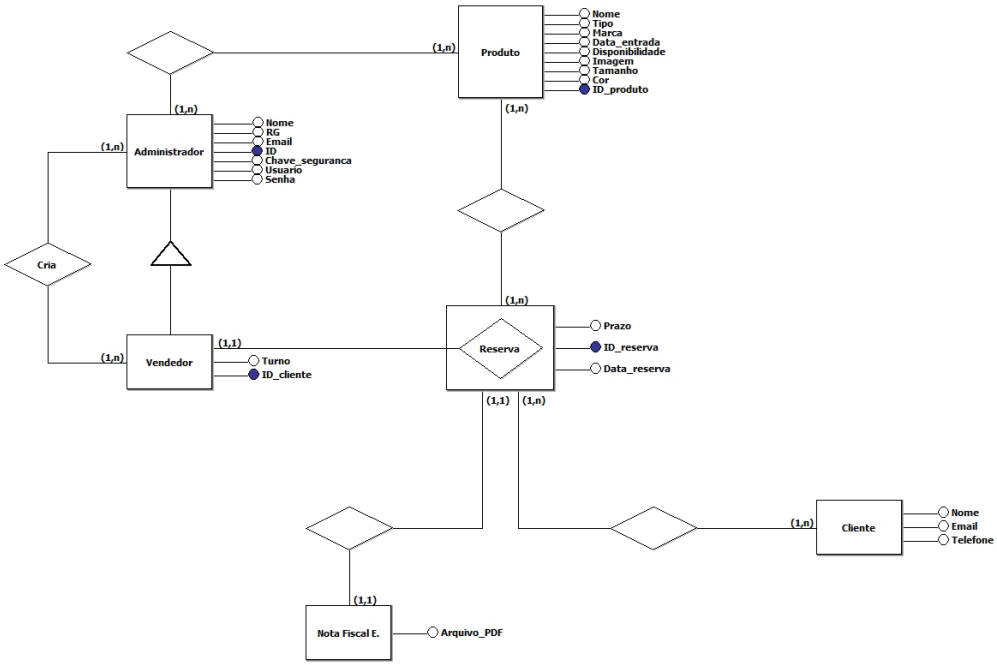

Figura 10: Representação do Modelo Entidade Relacionamento

A aplicação foi desenvolvida conforme o levantamento de requisitos, seguindo o modelo incremental. Iniciou-se o desenvolvimento com o programa *desktop*, sendo desenvolvido por funcionalidades e integrado a um todo. As figuras a seguir representam as interfaces do sistema *desktop*, começando com as imagens 11 e 12 que correspondem às telas de cadastro de usuário. Destacando que o cadastro do primeiro usuário do sistema é obrigatoriamente administrador, visto que, o tipo de usuário vendedor não tem acesso a todas as funcionalidades do sistema, como esboça a figura 8.

Figura 11: Tela de cadastro de usuário (aplicação *desktop*)

Figura 12: Tela de cadastro de usuário (aplicação *desktop*)

A figura 13 representa a tela de autenticação do usuário e a função de recuperar a senha, em que, um código de segurança é enviado para o e-mail do usuário cadastrado, e após o envio é efetuado um redirecionamento para a tela de recuperação de senha, como mostra a figura 14.

Figura 13: Tela de *login* (aplicação *desktop*)

Figura 14: Tela de recuperação de senha (aplicação *desktop*)

A figura 15 corresponde à tela principal do *software*. Por meio dela é possível gerenciar usuários, produtos, clientes, reservas e as próprias informações da conta autenticada. Para facilitar o uso da ferramenta, todas as telas gerenciais seguem o mesmo padrão de interface.

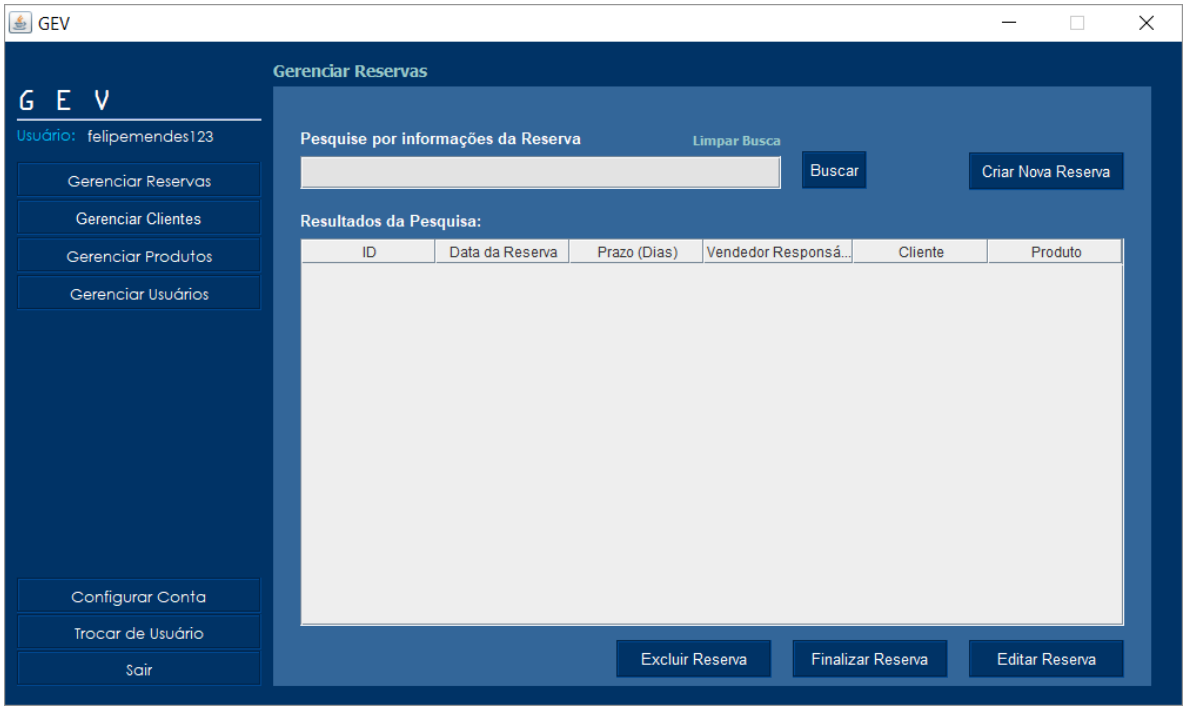

Figura 15: Tela principal (aplicação *desktop*)

Além do padrão da interface gerencial da ferramenta, a imagem acima ilustra a parte gerencial de reservas, em que, a mesma tem uma funcionalidade específica para finalizar uma reserva sempre que uma venda for realizada, enfatizando que uma reserva está obrigatoriamente atrelada a um produto, um vendedor e um cliente. Essa funcionalidade tem o propósito de fazer o decremento do produto reservado no estoque.

A figura abaixo representa as configurações da conta, sendo possível a edição de informações pessoais do usuário autenticado.

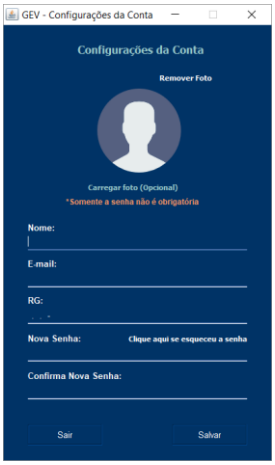

Figura 16: Tela de configurações de conta (aplicação *desktop*)

A aplicação *mobile* contém as funcionalidades de autenticação de usuário, recuperação de senha, pesquisa e reserva de produtos, gerencia clientes e configurações da conta, como foi relatado anteriormente na figura 9. As imagens a seguir representam as interfaces das principais funcionalidades da aplicação *mobile*, começando pela tela inicial, destinada a autenticação do usuário e recuperação da conta.

Figura 17: Tela de autenticação de usuário (aplicação *mobile*)

Assim como o *software desktop*, a aplicação *mobile* também disponibiliza um *menu* com acesso a todas as funcionalidades disponíveis.

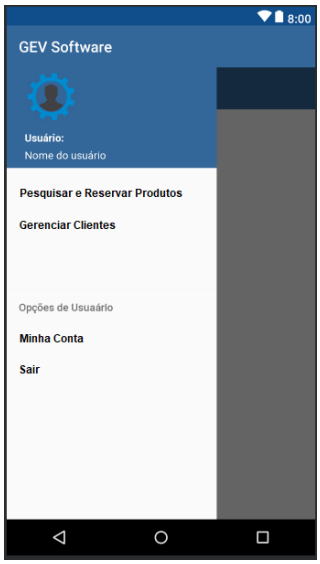

Figura 18: Tela principal (aplicação *mobile*)

As figuras 19 e 20 representam a pesquisa e reserva de produtos, em que, o usuário pode pesquisar o produto, ver os detalhes do mesmo e fazer a reserva.

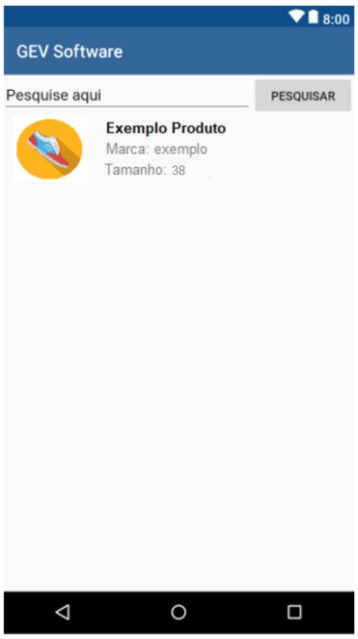

Figura 19: Tela de pesquisa de produtos (aplicação *mobile*)

Figura 20: Tela de reserva de produtos (aplicação *mobile*)

Para finalizar, abaixo temos figuras que representam trechos de códigos que foram importantes para o desenvolvimento do sistema como um todo.

A imagem 21 corresponde a um *singleton*, utilizado para manter as informações em uma instancia única de um objeto, por exemplo, os dados do usuário autenticado.

```java
// Cria um Singleton para unificar o usuário quando logado, sem novas instâncias
public static Modelo_Usuario getInstance() {
    if (instancia == null) {
        instancia = new Modelo_Usuario();
    }
    return instancia;
}
```

Figura 21: Exemplo *Singleton*

A figura 22 representa um trecho de código usado na aplicação *mobile*, para obter os dados do usuário autenticado, fazendo uma requisição ao banco de dados.

```java
Ion.with( context: Login.this) LoadBuilder<B>
        .load( uri: Modelo_URL.host + "autenticacao_login.php") B
        .setBodyParameter( name: "usuario", usuario) U
        .setBodyParameter( name: "senha", senha) U
        .asJsonArray() ResponseFuture<JsonArray>
        .setCallback(new FutureCallback<JsonArray>() {
            @Override
            public void onCompleted(Exception e, JsonArray result) {


                for (int i =0; i < result.size(); i++){
                    JsonObject jsonObject = result.get(i).getAsJsonObject();

                    Modelo_Usuario.setInstancia(controle_usuario.setModeloUsuario(jsonObject.get("id").getAsInt(),
                            jsonObject.get("nome").getAsString(),
                            jsonObject.get("rg").getAsString(),
                            jsonObject.get("email").getAsString(),
                            jsonObject.get("tipo").getAsString(),
                            jsonObject.get("usuario").getAsString(),
                            jsonObject.get("senha").getAsString(),
                            jsonObject.get("chaveSeguranca").getAsString()));
                }
```

Figura 22: Exemplo de requisição ao banco

5. Conclusão

O objetivo principal deste trabalho foi produzir um sistema para obter uma maior eficiência na otimização de consultas de produtos e no processo de venda de lojas que têm o segmento de vestuários de calçados. O sistema GEV atende a ideia inicial, em que, as principais funcionalidades relacionadas ao objetivo principal foram desenvolvidas. Entretanto, a criptografia de senhas e a funcionalidade de gerar nota fiscal serão trabalhos futuros.

Para o desenvolvimento da proposta houve, principalmente, a aplicação de conceitos das áreas de conhecimento do curso: engenharia de *software*, programação orientada a objetos, linguagem de programação, estrutura de dados, arquitetura de *software* e banco de dados.

Uma das maiores dificuldades foi o desenvolvimento da aplicação *mobile* em conjunto com as outras partes do sistema, por falta de conhecimento prévio na área. Porém, foram adquiridos novos conhecimentos relacionados à programação *mobile*.

Como trabalhos futuros é possível complementar o sistema com recursos que contemplem outros segmentos de vendas, considerando um estudo para coleta de requisitos, além de um

possível SGBD mais robusto com maiores opções de segurança quanto ao acesso aos dados armazenados. Outra proposta são os testes de usabilidade para mapeamento de melhorias quanto às funcionalidades, como as interfaces. Quanto a futuras funcionalidades, pode ser feito um estudo para associar o sistema GEV com uma *API* a fim de gerar uma nota fiscal atrelada aos produtos da venda através de uma simples consulta no banco de dados, além de novas ferramentas para a gestão de vendas, implantação de gráficos que podem ser gerados a partir dos dados, amplificando a gestão das informações relacionadas ao usuário final.

6. Referências

AMADEU, C. V. **Banco de Dados**. 1. ed. São Paulo: Pearson Education do Brasil, 2014.

Android Studio. Disponível em: <https://developer.android.com/studio/>. Acesso em: 05 abr. 2018.

BARNES, D. J.; K´OLLING, M. **Programação orientada a objetos com Java**: Uma introdução a prática usando o BlueJ. 4. ed. São Paulo, SP: Prentice Hall, 2009.

BARBOSA, T. D.; TRIGO, A. C.; SANTANA, L. C. **Qualidade no atendimento como fator de crescimento empresarial**. Disponível em: <http://www.cairu.br/riccairu/pdf/artigos/2/08 QUALIDADE ATENDIMENTO FATOR.pdf>. Acesso em: 10 mar. 2018.

DEITEL, P. **Java:** como programar. 10. ed. São Paulo: Pearson Education do Brasil, 2017.

ECLIPSE. Desktop IDEs. Disponıvel em: <https://www.eclipse.org/ide/>. Acesso em: 05 abr. 2018.

MEDEIROS, E. **Desenvolvendo *Software* com *UML* 2.0 Definitivo**. São Paulo: Pearson Makron Books, 2004.

MYSQL. **Manual de Referência do MySQL 4.1**. Disponível em: <https://downloads.mysql.com/docs/refman-4.1-pt.a4.pdf>. Acesso em: 20 abr. 2018.

OPEN HANDSET ALLIANCE (OHA). **Android**. Disponível em: <http://www.openhandsetalliance.com/android overview.html>. 2008. Acesso em: 09 mar. 2018.

PRESSMAN, R. S. Engenharia de *software*: uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011.

PROGRAMANEX. **Programa NEX**. Disponível em: <https://www.programanex.com.br/controle-de-estoque>. Acesso em: 07 mar. 2018.

SOMMERVILLE, I. **Engenharia de *Software***. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

STARUML. **About StarUML**. Disponível em: <http://staruml.sourceforge.net/v1/about.php>. Acesso em: 18 abr. 2018.

WALKER, D. **O cliente em primeiro lugar:** O atendimento e a satisfação do cliente como uma arma poderosa de fidelidade e vendas. São Paulo: Makron Books, 1991.