SENAC SÃO PAULO

UNIDADE LAPA TITO

William Francisco Leite

Kelly Taniguchi

Osnir Estevam de Lima

Desenvolvimento de API para Cotação de Produtos em Redes Atacadistas

SÃO PAULO

2016

William Francisco Leite

Kelly Taniguchi

Osnir Estevam de Lima

Desenvolvimento de API para Cotação de Produtos em Redes Atacadistas

Trabalho de Conclusão de Curso apresentado ao Centro Universitário Senac – unidade Lapa Tito, como exigência parcial para obtenção do grau de especialista em Engenharia Web.

SÃO PAULO

2016

Ficha catalográfica elaborada pela Biblioteca do Centro Universitário Senac

T266t Teixeira, Bruno Louredo

Transparente como a água: desenvolvimento de um portal de transparência de dados sobre temas hídricos / Bruno Louredo Teixeira et. al. - São Paulo, 2015.

67 f.

Orientador: Profº. Me Daniel Paz de Araújo

Trabalho de Conclusão de Curso (Especialização em Engenharia Web) – Centro Universitário Senac – Unidade Lapa Tito, São Paulo, 2015.

1. Transparência de dados 2. Conjunto de dados 3. Água 4. Visualização de dados 5. Gamificação I. Felix, Camila da Silva (autora) II. Lins, Daniel Alves de Sá (autor) III. Alves, João Ricardo Rampinelli (autor). IV. Araújo, Daniel Paz de (Orient.) V. Título. CDD 005.01

William Francisco Leite

Kelly Taniguchi

Osnir Estevam de Lima

Desenvolvimento de API para Cotação de Produtos em Redes Atacadistas

Trabalho de Conclusão de Curso apresentado ao Centro Universitário Senac – unidade Lapa Tito, como exigência parcial para obtenção do grau de especialista em Engenharia Web.

Orientador: Prof. Daniel Paz de Araújo

A banca examinadora dos Trabalhos de Conclusão, em sessão pública realizada em \_\_\_\_\_\_\_\_\_\_/\_\_\_\_ /\_\_\_\_\_\_ , considerou o(a) candidato(a):

1) Examinador(a)

2) Examinador(a)

3) Presidente

Dedicatória (opcional)

**AGRADECIMENTOS (opcional)**

Aos...

Epígrafe (opcional)

**RESUMO**

Otimizar o tempo é uma tarefa determinando para o sucesso de um estabelecimento. Utilizando a internet como ferramenta de pesquisa é possível criar uma ponte de ligação direta entre clientes e fornecedores, onde a informação necessária chega a mão do cliente sem que haja necessidade de locomoção beneficiando o cliente com o melhor custo benefício na escolha de matéria prima e ao fornecedor com maior procura em seus produtos.

O objetivo deste trabalho é apresentar uma API de consumo de dados para redes atacadistas para que supermercados atacadistas possam divulgar em tempo real preços e promoções de seus produtos, através de um site que possibilite a distribuição dos dados coletados diariamente. Para construção dessa aplicação foram aplicados os conhecimentos adquiridos durante todo o curso, utilizando conceitos e práticas de Engenharia de Web e baseando nos fundamentos teóricos de pesquisas realizadas, como o livro Engenharia de Software, Pressman, 2006 e Engenharia Web, Pressman, 2009.

**Palavras-chave**: 1. Cotação. 2. API. 3. Engenharia Web. 4. Otimização.

***ABSTRACT***

Optimize time is a task for determining the success of an establishment. Using the internet as a research tool you can create a direct bridge between customers and suppliers, where the necessary information reaches the customer's hand without the need of mobility benefiting the customer with the best value in the choice of raw materials and supplier with increased demand for their products.

The aim of this paper is to present a data consumer API for wholesale networks for supermarkets wholesalers to disseminate real-time pricing and promotion of its products through a website that allows the distribution of data collected daily. To build this application have applied the knowledge acquired throughout the course, using concepts and Web Engineering practices and based on the theoretical foundations of research carried out, as the book Engenharia de Software, Pressman, 2006 and Engenharia Web, Pressman, 2009.

**Keywords**: 1. Quotation. 2. API. 3. Web Engineering. 4. Optimization.

**LISTA DE ILUSTRAÇÕES**

[Figura 1 - Interoperabilidade de uma API 1 5](#_Toc454907587)

[Figura 2: Modelo Entidade 6](#_Toc454907588)

[Figura 3: Modelo Relacionamento 7](#_Toc454907589)

[Figura 4: Modelo Atributo 7](#_Toc454907590)

[Figura 5 - Gráfico para Glory of REST 10](#_Toc454907591)

[Figura 6: Clico dos testes no TDD 13](#_Toc454907592)

[Figura 7 - BPM (Business Process Model) 20](\\\\UBUNTU-SERVIDOR\\publica\\TCC\\Modelo SENAC(V2).docx" \l "_Toc454907593)

[Figura 9 - Big Picture (Arquitetura) 20](\\\\UBUNTU-SERVIDOR\\publica\\TCC\\Modelo SENAC(V2).docx" \l "_Toc454907594)

[Figura 7 - BPM (Business Process Model) 21](\\\\UBUNTU-SERVIDOR\\publica\\TCC\\Modelo SENAC(V2).docx" \l "_Toc454907595)

[Figura 8 - MER (Modelo Entidade Relacional) 25](#_Toc454907596)

[Figura 9 - Big Picture (Arquitetura) 26](\\\\UBUNTU-SERVIDOR\\publica\\TCC\\Modelo SENAC(V2).docx" \l "_Toc454907597)

[Figura 10 - Persona (Pedro da Silva) 28](#_Toc454907598)

[Figura 11 - Persona (Bruno Siqueira) 28](#_Toc454907599)

[Figura 12 - Persona (Ana Carolina) 29](#_Toc454907600)

[Figura 13 – Moodboard 30](#_Toc454907601)

[Figura 14 - Style Guide (Tipografia) 31](#_Toc454907602)

[Figura 15 - Style Guide (Cores) 32](#_Toc454907603)

[Figura 16 - Style Guide (Ícones) 33](#_Toc454907604)

[Figura 17 - Story board (Pesquisa) 34](#_Toc454907605)

[Figura 18 - Story board (Cotação) 34](#_Toc454907606)

**LISTA DE TABELAS**

[Tabela 1 - Requisitos Funcionais 22](#_Toc454907583)

[Tabela 2 - Requisitos Não Funcionais 22](#_Toc454907584)

[Tabela 3 - Recursos disponibilizados pela API 23](#_Toc454907585)

[Tabela 4 - Descrição dos parâmetros das URIs 23](#_Toc454907586)

**SUMÁRIO**

[1. INTRODUÇÃO 1](#_Toc454909128)

[1.1. Contexto e Justificativa 1](#_Toc454909137)

[1.2. Problema de Pesquisa 1](#_Toc454909138)

[1.3. Objetivo 2](#_Toc454909139)

[1.4. Escopo 2](#_Toc454909140)

[1.5. Metodologia 3](#_Toc454909141)

[1.6. Organização do Trabalho 3](#_Toc454909143)

[2. REVISÃO DA LITERATURA 5](#_Toc454909144)

[2.1. API 5](#_Toc454909145)

[2.2. Modelo de Entidade Relacional (MER): 6](#_Toc454909146)

[2.2.1. Entidades 6](#_Toc454909147)

[2.2.2. Relacionamentos 6](#_Toc454909148)

[2.2.3. Atributo 7](#_Toc454909149)

[2.3. BPM (Business Process Model) 8](#_Toc454909150)

[2.4. Protocolo HTTP (Hyper Transfer Protocol) 8](#_Toc454909151)

[2.5. Arquitetura REST 8](#_Toc454909152)

[2.6. RESTful 9](#_Toc454909153)

[2.7. Swagger 11](#_Toc454909154)

[2.8. TDD (Test Driven Development) 11](#_Toc454909155)

[2.8.1. Teste de Unidade 12](#_Toc454909156)

[2.9. Personas 13](#_Toc454909157)

[2.10. Design de Interação 14](#_Toc454909159)

[2.11. Framework Material Design 14](#_Toc454909160)

[2.12. Framework AngularJS 15](#_Toc454909161)

[2.13. Moodboard 15](#_Toc454909163)

[2.14. StyleGuide 15](#_Toc454909164)

[2.15. Story board 16](#_Toc454909167)

[3. A COTAÇÃO DE PRODUTOS EM ATACADISTAS 18](#_Toc454909168)

[3.1. Dificuldade em encontrar o melhor custo benefício 18](#_Toc454909169)

[3.2. Cotando os produtos em apenas um local 19](#_Toc454909170)

[4. DESENVOLVIMENTO DA API PARA COTAÇÃO DOS PRODUTOS 21](#_Toc454909171)

[4.1. BPM (Business Process Model) 21](#_Toc454909172)

[4.2. Requisitos 22](#_Toc454909173)

[4.2.1. Requisitos Funcionais 22](#_Toc454909174)

[4.2.2. Requisitos Não Funcionais 22](#_Toc454909175)

[4.3. URIs (Recursos) da API 23](#_Toc454909176)

[4.3.1. Descrição dos Parâmetros 23](#_Toc454909177)

[4.4. MER (Modelo Entidade Relacional) 24](#_Toc454909178)

[4.5. Big Picture (Arquitetura da Solução) 25](#_Toc454909179)

[4.6. Plano de Teste 26](#_Toc454909180)

[4.6.1. Testes URI 26](#_Toc454909181)

[4.6.2. TDD 27](#_Toc454909182)

[4.6.3. Consumo da API 27](#_Toc454909183)

[4.7. Desenvolvimento do Protótipo 27](#_Toc454909184)

[4.7.1. Personas 27](#_Toc454909185)

[4.7.2. Design de Interação 29](#_Toc454909186)

[4.7.3. Moodboard 29](#_Toc454909187)

[4.7.4. Style Guide 30](#_Toc454909188)

[4.7.5. Tipografia 30](#_Toc454909189)

[4.7.6. Cores 31](#_Toc454909190)

[4.7.7. Ícones 32](#_Toc454909191)

[4.7.8. Story board 33](#_Toc454909192)

[5. METODOLOGIA DE PESQUISA 35](#_Toc454909194)

[5.1.1. Instrumentos de Pesquisa 35](#_Toc454909197)

[5.1.1.1. Mercado 35](#_Toc454909198)

[5.1.1.2. Cientifico 35](#_Toc454909199)

[5.1.2. Coleta de Dados 35](#_Toc454909200)

[5.1.3. Resultados Esperados 36](#_Toc454909201)

[6. CONSIDERAÇÕES FINAIS 37](#_Toc454909203)

[REFERÊNCIAS BIBLIOGRÁFICAS 38](#_Toc454909204)

1. INTRODUÇÃO
   1. Contexto e Justificativa

O tempo é curto e precioso para todos, podemos otimizá-lo com soluções móveis.

As redes atacadistas brasileiras atualmente não publicam em seus sites informações sobre valores dos produtos, apenas promoções, sendo assim dificulta a escolha dos clientes sobre qual estabelecimento oferece o melhor custo benefício.

Certas tarefas podem ser difíceis de serem realizadas, tais como comparação de preços entre estabelecimentos atacadistas. Onde o cliente tem que se locomover até os estabelecimentos atacadistas para pesquisar os melhores preços de produtos.

Analisando o contexto do problema, entende-se que uma possível solução contendo diversas redes atacadistas e seus respectivos produtos reduziria o transtorno do cliente. Ao realizar as pesquisas de comparação do preço dos produtos necessários de acordo com o tipo de estabelecimento do mesmo.

Realizando pesquisas com o Público-alvo visamos coletar o máximo de informações sobre as dificuldades pesquisar por preço e localização dos produtos.

Para obter o embasamento teórico foram feitas diversas pesquisas em materiais acadêmicos, como monografias e teses de doutorado com foco em desenvolvimento mobile.

Através da engenharia WEB, computação distribuída e metodologia cientifica será possível desenvolver uma solução para disponibilizar informações sobre os produtos. O cliente poderá realizar a comparação de preços entre as redes atacadistas.

Utilizando essa solução o cliente terá mais informações disponíveis para tomar decisões relacionadas ao custo benefício. Acompanhando a tendência de mobilidade os clientes poderão utilizar a solução em Smartphones, trazendo maior conforto e comodidade.

* 1. Problema de Pesquisa

Observando o mercado de redes de supermercados atacadistas é notável a dificuldade para realizar a cotação entre diversos estabelecimentos. Principalmente pela distância entre um supermercado e outro, torna-se inviável comparar preços de produtos em mais de um estabelecimento.

A questão central do trabalho está estabelecida em como fornecer as informações necessárias aos clientes com um instrumento que os coloque em posição favorável com relação ao custo benefício.

- Como gerar vantagem competitiva aos supermercados atacadistas através da troca de informação?

- Como auxiliar clientes na cotação de produtos?

- Quais informações fornecer aos consumidores para que tenham o benefício desejado?

* 1. Objetivo

Os clientes encontram dificuldade em localizar os preços dos produtos pois as redes atacadistas não fornecem essa informação online, publicando apenas promoções. Necessitando do deslocamento do cliente até os estabelecimentos para tal conferencia.

Criar um ambiente contendo diversas redes atacadistas e seus respectivos produtos seria a solução para o problema apresentado, pois reduziria o transtorno do cliente ao realizar pesquisas de comparação de preço.

A API deverá permitir comunicação com um segundo aplicativo em local remoto através de uma série de chamados via internet;

A API fará o intercâmbio de informações e deverá ser administrado através de serviços web (xml);

Todos os recursos disponibilizados pela API serão baseados em Webservices Restful e as mensagens trafegadas no padrão JSON;

A API deve consumir os dados enviados pelos supermercados atacadistas através de serviços web (xml) seguindo padrão pré-estabelecido.

* 1. Escopo

O problema em estudo é como apresentar os diferentes preços para determinados produtos, afim de que o consumidor do estabelecimento atacadista possa buscar o produto que retornará melhor custo benefício, pesquisando entre diversos estabelecimentos.

O estabelecimento atacadista deverá ser previamente cadastrado no sistema através de um processo manual no sistema administrativo, com todas as informações relacionadas a endereço e geolocalização.

Para iniciar o carregamento dos dados deverá ser realizada a coleta de informações de estabelecimentos e produtos, através de arquivo de serviço web no formato XML solicitado aos estabelecimentos atacadistas.

 Serão utilizadas para coleta de informações apenas redes de supermercados atacadistas do estado de São Paulo.

Os dados serão disponibilizados através de uma API RESTful onde o retorno será com textos no formato JSON, os recursos (URIs) contidos nessa API são apenas para consulta.

A API não disponibiliza nenhum recurso para entrada (gravação) de informações das redes atacadistas ou dos produtos, atualizações em promoções e produtos serão feitas através dos arquivos XMLs disponibilizados pelas redes Atacadistas.

* 1. Metodologia

Como instrumento de coleta de dados serão realizadas entrevistas utilizando personas e investigando cenários. Na elaboração de coleta de dados foi necessário estudar o comportamento de consumidores de supermercados atacadistas. O instrumento foi aplicado a três consumidores de redes de supermercados atacadistas.

Como complementação, serão realizadas pesquisas bibliográficas baseadas em livros, revistas e sites sobre o tema em questão, de maneira, conduza o entendimento do tema em questão permitindo o estabelecimento de referência teórica. Os textos e conteúdos serão elaborados com as conclusões extraídas a partir dessas leituras.

* 1. Organização do Trabalho

O presente trabalho está organizado da seguinte forma:

No Capítulo 2 - Revisão da Literatura aborda em quais teorias e pesquisas este trabalho se fundamentou.

No Capítulo 3 – É detalhado o problema com base em fundamentação teórica e também o detalhamento da solução e os resultados esperados.

No Capítulo 4 - Serão apresentados os detalhes do desenvolvimento da solução.

No Capítulo 5 – Será descrito as metodologias de pesquisa do trabalho.

No Capítulo 6 – São expostas as considerações finais.

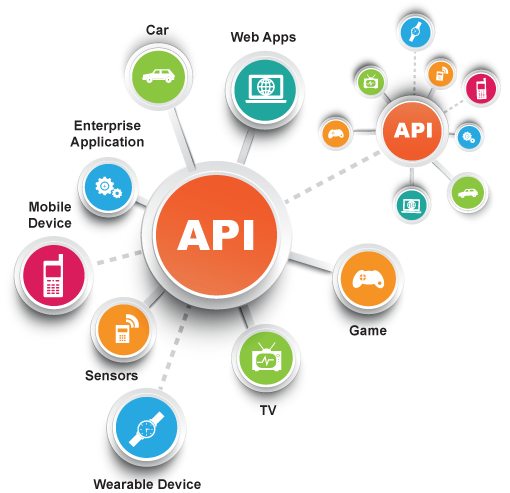
1. REVISÃO DA LITERATURA
   1. API

Praticamente todos os softwares tem de pedir outro software para fazer algumas coisas para ele.

Para alcançar este objetivo, o programa perguntando utiliza um conjunto de solicitações padronizados, chamados interfaces de programação de aplicativos (API), que foram definidos para o programa que está sendo chamado.

Quase todas as aplicações dependem das APIs do sistema operacional subjacente para executar tais funções básicas como o acesso ao sistema de arquivos. Em essência, API de um programa define a maneira correta para um desenvolvedor para solicitar serviços a partir desse programa. (ORENSTEIN, 2016)

Figura 1 - Interoperabilidade de uma API 1



Fonte: Happiest Minds Technologies (2016)

* 1. Modelo de Entidade Relacional (MER):

A técnica de modelagem de dados mais difundida e utilizada é a abordagem entidade-relacionamento (ER). Nesta técnica, o modelo de dados é representado através de um modelo entidade-relacionamento (modelo ER). Usualmente, um modelo ER é representado graficamente, através de um diagrama entidade relacionamento (DER). A abordagem ER foi criada em 1976 por Peter Chen. Ela pode ser considerada como um padrão de fato para modelagem conceitual.

Mesmo as técnicas de modelagem orientada a objetos que têm surgido nos últimos anos baseiam-se nos conceitos da abordagem ER. (HEUSER, 1998, p. 11)

* + 1. Entidades

Uma entidade representa, no modelo conceitual, um conjunto de objetos da realidade modelada. Como o objetivo de um modelo ER é modelar de forma abstrata um BD, interessam-nos somente os objetos sobre os quais deseja-se manter informações. Alguns exemplos de entidades poderiam ser os produtos, os tipos de produtos, as vendas ou as compras. Já em um sistema de contas correntes, algumas entidades podem ser os clientes, as contas correntes, os cheques e as agências. Observe que uma entidade pode representar tanto objetos concretos da realidade (uma pessoa, um automóvel), quanto objetos abstratos (um departamento, um endereço2). (HEUSER, 1998, p. 12)

Figura 2: Modelo Entidade



Fonte: Livro Projeto de Banco de Dados (1998)

* + 1. Relacionamentos

Além de especificar os objetos sobre os quais deseja-se manter informações, o DER deve permitir a especificação das propriedades dos objetos que serão armazenadas no BD. Uma das propriedades sobre as quais pode ser desejável manter informações é a associação entre objetos. Exemplificando, pode ser desejável saber quais pessoas estão associadas a quais departamentos em uma organização.

Relacionamento = conjunto de associações entre entidades

Figura 3: Modelo Relacionamento



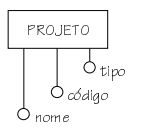
Fonte: Livro Projeto de Banco de Dados (1998)

Em um DER, um relacionamento é representado através de um losango, ligado por linhas aos retângulos representativos das entidades que participam do relacionamento. (HEUSER, 1998, p. 13)

* + 1. Atributo

Para associar informações a ocorrências de entidades ou de relacionamentos usa-se o conceito de atributo.

Figura 4: Modelo Atributo



Fonte: Livro Projeto de Banco de Dados (1998)

Atributos são representados graficamente conforme mostra a Figura 2.14. A figura expressa que a cada ocorrência de PROJETO é associado exatamente um nome, um código e um tipo.

Na prática, atributos não são representados graficamente, para não sobrecarregar os diagramas, já que muitas vezes entidades possuem um grande número de atributos. Prefere-se usar uma representação textual que aparece separadamente do diagrama ER. Ao final deste capítulo, é fornecida uma possível sintaxe para uma representação textual dos atributos. No caso de ser usado um software para construção de modelos ER, o próprio software encarrega-se do armazenamento da lista de atributos de cada entidade em um dicionário de dados. (HEUSER, 1998, p. 21)

* 1. BPM (Business Process Model)

Um modelo padrão de Processos de Negócios e Notation (BPMN) fornece para as empresas a capacidade de compreender os seus procedimentos internos de negócios em uma notação gráfica e dará as organizações a capacidade de comunicar esses procedimentos de uma forma padrão. Além disso, a notação gráfica irá facilitar o entendimento das colaborações de desempenho e transações de negócios entre as organizações. Isso irá garantir que as empresas vão compreender-se e participantes em seus negócios e permitirá que as organizações para se adaptarem às novas circunstâncias de negócios interna e B2B rapidamente. (OMG , 2016)

* 1. Protocolo HTTP (Hyper Transfer Protocol)

O Hypertext Transfer Protocol (HTTP) é um protocolo em nível de aplicação, para colaboração de sistemas distribuídos, hipermídia. É genérico, não guarda estado (stateless), o protocolo pode ser usado para diversos tipos de tarefas além do uso para hipertexto, como servidores de nomes e sistemas de gerenciamento de objetos distribuídos. Através da extensão dos seus métodos de requisição, códigos de erro e cabeçalhos.

Uma característica do protocolo HTTP é entrada e a negociação para representação dos dados, permitindo que os sistemas possam ser construídos independentes dos dados recebidos. (W3, 2016)

* 1. Arquitetura REST

Representational State Transfer (REST) é uma abstração dos elementos arquitetônicos dentro de um sistema hipermídia distribuído. REST ignora os detalhes da implementação do componente e da sintaxe do protocolo, concentrando-se apenas nos papéis dos componentes, as restrições sobre sua interação com outros componentes, e sua interpretação de elementos dos dados significativos.

REST é diferente de protocolos como SOAP ou XML-RPC, é mais uma filosofia ou um conjunto de princípios do que um protocolo propriamente dito. Ele consiste em um conjunto de ideias sobre como os dados podem ser transferidos de modo elegante e tira grande vantagem dos recursos (verbos) disponíveis no protocolo HTTP.

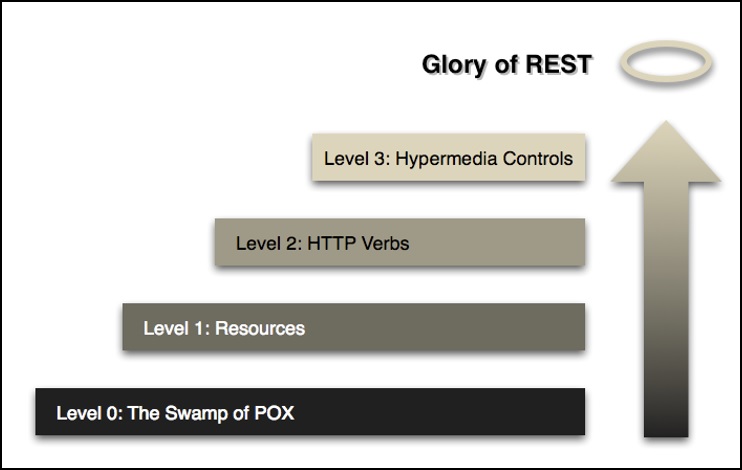
O termo REST foi proposto pela primeira vez por Roy Fielding (um dos criadores do protocolo HTTP) em sua tese de doutorado que foi publicada no ano de 2000. Roy Fielding define REST como um estilo de arquitetura para sistemas, essa arquitetura abrange as restrições fundamentais sobre os componentes, conectores e dados que definem a base da arquitetura WEB e, portanto, a essência do seu comportamento com um aplicativo baseado na rede. (FIELDING, 2000)

* 1. RESTful

O termo RESTful se refere a implementação de um serviço WEB que utilize o estilo arquitetural REST, ou seja, é a capacidade do serviço em disponibilizar seus recursos seguindo as diretrizes da arquitetura REST.

Para desenvolver uma API RESTful seguindo todas as restrições da arquitetura REST existe um modelo conhecido como “Richardson Maturity Model”. Ele descreve quatro níveis (0-3) contendo especificações a serem seguidas, quanto mais a API RESTful for aderente aos níveis, maior será sua compatibilidade com a arquitetura REST chegando ao termo “Glory of REST” onde a API é reconhecida como bem estruturada e desenhada.

Figura 5 - Gráfico para Glory of REST



Fonte: Rest in Practice (2010)

Os 4 níveis são:

Nível 0: Utiliza-se do protocolo HTTP como camada de transporte para interações remotas, o HTTP é usado apenas como uma forma de RPC (Remote Procedure Call). Todos os serviços são centralizados em um único endpoint, ou seja, todas as solicitações são feitas em uma única URI. Porém explora-se muito pouco os recursos HTTP, usando poucos verbos, não permitindo múltiplos tipos de media, uso de headers e código de status. Podendo ser utilizado apenas o verbo POST para inserir/editar/deletar recursos e o verbo GET para consultar.

Nível 1: Diferente do nível 0, agora é usado URIs individuais para cada recurso (Resource), ao invés de centralizar todas as chamadas em um único endpoint. Também é utilizado URIs únicas para acessar itens individuais de recursos. Nesse nível ainda não é explorado verbos HTTP, múltiplos tipos de retorno de media, links entre serviços e etc.

Nível 2: Para chegar nesse nível é necessário que a API esteja seguindo as práticas dos níveis 0 e 1. Nesse nível a API deve acrescentar o uso dos Verbos, Headers e Código de status do protocolo HTTP.

Usando os verbos de maneira apropriada:

\* GET (operações que não modificam recursos)

\* POST (inserções de novos recursos)

\* PUT (atualização completa do recurso)

\* PATCH (atualização parcial de um recurso)

\* DELETE (exclusão de recursos)

Nível 3: Chegando nesse nível a API já possui um nível de qualidade elevado, mas para atingir o “Glory of REST” ainda é necessário implementar o controle de hipermídia. Essa implementação consiste em usar links para outros recursos e coleções dentro da API, por exemplo um controle de paginação dos dados, onde através de um valor informado na URI é possível selecionar qual página deve ser retornada pela API. (FOWLER, 2016)

* 1. Swagger

Swagger é um software open-source e 100% free, é uma representação simples e poderosa de API RESTful. Com o maior ecossistema de de ferramentas API do planeta, milhares de desenvolvedores estão utilizando Swagger em quase todas as linguagens de programação e ambiente de implantação. Com uma API habilitada para Swagger, é possível obter a documentação interativa, geração de SDK cliente e descoberta. (SWAGGER, 2016)

* 1. TDD (Test Driven Development)

TDD ficou bastante popular após a publicação do livro TDD: By Example, do Kent Beck, em 2002. O próprio Kent afirma que TDD não foi uma ideia totalmente original. Ele conta que em algum momento de sua vida, que leu em algum dos livros de seu pai (que também era programador), sobre uma técnica de testes mais antiga, onde o programador colocava na fita o valor que ele esperava daquele programa, e então o programador desenvolvia até chegar naquele valor.

Ele próprio conta que achou a ideia estúpida. Qual o sentido de escrever um teste que falha? Mas resolveu experimentar. Após a experiência, ele disse que “as pessoas sempre falavam pra ele conseguir separar o que o programa deve fazer, da sua implementação final, mas que ele não sabia como fazer, até aquele momento em que resolveu escrever o teste antes.”

Daquele momento em diante, Kent Beck continuou a trabalhar na ideia. Em 1994, ele escreveu o seu primeiro framework de testes de unidade, o SUnit (para Smalltalk). Em 1995, ele apresentou TDD pela primeira vez na OOPSLA (conferência muito famosa da área de computação. Já em 2000, o JUnit surgiu e Kent Beck, junto com Erich Gamma, publicou o artigo chamado de “Test Infected” [2], que mostrava as vantagens de se ter testes automatizados e como isso pode ser viciante. Finalmente em 2002, Kent Beck lançou seu livro sobre isso, e desde então a prá- tica tem se tornado cada vez mais popular entre os desenvolvedores. (ANICHE, 2012, p. 28)

* + 1. Teste de Unidade

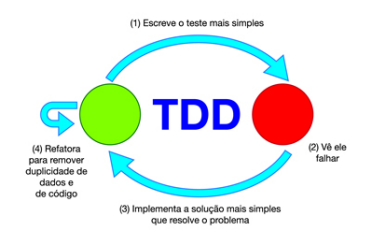
Imagine-se passeando em uma loja virtual qualquer na web. Ao selecionar um produto, o sistema coloca-o no seu carrinho de compras. Ao finalizar a compra, o sistema fala com a operadora de cartão de crédito, retira o produto do estoque, dispara um evento para que a equipe de logística separe os produtos comprados e te envia um e-mail confirmando a compra.

O software que toma conta de tudo isso é complexo. Ele contém regras de negó- cio relacionadas ao carrinho de compras, ao pagamento, ao fechamento da compra. Mas, muito provavelmente, todo esse código não está implementado em apenas um único arquivo; esse sistema é composto por diversas pequenas classes, cada uma com sua tarefa específica.

Desenvolvedores, quando pensam em teste de software, geralmente imaginam um teste que cobre o sistema como um todo. Um teste de unidade não se preocupa com todo o sistema; ele está interessado apenas em saber se uma pequena parte do sistema funciona.

Um teste de unidade testa uma única unidade do nosso sistema. Geralmente, em sistemas orientados a objetos, essa unidade é a classe. Em nosso sistema de exemplo, muito provavelmente existem classes como “CarrinhoDeCompras”, “Pedido”, e assim por diante. A ideia é termos baterias de testes de unidade separadas para cada uma dessas classes; cada bateria preocupada apenas com a sua classe. (ANICHE, 2012, p. 5)

Figura 6: Clico dos testes no TDD



Fonte: Livro Test-Driven Development

* 1. Personas

Persona é a representação fictícia do seu cliente ideal. Ela é baseada em dados reais sobre comportamento e características demográficas dos seus clientes, assim como uma criação de suas histórias pessoais, motivações, objetivos, desafios e preocupações.

Uma boa definição de persona passa justamente pelo contato com o seu público-alvo, de modo que em uma rápida análise você possa identificar características comuns entre os potenciais compradores.

Se você possui uma base de clientes, esse será o lugar perfeito para começar suas investigações. Mesmo que você tenha perfis diferentes de pessoas ou empresas que consumiram seu produto, alguns deles tendem a exemplificar a sua persona.

Uma dica importante é focar tanto em clientes satisfeitos quanto insatisfeitos. Em ambos os casos, você certamente aprenderá algo sobre a percepção do seu produto e quais desafios seus clientes estão encarando. (SIQUEIRA, 2016)

* 1. Design de Interação

Por *design* de interação, entendemos o seguinte:

Design de produtos interativos que fornecem suporte às atividades cotidianas das pessoas, seja no lar ou no trabalho.

Especificamente, significa criar experiências que melhorem e estendam a maneira como as pessoas trabalham, se comunicam e interagem. Winograd (1997) descreve o *design* de interação como “o projeto de espaços para comunicação e interação humana”. Nesse sentido, consiste em encontrar maneiras de fornecer suporte às pessoas. Tal projeto contrasta com o da engenharia de *software*, que enfoca principalmente a produção de soluções de *software* para certas aplicações. Podemos fazer uma outra analogia simples com outra profissão, para tentar explicar melhor essa distinção.

Ao descrever o *design* de interação, Terry Winograd questiona em que medida arquitetos e engenheiros civis se diferenciam ao deparar-se com o problema de construir uma casa. Os arquitetos estão preocupados com as pessoas, suas interações e o interior da casa. Por exemplo, existe a proporção certa entre áreas privadas sociais? As áreas para cozinhar e fazer as refeições estão próximas? As pessoas utilizarão os espaços projetados da maneira como foram pensados? Em contrapartida, os engenheiros estão interessados em questões relacionadas com a realização do projeto, o que inclui aspectos práticos, como custo, durabilidade, aspectos estruturais, aspectos ambientais, regulamentações contra incêndio e métodos de construção. Assim como há uma diferença entre projetar e construir uma casa, há também uma diferença entre o *design* de interação e a engenharia de *software*. Resumindo, aquele está para esta como a arquitetura está para a engenharia civil. (Preece, J; Rogers, Y; Sharp, H. 2005, p. 28)

* 1. Framework Material Design

Material Design é um framework criado pelo Google que possui uma diversidade de componentes e plug-ins em JavaScript (Jquery) para auxiliar designers e desenvolvedores a implementar menus, modals, slideshow e outros objetos com efeitos visuais com facilidade. Utilizando o framework é possível inserir os itens acima inserindo apenas algumas linhas de código.

O Material Design foi apresentado pela Google em 2014 em um evento. Essa nova linguagem de design animou designers e desenvolvedores de todo o mundo com um conjunto de princípios e estilos mostrados de uma forma diferente e com novidades como interação em terceira dimensão. Eles foram inspirados em materiais e objetos do mundo real, reagindo na medida em que são manuseados. (CASTELLI, 2016)

* 1. Framework AngularJS

O AngularJS é um framework JavaScript MVC para Web que consideramos ser um super-herói. Nós o chamamos de super-herói porque o AngularJS faz tanto por nós que tudo o que devemos fazer é focar em nossa aplicação principal e deixar que ele cuide do resto. O AngularJS permite aplicar práticas-padrão de engenharia de software testadas e aprovadas, tradicionalmente utilizadas do lado do servidor, na programação do lado cliente para acelerar o desenvolvimento de frontends. Esse framework oferece uma estrutura consistente e escalável, que facilita desenvolver aplicações complexas e de grande porte como parte de uma equipe (SESHADRI; GREEN, 2014, p. 20).

* 1. Moodboard

O mood board apóia o designer através da articulação do pensamento imaginativo e do raciocínio por ANALOGIA, algo que ajuda na resolução de problemas complexos pela identificação de determinados aspectos através da articulação de aprendizados novos e antigos.

Essa ferramenta apresenta-se sob a forma de um quadro que combina uma série de referências visuais que apoiam a criação de uma ATMOSFERA do projeto, principalmente em suas etapas iniciais. A própria palavra inglesa “mood” ajuda na compreensão desse instrumento, podendo ser entendida como humor, atmosfera ou mesmo um estado temporário de nossa mente. (FACCA, 2012)

* 1. StyleGuide

Um dos elementos mais importantes do processo de criação de identidade visual é a elaboração de um documento com informações concretas sobre cores, tipografia, forma, exemplos de uso, tamanho mínimo e área de reserva do logotipo, etc. Este documento, conhecido como Guia de Identidade Visual ou simplesmente Manual da Marca, não se restringe apenas ao aspecto gráfico e pode muitas vezes incluir explicações sobre a visão, história e filosofia da empresa, uma perspectiva mais aprofundada dos símbolos e signos utilizados para compor o design ou até instruções gerais sobre a linguagem a ser utilizada na comunicação escrita, por exemplo. Seja um documento em PDF de apenas uma página com diretrizes simples do tipo “faça e não faça” ou um livro complexo com dezenas de exemplos de aplicação, o Guia de Identidade Visual é o que garante a consistência e uniformidade da percepção de uma marca. É o que vai fazer ela ser imediatamente reconhecida como uma entidade.

Mas estes guias não se restringem ao universo do design. E eles podem ser mais ou menos formais. Uma revista ou jornal possui um manual de estilo com padronizações de elementos como tamanho da lauda, ortografia, e estilo geral da redação.

Na moda e no design de interiores é muito comum o uso de mood boards, literalmente quadros de humor. Podem ser organizados nos mais diferentes meios, de pastas fichário com recortes e colagens, a quadro de cortiça com alfinites ou boards virtuais no Pinterest. Não importa muito o formato, a idéia aqui é reunir inspirações de cores, texturas, padrões, etc de maneira mais ou menos complexa para ajudar a compor o clima geral do design. Um mood board pode parecer confuso e bagunçado, mas é um guia de estilo a sua maneira.

Websites também podem possuir guias de estilo, mas, infelizmente, poucos profissionais incorporam está boa prática no workflow. (GUERRATO, 2013)

* 1. Story board

O storyboard em sua essência é basicamente um guia visual narrando as principais cenas de uma obra audiovisual. Os storyboards no geral são desenhos rápidos e com poucos detalhes, sendo o mais objetivo possível. Audiovisual é puro movimento, o storyboard é a chave de ignição deste movimento que é traçado por linhas e gestos no papel.

O principal objetivo é transpor as cenas do roteiro para quadros dinâmicos e de fácil visualização. Os desenhos por mais simples que sejam auxiliam a visualizar toda a dinâmica de movimento de câmera ao posicionamento de atores. Ou seja, o storyboard é como um mapa que auxilia desde o diretor, atores ao diretor de fotografia.

O storyboard é utilizado em qualquer mídia audiovisual como filmes, animações 2D ou 3D, videogames, seriados, novelas, comerciais, videoclipes e até em propaganda política. Os storyboards até podem ser utilizados no teatro para o diretor da peça compreender melhor o cenário como também o posicionamento dos atores no palco. (Thot, 2014)

1. A COTAÇÃO DE PRODUTOS EM ATACADISTAS
   1. Dificuldade em encontrar o melhor custo benefício

As redes atacadistas brasileiras atualmente não publicam em seus sites informações sobre todos os produtos disponíveis e seus respectivos valores. Em casos raros é possível encontrar apenas as promoções, sendo assim os clientes tem dificuldade na escolha dos estabelecimentos que oferecem o melhor custo benefício e ao mesmo tempo estão localizados próximo ao cliente.

Tarefas como cotação de preços entre diversos estabelecimentos atacadistas podem se tornar difíceis de serem realizadas. Isso porque o cliente tem que se locomover até os estabelecimentos atacadistas para pesquisar os melhores preços de produtos e muitas vezes nem todos os produtos são pesquisados devido ao tamanho físico dos estabelecimentos.

Em grandes cidades a locomoção entre os estabelecimentos atacadistas de diversas regiões é demorada devido ao trânsito diário, isso acarreta estresse para o cliente. Outro ponto negativo é o meio de transporte do cliente que muitas vezes não possui veículo próprio e depende de transporte público, sendo assim existe o custo do valor da passagem para executar uma simples cotação, ou se o cliente possui veículo ainda é necessário desembolsar valores com combustível e pedágio.

É quase impossível ter acesso as promoções semanais das redes atacadistas, pois em raros casos essas informações estão disponíveis no site ou em folders, existem muitos clientes que planejam suas compras baseadas em promoções para diminuir os custos. Essas informações disponibilizar em real-time poderiam melhorar o poder de decisão na aquisição dos produtos.

A diversidade de produtos comercializados por uma rede atacadista é muito grande, mesmo assim sempre existe a possibilidade de faltar uma marca ou tipo de produto, sem saber disso o cliente se locomove até a rede atacadista procurando por um produto que não é comercializado pelo atacadista. Se existisse uma listagem contendo todos os produtos da rede, o cliente poderia ser mais objetivo para decidir em qual estabelecimento deve comprar o produto, evitando perda de tempo na locomoção.

Quando o cliente está dentro do estabelecimento atacadista e decidi adquirir um determinado produto surge uma nova dificuldade, devido as grandes dimensões físicas dos estabelecimentos que comportam diversos corredores e prateleiras, fica difícil encontrar produtos específicos e muitas vezes o cliente acaba não encontrando o produto que precisa.

Produtos com pouco volume físico acabam ficando escondidos nas prateleiras entre caixas vazias e outras embalagens. Outra dificuldade é a grande quantidade de prateleiras existente no mesmo corredor contendo uma diversidade de produtos, que também provoca dúvidas no cliente.

* 1. Cotando os produtos em apenas um local

Analisando o contexto do problema seria viável uma solução computacional nas nuvens, contendo uma base de dados com informações sobre diversas redes atacadistas e seus respectivos produtos. Com essas informações centralizadas seria possível executar pesquisas e cotações de preços entre os produtos dos diversos atacadistas.

As redes atacadistas interessadas em fornecer os produtos para o banco de dados teriam que inicialmente se conveniar fornecendo dados da empresa e localização geográfica (latitude e longitude). As informações de localização de todas as redes atacadistas conveniadas também poderão ser consumidas posteriormente.

Essa solução precisa manter os dados atualizados dos produtos e também as promoções que cada rede atacadista disponibiliza semanalmente. As redes atacadistas conveniadas devem atualizar e disponibilizar os dados dos seus produtos e promoções para que a solução atualize diariamente a base de dados central.

As informações disponíveis no banco de dados poderão ser consumidas de maneira organizada e padronizada através da utilização de uma API (Application Programming Interface). Essa solução tem como objetivo disponibilizar recursos para serem consumidos via protocolo HTTP, retornando dados no formato JSON (JavaScript Object Notation) que podem ser consumidos por diversas plataformas de desenvolvimento.

O cliente só poderá visualizar as informações disponibilizadas pela API através de aplicações desenvolvidas por terceiros, contendo uma interface gráfica para interação do cliente. Para consumir os dados da API a aplicação deverá informar o tipo de recurso desejado e enviar os parâmetros necessários conforme padrão descrito na documentação.

A API deve encapsular todas as regras de negócio e práticas de segurança fornecendo apenas recursos para pesquisar localização das redes atacadistas, pesquisa e cotação de produtos, retornando dados formatados em real-time após requisição ser enviada pela aplicação desenvolvida por terceiros.

Com as informações centralizadas em uma base de dados única, as pesquisas de produtos e cotações de preços tendem a ser menos custosas, deixando o retorno dos dados mais ágeis para aplicações que estejam consumindo os recursos da API. As informações dos produtos sendo atualizadas diariamente e a API retornando os dados em tempo real o cliente tem maior poder de decisão para escolher o estabelecimento atacadista que possui o melhor custo benefício para aquisição dos produtos.







As redes atacadistas conveniadas que possuírem grande diversidade de produtos e preços competitivos podem se destacar nas pesquisas e cotações realizadas na API, os atacadistas que lançam promoções frequentes também podem aumentar o fluxo de consumidores.







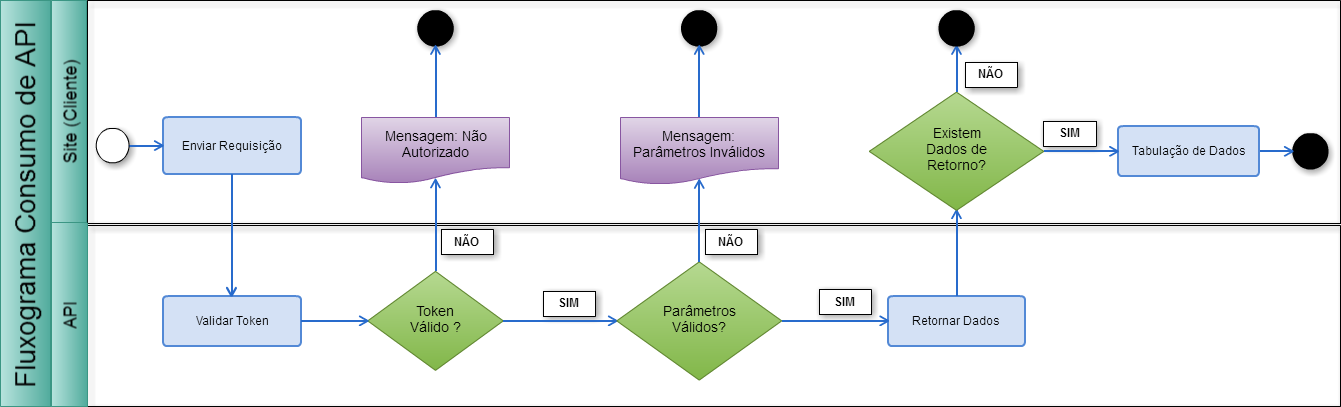
1. DESENVOLVIMENTO DA API PARA COTAÇÃO DOS PRODUTOS
   1. BPM (Business Process Model)

O Business Process Model demonstra o fluxo dos processos e validações envolvidas em todas as requisições que forem enviadas para API, como exemplo um site (cliente) está enviando uma requisição para API processar uma consulta.

Por segurança em todas as requisições para consumo da API deverá ser enviado o token para autenticação do cliente, essa prática é necessária pois o protocolo HTTP é stateless e não guarda estado ou session de autenticação. Caso o token não seja válido será retornado uma mensagem informando que o usuário é inválido e será finalizado o fluxo da requisição.

Após validar o token do cliente o processo seguinte é validar se a quantidade dos parâmetros e o formato dos valores enviados estão de acordo com o padrão esperado pela API, caso não seja validado será retornada uma mensagem informando o erro no formato dos parâmetros e finalizado o fluxo da requisição.

Se o token do cliente for válido e os parâmetros enviados forem validados, o próximo passo é verificar o recurso solicitado, executar a consulta e retornar os dados no formato JSON através da requisição HTTP.

Aplicação do cliente verifica se houve retorno de dados, caso exista conteúdo esses dados serão renderizados no dispositivo e finaliza o fluxo da requisição.

Fonte: Autoria Própria (2016)

Figura 7 - BPM (Business Process Model)

* 1. Requisitos
     1. Requisitos Funcionais

|  |  |
| --- | --- |
| **ID** | **Requisito Funcional** |
| RF1 | Eu como sistema devo Importar diariamente o XML com produtos dos Atacadistas para manter o banco de dados atualizado. |
| RF2 | Eu como API devo exibir todos os Atacadistas conveniados para que o usuário selecione em qual rede deseja executar pesquisa. |
| RF3 | Eu como API devo pesquisar os atacadistas que estão dentro dos valores de raio e geolocalização (latitude e longitude) enviados pelo usuário. |
| RF4 | Eu como API deve o exibir todas as promoções a partir do nome de um atacadista enviado pelo usuário. |
| RF5 | Eu como API devo executar cotação e exibir os produtos a partir de uma descrição parcial do produto enviado pelo usuário. |
| RF6 | Eu como API devo exibir todos os produtos a partir de um determinado corredor e nome do atacadista enviados pelo usuário. |
| RF7 | Eu como API devo pesquisar e exibir os produtos a partir do nome do atacadista e descrição parcial do produto. |

Tabela 1 - Requisitos Funcionais

* + 1. Requisitos Não Funcionais

|  |  |
| --- | --- |
| **ID** | **Requisito Não Funcional** |
| **RNF1** | Eu como API devo permitir acesso por tecnologias WEB. |
| **RNF2** | Eu como API devo validar todos os parâmetros enviados na requisição e emitir mensagem síncronas caso os parâmetros sejam inválidos. |
| **RNF3** | Eu como API devo ser capaz de se comunicar com API do Google Maps para trazer o Atacadista conforme localização enviada pelo cliente. |
| **RNF4** | Eu como API devo retornar mensagens síncronas para todas requisições enviadas, podendo ser de sucesso ou de advertência. |
| **RNF5** | Eu como API não posso permitir ultrapassar o limite de consultas diárias disponibilizadas para o cliente. |
| **RNF6** | Eu como API devo validar o token para autenticação do cliente que será enviado em todas as requisições. |
| **RNF7** | Eu como administrador devo disparar a geração de tokens através do sistema. |
| **RNF8** | Eu como API devo gravar log de todas as pesquisas executas. |
| **RNF9** | Eu como sistema devo disponibilizar relatórios com métricas dos produtos mais pesquisados por atacadista. |

Tabela 2 - Requisitos Não Funcionais

* 1. URIs (Recursos) da API

A API de cotação vai disponibilizar 6 recursos para serem consumidos pelas aplicações de terceiros, seguindo o modelo arquitetural REST todos os recursos estarão disponíveis em forma URIs.

Baseado nos recursos disponíveis na API a aplicação de terceiros vai enviar os parâmetros em determina ordem.

|  |  |
| --- | --- |
| **ID** | **URI** |
| **R1** | http://api.cotacao.com.br/atacadistas/ |
| **R2** | http://api.cotacao.com.br/atacadistas/{raio}/{latitude}/{longitude} |
| **R3** | http://api.cotacao.com.br/atacadista/promocao/{atacadista} |
| **R4** | http://api.cotacao.com.br/atacadista/produto/{atacadista}/{produto} |
| **R5** | http://api.cotacao.com.br/atacadista/cotacao/{produto} |
| **R6** | http://api.cotacao.com.br/atacadista/produto/localizacao/{atacadista}/{corredor} |

Tabela 3 - Recursos disponibilizados pela API

* + 1. Descrição dos Parâmetros

|  |  |
| --- | --- |
| **ID** | **Descrição dos Parâmetros** |
| **R1** | Não é necessário informar parâmetros |
| **R2** | Raio: Valor numérico, informa distância de um ponto central até extremidade de uma circunferência.  Latitude: Valor numérico, coordenada geográfica.  Longitude: Valor numérico, coordenada geográfica. |
| **R3** | Atacadista: Valor alfanumérico, nome do estabelecimento atacadista. |
| **R4** | Atacadista: Valor alfanumérico, nome do estabelecimento atacadista.  Produto: Valor alfanumérico, descrição parcial do produto. |
| **R5** | Produto: Valor alfanumérico, descrição parcial do produto. |
| **R6** | Atacadista: Valor alfanumérico, nome do estabelecimento atacadista.  Corredor: Valor alfanumérico, informa a identificação do corredor. |

Tabela 4 - Descrição dos parâmetros das URIs

* 1. MER (Modelo Entidade Relacional)

No modelo de entidade relacional constam as entidades envolvidas no processo de cadastro dos estabelecimentos atacadistas, importação e atualização dos produtos.

Todos os estabelecimentos atacadistas que forem participar da cotação de produtos deveram ser cadastrados previamente na entidade “ATACADISTA”, após esse processo o estabelecimento atacadista informa a periodicidade das atualizações em seu arquivo XML. Com essa informação será cadastrado na entidade “AGENDAMENTO” a periodicidade com que o sistema de agendamento da API vai importar e atualizar as informações do XML.

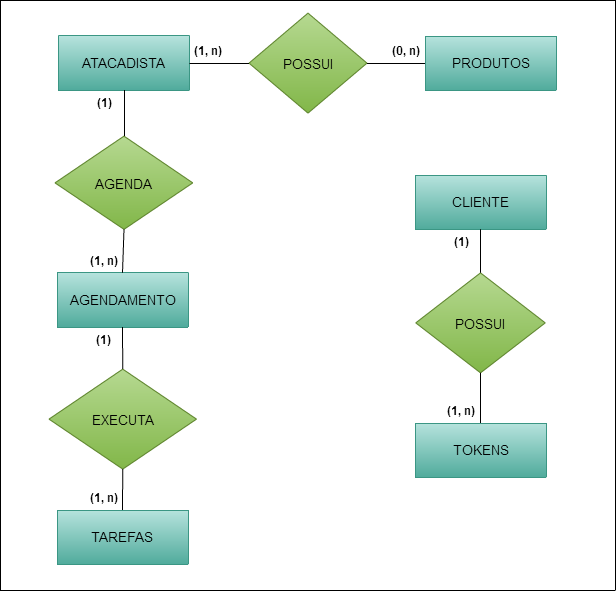
Para monitorar os agendamentos diariamente será escrito um sistema em Python que irá rodar como serviço 24 horas por dia e 7 dias por semana (24x7), quando for detectado a necessidade de importação será cadastrada uma tarefa na entidade “TAREFA”, esse registro vai conter todas as informações da importação em andamento.

Todos os produtos importados a partir dos XMLs disponibilizados pelos estabelecimentos atacadistas serão gravados na entidade “PRODUTO”, essa entidade vai conter também os valores de promoção dos produtos, quando existir.

As aplicações de terceiros que tiverem interesse em consumir os dados da API, deverão ser cadastradas previamente na entidade “CLIENTE”. Após o cadastro será gerado um token de segurança gravado na entidade “TOKEN” e enviado para o cliente. Esse token será atualizado e enviado para o cliente mediante periocidade descrita nas medidas de segurança.

Sempre que uma aplicação de terceiros for consumir os dados da API, no “header” da requisição HTTP deverá ser enviado o token de autorização para validação na API.

Figura 8 - MER (Modelo Entidade Relacional)



Fonte: Autoria Própria (2016)

* 1. Big Picture (Arquitetura da Solução)

Para infraestrutura da API será utilizado um serviço Cloud rodando o sistema operacional Linux (Ubuntu), com banco de dados MySQL, servidor de aplicações Apache, Python e PHP.

Todo o tráfego das informações em XML entre o estabelecimento atacadista e o servidor será sob o protocolo HTTP, através da URL informada pelo atacadista. A API será desenvolvida sob o modelo arquitetural REST, dessa maneira todas as informações também serão trafegadas sob o protocolo HTTP, fazendo uso dos verbos,

A escolha de um serviço Cloud é baseada na facilidade em escalabilidade dos recursos do servidor conforme demanda, pois recursos como memória RAM, HD e processador podem receber um upgrade em pouco tempo, reduzindo a quantidade de horas que a aplicação teria que ficar fora do ar.

O sistema operacional Ubuntu possui excelente estabilidade e segurança, além de receber atualizações em seus pacotes constantemente, principalmente para versões LTS. Como as distribuições Linux, o Ubuntu possui uma gama de ferramentas para monitoramento dos recursos do servidor.

Para monitorar os agendamentos para importação dos XMLs de produtos será utilizado uma aplicação escrita na linguagem Python, pois a mesma deverá rodar em backgroud no servidor, essa característica impossibilita usar uma aplicação escrita em PHP para monitorar os agendamentos.

A API será construída utilizando a linguagem PHP, pois possui boa aderência com o banco de dados MySQL e também possui funções nativas para trabalhar com o formato JSON, esse formato será utilizado para troca de informações entre aplicações de terceiros e a API.

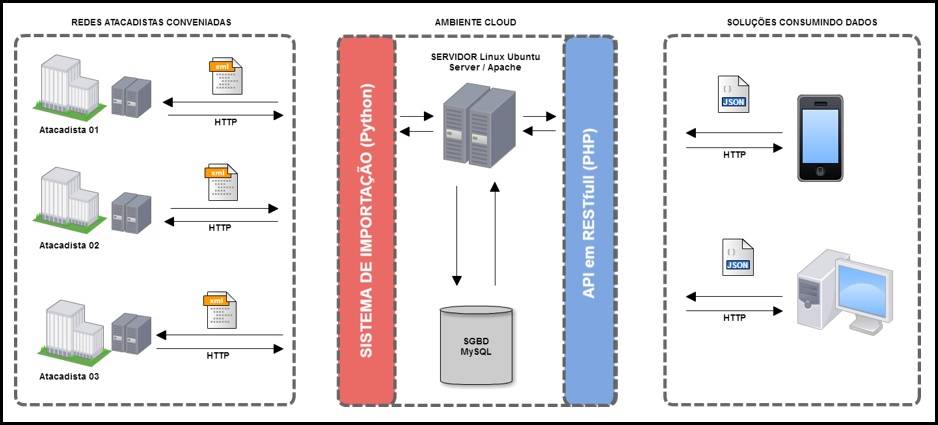


Figura 9 - Big Picture (Arquitetura)

FFonte: Autoria Própria (2016)

* 1. Plano de Teste
     1. Testes URI

Após a construção do contrato inicial na ferramenta Swagger, todas as rotas serão geradas na API. Para validar o status code 200 das requisições nas URIs será utilizado a ferramenta SoapUI.

* + 1. TDD

Serão realizados testes unitários nas classes de acesso a dados para validar se os dados retornados do banco estão de acordo com os parâmetros enviados durante o teste.

* + 1. Consumo da API

Utilizando a ferramenta SoapUI serão enviadas requisições com os devidos parâmetros para validar se os dados retornados estão no formato JSON correto.

* 1. Desenvolvimento do Protótipo
     1. Personas

Pedro, Bruno e Ana… pessoas com características diferentes, porém com interesses em comum. Pedro e Bruno são empresários, um é chefe de cozinha e o outro proprietário de um Food Truck, já Ana é uma esposa amorosa, mãe e dona de casa. Todos possuem interesse em tecnologia e acompanham notícias de novos produtos recém lançados no mercado.

Para compreender melhor os padrões das personas supracitadas, foram realizadas perguntas para identificar hábitos e comportamentos determinantes para o desenvolvimento do projeto. As perguntas foram divididas em categorias, são elas: Interesse, personalidade, habilidades, sonhos, relação com a tecnologia, razões para usar e comprar produtos e serviços.

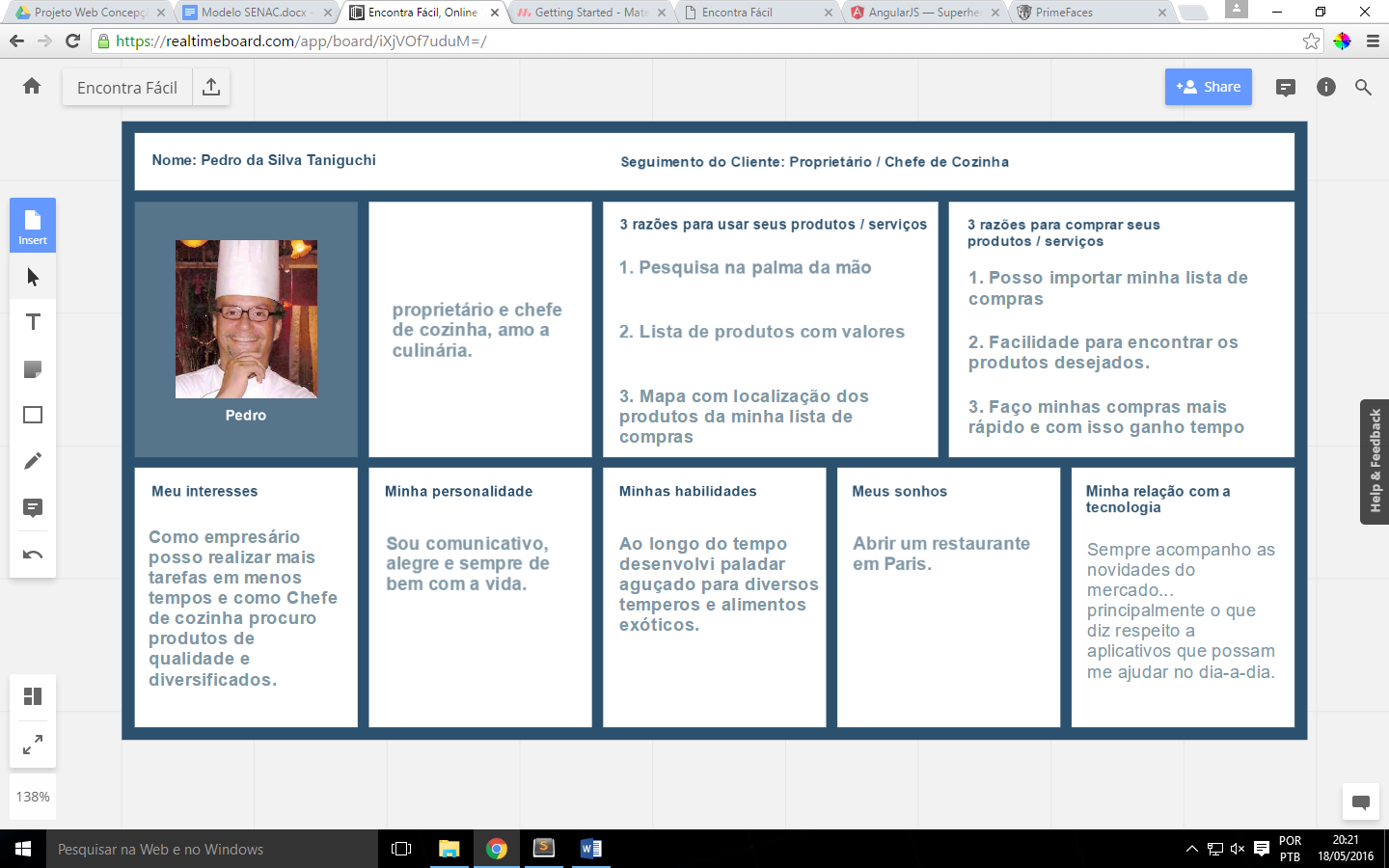
Com base na análise dos dados obtidos, encontramos características em comuns entre as personas no que diz respeito à problemas, reclamações e possíveis soluções.

Ficou explicito que, todos fazem uso e procuram obter vantagens com a internet no dia a dia. As personas também relataram que uma das atividades mais corriqueira e trabalhosa é a de realizar compras. Todos relatam sobre a dificuldade de encontrar por determinados produtos e se agrava quando se trata de promoções.

Quando perguntado se já haviam se deparado com um problema similar e qual foi a solução encontrada, uma das personas relatou que ao planejar suas férias procurava por hotéis nas cidades onde pretendia visitar e graças a aplicação Web Trivago a busca se tornou ágil.

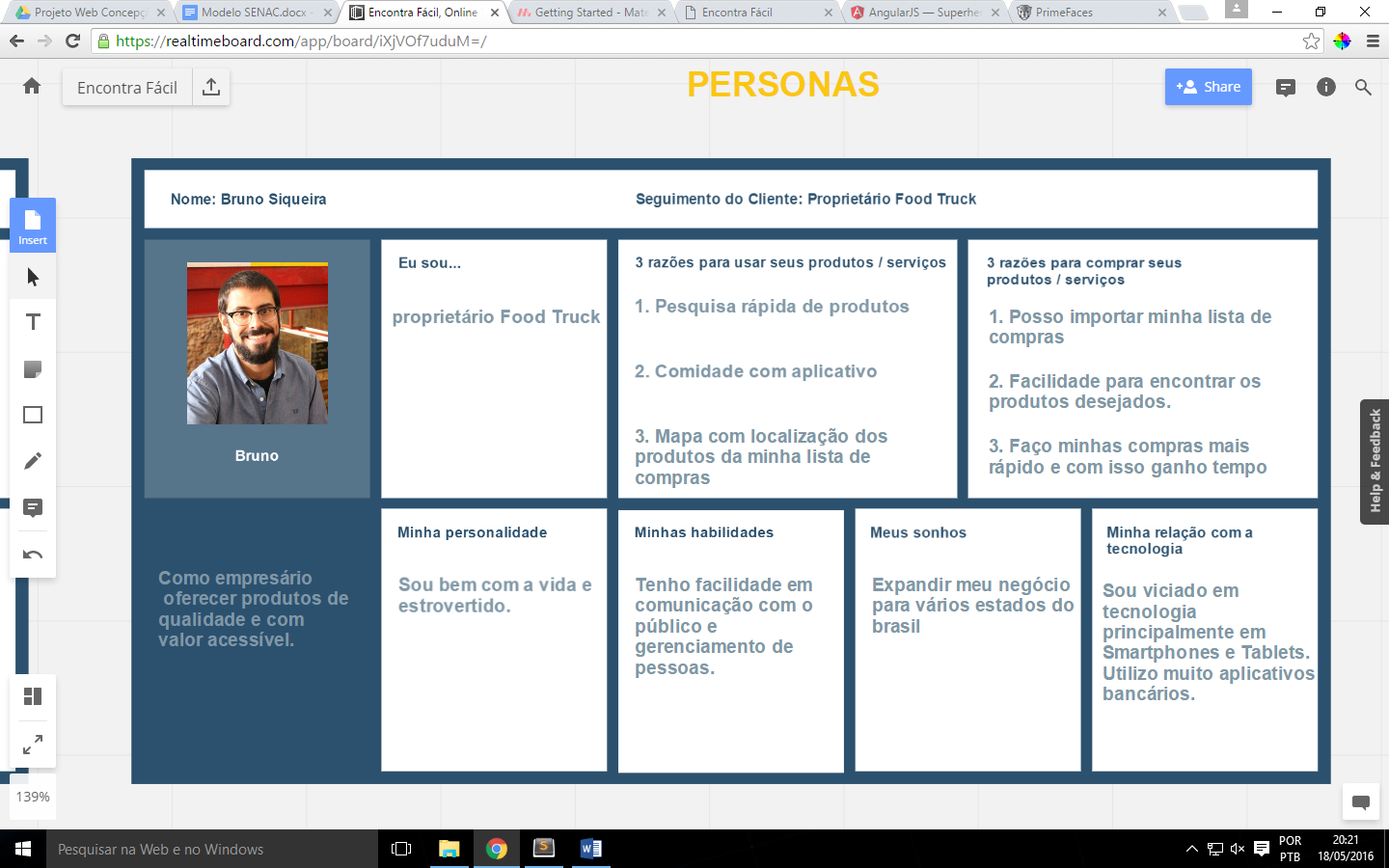
Com base no relato das personas e se os mesmos gostariam de ver algo no mercado que facilitasse a busca por produtos e promoções utilizando a internet como ferramenta o desenvolvimento de uma API para cotação se tornou uma oportunidade.

Figura 10 - Persona (Pedro da Silva)



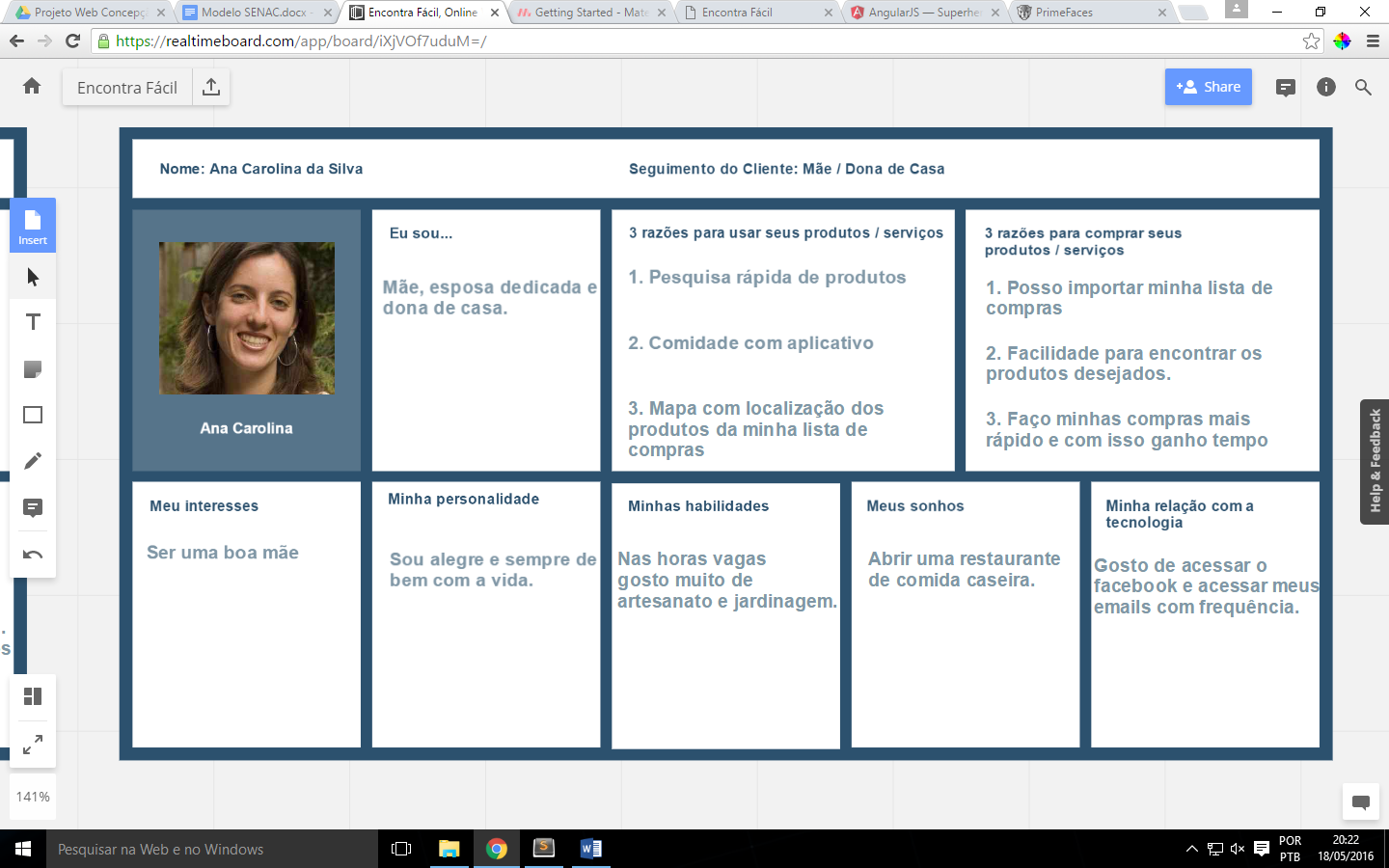
Fonte: Autoria Própria (2016)

Figura 11 - Persona (Bruno Siqueira)



Fonte: Autoria Própria (2016)

Figura 12 - Persona (Ana Carolina)



Fonte: Autoria Própria (2016)

* + 1. Design de Interação

A chave para o sucesso de um bom design de interação é focar no usuário, entendê-lo e questionando suas escolhas. Para o desenvolvimento do protótipo foi observado as necessidades dos clientes como, por exemplo, a escolha do estabelecimento da rede atacadista, bem como, pesquisa e cotação de produtos, consideração o melhor preço e benefício.

Uma vez que o entendimento do contexto tornou-se claro, foi possível o desenvolvimento de protótipos, com a finalidade de separar e tornar tangível diversas soluções.

Trabalhando de maneira colaborativa e utilizando vários recursos tecnológicos, como por exemplo, Google Materials, estudo de cores e layout, tornou-se possível o elo entre os usuários e a aplicação.

* + 1. Moodboard

Com base nas informações levantadas ao longo da pesquisa, foi sendo construído um painel semântico, com o objetivo de combinar uma série de referências visuais visando criar uma atmosfera para o projeto e encontrar uma maneira criativa e inovadora para solucionar o problema.

Ao término da pesquisa, o próximo passo foi a organização das informações em três grupos principais: objetos, pessoas e ambientes.

Uma vez que o painel foi concluído ficou nítido o que esperar do projeto no que diz respeito ao uso das cores, formas, texturas, conceitos, cenários e etc.

Essas representações visuais apresentam as emoções e inspirações necessárias para o desenvolvimento do trabalho.

Figura 13 – Moodboard



Fonte: Autoria Própria (2016)

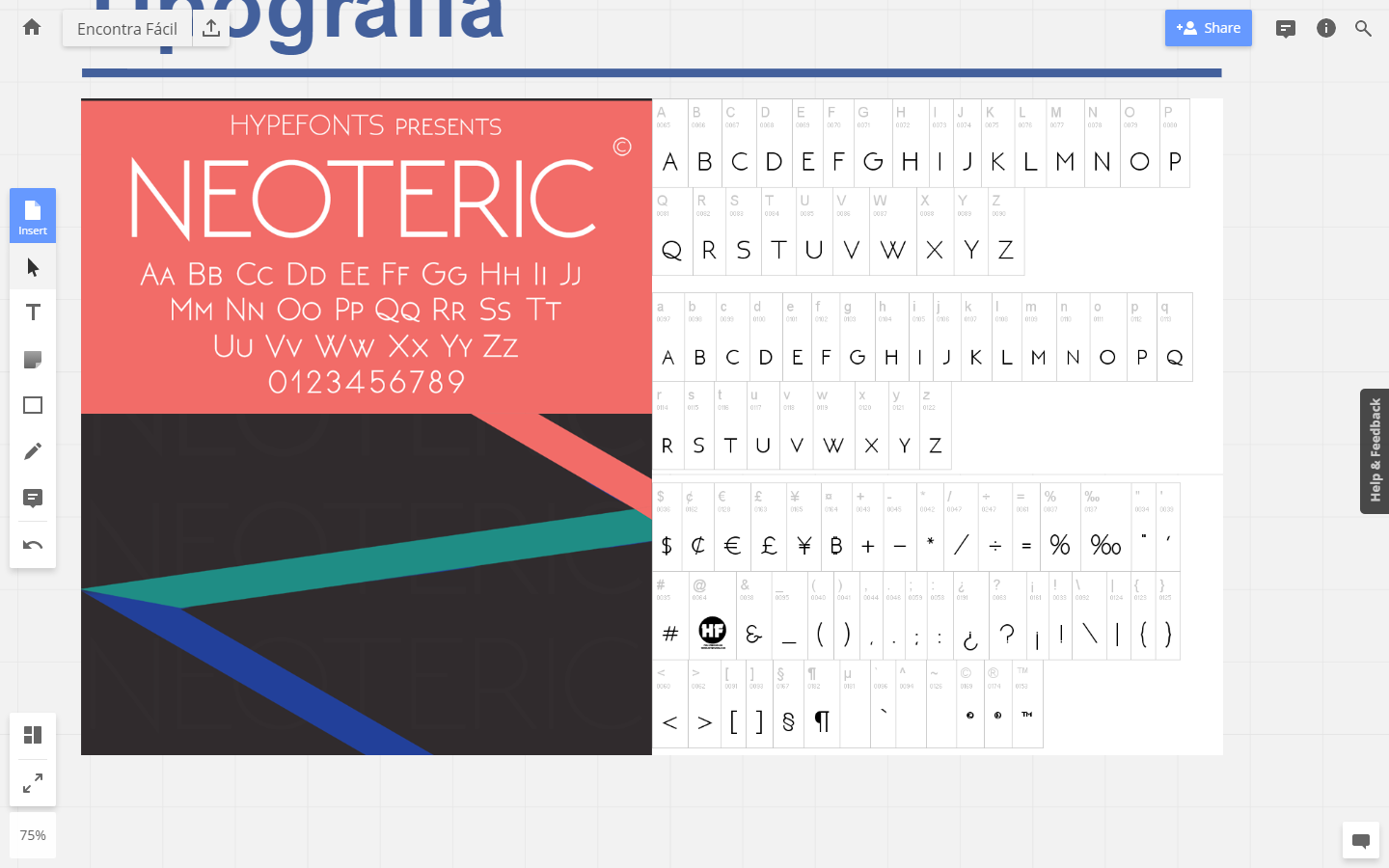
* + 1. Style Guide

O desenvolvimento do Style Guide permitiu reunir em uma única documentação informações importantes para compor a aplicação, levando em consideração a harmonia entre tipografia, cores e ícones.

* + 1. Tipografia

As fontes Sans Serif (sans, em francês, significa “sem”) possuem uma aparência simples e mais “limpa”, ou seja, mais fáceis de serem lidas no meio digital. A NEOTERIC foi escolhida como fonte para compor a aplicação web, por ser uma fonte básica e simples, além de possuir variações para compor títulos.

Figura 14 - Style Guide (Tipografia)



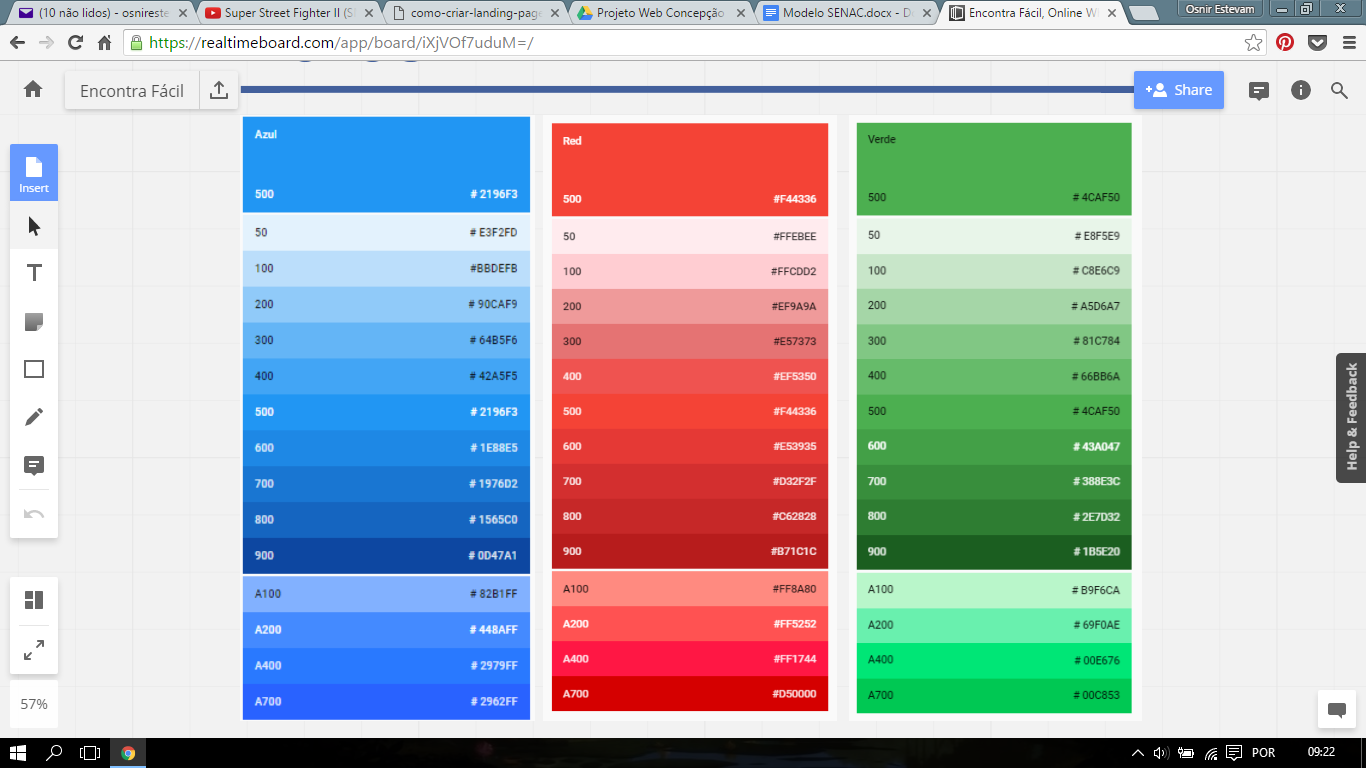
Fonte: Autoria Própria (2016)

* + 1. Cores

No que tange as cores foi realizado um estudo com as cores triádicas. Esse estudo utiliza as três cores da Roda de cores, igualmente espaçadas (120º uma da outra), em torna da roda de cores.

Esse método é popular porque oferece um forte contraste visual e é considerado por muitos como o melhor dos esquemas. Nesse aspecto as cores escolhidas foram, azul por representar calma, confiança, segurança, aumento da criatividade, contemplação, espiritualidade, bem como, cria um senso de segurança e promove confiança na marca; o vermelho por ser ativa, apaixonante e emocionante, estimula a energia; e finalmente o verde por ser uma cor equilibrada e rejuvenescedora, representa estabilidade e possibilidade, crescimento, vitalidade, abundância e natureza.

Figura 15 - Style Guide (Cores)



Fonte: Autoria Própria (2016)

* + 1. Ícones

O estudo da Gestalt foi fundamental para a escolha dos ícones, uma vez que, tal estudo visa ajudar na construção de um produto sólido e de fácil utilização. Um dos requisitos foi o reconhecimento imediato dos ícones por parte do usuário.

Ao final do trabalho foi possível definirmos os ícones com melhor coerência, consistência, clareza e significativo, além de possuir tamanho econômico em relação ao espaço da tela.

Figura 16 - Style Guide (Ícones)

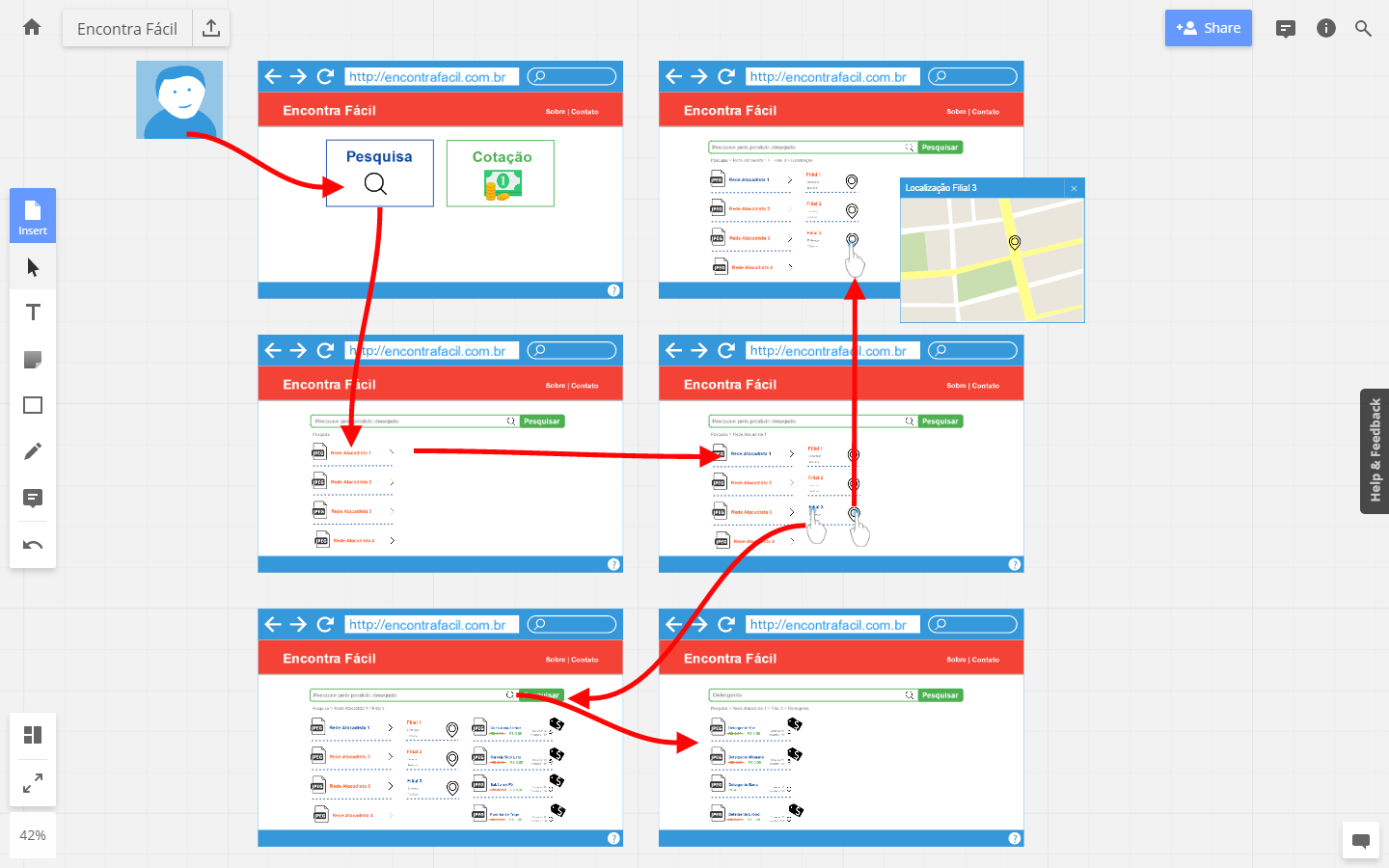


Fonte: Autoria Própria (2016)

* + 1. Story board

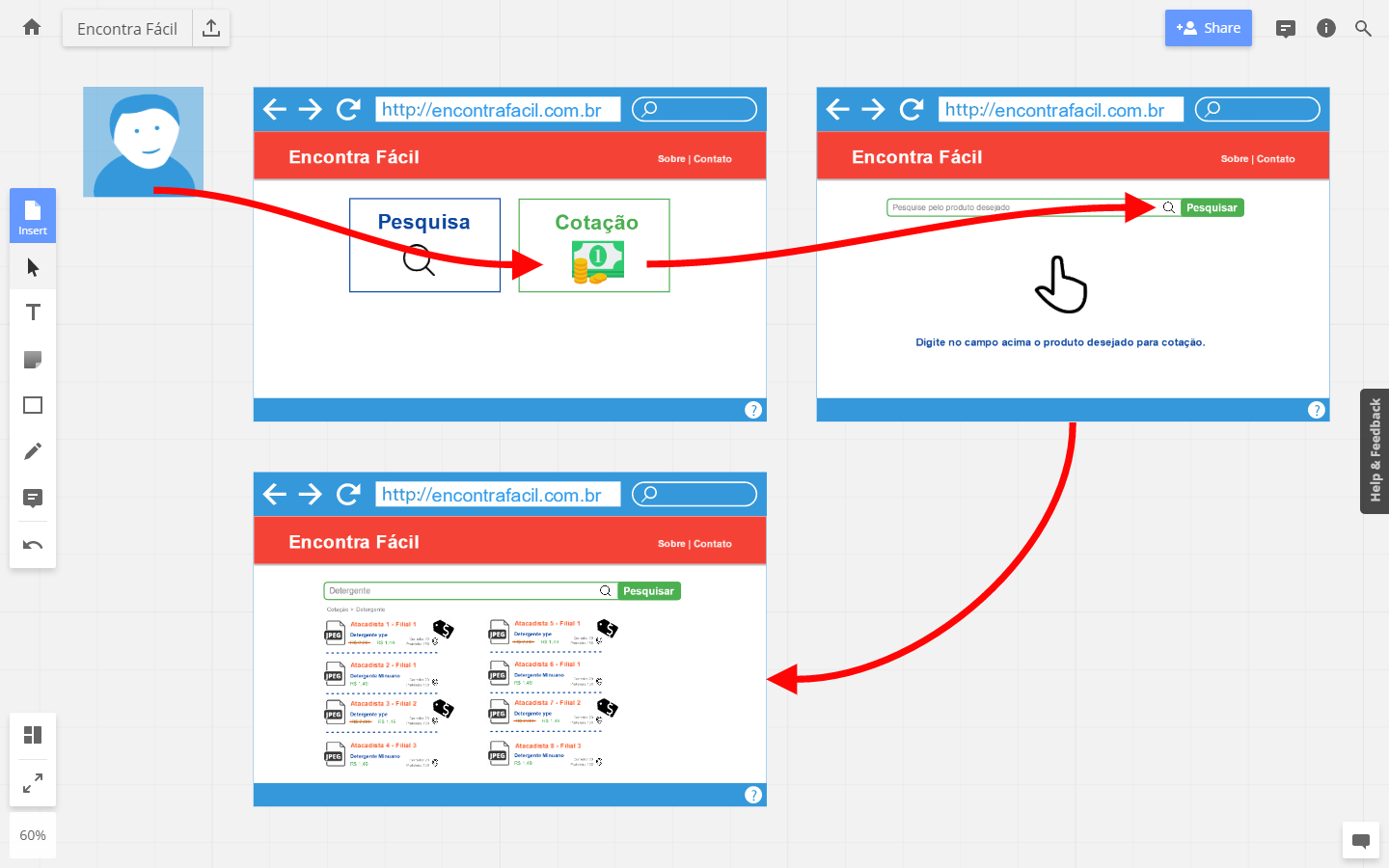
O Story board proporcionou uma maior visibilidade do projeto por meio de imagens dos elementos interativos. Durante o desenvolvimento foi percebido nuances na sequência da interação e, posteriormente a correção mais adequada possível. O Story board da aplicação é constituída por cinco telas a partir do botão “pesquisa” e duas telas a partir do botão “cotação” ambos os botões localizados na home da aplicação.

Figura 17 - Story board (Pesquisa)



Fonte: Autoria Própria (2016)

Figura 18 - Story board (Cotação)



Fonte: Autoria Própria (2016)

1. METODOLOGIA DE PESQUISA
   * 1. Instrumentos de Pesquisa
        1. Mercado

Foram realizadas pesquisas em lojas de aplicativos e na internet a procura de soluções com propostas semelhantes a solução apresentada.

Serão analisar as redes atacadistas a nível de pesquisa de mercado para conhecer o fluxo de dados, rotina de liberação das promoções, arranjo dos produtos nas prateleiras e também o perfil do público alvo.

Entrevista com clientes proprietários de comercio (Personas). Colhendo a opinião dos clientes sobre a solução aqui apresentada como ferramenta.

* + - 1. Cientifico

Pesquisas de monografias sobre o desenvolvimento de aplicativos híbridos, usabilidade, disponibilização de APIs.

* + 1. Coleta de Dados

Levantamento de requisitos da solução, detalhando requisito funcionais, não funcionais e transacionais, através de diagramas de caso de uso.

Aplicar técnicas de design utilizando Google Material Design, utilizando boas práticas de cores e posicionamento de elementos para o desenvolvimento do layout.

O aplicativo será desenvolvido de forma hibrida, utilizando o framework Cordova e para interface o framework Ionic, ambos com suporte a desenvolvimento com HTML, CSS e JavaScript.

Os dados serão importados em base de dados na WEB e posteriormente serão consumidos através de API desenvolvida com PHP e o framework Swagger.

Serão realizados testes usabilidade em diferentes modelos de dispositivos móveis e também testes com usuários para detecção a experiência com a interface.

Para o desenvolvimento da API será utilizado a metodologia TDD (Test Driven Development) para realizar testes nos módulos de importação de dados.

* + 1. Resultados Esperados

Centralizando as informações em uma base de dados única, espera-se que as pesquisas de produtos e cotações de preços obtenham um ganho de performance no tempo de resposta das requisições, deixando mais rápido o retorno dos dados para aplicações que estejam consumindo os recursos da API.

O cliente poderá aumentar seu poder de decisão no momento da pesquisa, pois irá possuir em tempo real as informações sobre o melhor custo benefício para aquisições dos produtos. Para que seja possível uma boa qualidade nas informações disponibilizadas pela API é importante que as informações dos produtos sejam atualizadas diariamente.

A pesquisa e cotação de produtos realizadas pela API pode estimular a concorrência entre as redes atacadistas e com isso favorecer o consumidor final. Porém as redes atacadistas que não atualizarem os dados de seus produtos e promoções frequentemente podem perder consumidores que usem como base de pesquisa os dados fornecidos pela API.

As redes atacadistas conveniadas que possuírem grande diversidade de produtos, lançarem promoções com frequência e possuírem preços competitivos em relação aos seus concorrentes, podem se destacar nas pesquisas e cotações realizadas na API e com isso aumentarem a quantidade de suas vendas.

1. CONSIDERAÇÕES FINAIS

O seguimento atacadista no Brasil é deficitário na disponibilização de informações sobre seus produtos e promoções, não existe uma fonte de dados acessível para que os consumidores consigam identificar os melhores preços.

A API foi desenvolvida para suprir essa falta de informação no site das redes atacadistas que não publicam uma listagem com os produtos que são comercializados e também a falta de informações sobre promoções que são lançadas semanalmente.

Os recursos que serão consumidos pela API serão importantes para levar os melhores preços entre a rede atacadista e o consumidor final, auxiliando na propagação das promoções e das cotações de produtos.

Para sucesso da API é importante que as aplicações desenvolvidas por terceiros possuam boa usabilidade e interface amigável, deixando o cliente confortável para navegar e consumir os dados. Essas aplicações também devem ser desenvolvidas de maneira que suas funcionalidades sejam ágeis para enviar as requisições e também renderizar os dados recebidos da API.

Para validar os recursos e segurança da solução proposta foi desenvolvido um WEB Site como protótipo para simular uma aplicação que poderá ser desenvolvida por terceiros. Esse protótipo foi elaborado usando boas práticas de usabilidade e consome todos os recursos disponibilizados pela solução.

Como sugestão para trabalhos futuros poderá ser adicionado a funcionalidade de geo-localização de produtos indoor (internamente) nos estabelecimentos atacadistas conveniados, pois é possível identificar corredor e prateleira dos produtos.

Poderá ser acrescentado também um recurso para o usuário importar uma lista de produtos e a API retornará uma lista de produtos apresentando o melhor valor entre as redes atacadistas conveniadas, indicando o endereço das mesmas

O consumo de dados padronizado produzindo interoperabilidade entre plataformas pode ser reaproveitado para vários seguimentos do comércio. O conceito de centralização de dados promocionais, listagens de produtos e disponibilização desses recursos em forma de API se aplica amplamente para várias necessidades de negócio.

.

REFERÊNCIAS BIBLIOGRÁFICAS

DEFINING WELL-KNOWN UNIFORM RESOURCE IDENTIFIERS (URIS). **Hypertext transfer protocol -- http/1.1**. **Especificação do protocolo http/1.1 RFC: 2616. Este documento especifica um protocolo, normas e pistas para a Comunidade da Internet.** Disponível em: <https://tools.ietf.org/html/rfc2616>. Acessado em: 23 mar. 2016.

DEFINING WELL-KNOWN UNIFORM RESOURCE IDENTIFIERS (URIS). **Especificação do protocolo http/1.1 RFC: 2616/5785.** **Este memorando define um prefixo de caminho para "locais bem conhecidos", "/.well-known/", no Uniform Resource Identifier selecionado (URI).** Disponível em: <https://tools.ietf.org/html/rfc5785>. Acesso em: 23 mar. 2016.

USE OF THE CONTENT-DISPOSITION HEADER FIELD IN THE. **Especificação do protocolo http/1.1 RFC: 2616/6266. Define a resposta campo de cabeçalho Content-Disposition, mas salienta que não é parte do HTTP / 1.1 padrão. Este especificação assume a definição e registro de Content- Disposition, como usado no HTTP, e esclarece aspectos de internacionalização.** Disponível em: <https://tools.ietf.org/html/rfc6266>. Acesso em: 23 mar. 2016.

ADDITIONAL HTTP STATUS CODES. **Especificação do Protocolo HTTP/1.1 RFC: 2616/6585. Este documento especifica adicional HyperText Transfer Protocol (HTTP) códigos de status para uma variedade de situações comuns.** Disponível em: <https://tools.ietf.org/html/rfc6585>. Acesso em: 23 mar. 2016.

OPENAPI SPECIFICATION. **A especificação Swagger define um conjunto de arquivos necessários para descrever (API).** Disponível em: <http://swagger.io/specification/>. Acessado em: 29 mar. 2016.

FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures.** California: University of California, Irvine, 2000.

T. A. Guedes, Gilleanes. **UML Uma Abordagem Prática.** São Paulo: Novatec, 2ª Ed 2011.

SAUDATE, A. **SOA aplicado Integrando com WebServices e além.** São Paulo: Casa do Código, 2012.

SANDERS, W. **Padrões de Projeto em PHP.** São Paulo: Novatec,2013.

MITCHELL, L. J. **PHP WEB Services.** Sebastopol: O’ Reilly Media, 2014.

SESHADRI, S. e Green, B. **Desenvolvendo com AngulaJS.** São Paulo: Novatec, 2014.

HEUSER, C. A. **Projeto de Banco de Dados.** Rio Grande do Sul: Sagra-luzzatto, 1998.

PREECE, J; Rogers, Y; Sharp, H. **Design de Interação: Além da Interação Homem-Computador.** Porto Alegre: Bookman, 2005.

MATERIAL DESIGN. **Um olhar aprofundado sobre o novo estilo visual da Google.** Disponível em: <http://www.tecmundo.com.br/google/58278-material-design-olhar-aprofundado-novo-estilo-visual-google.htm>. Acesso em: 15 jun. 2016.

APPLICATION PROGRAMMING INTERFACE. **Em essência, API de um programa define a maneira correta para um desenvolvedor para solicitar serviços a partir desse programa**. Disponível em: <http://www.computerworld.com/article/2593623/app-development/application-programming-interface.html>. Acesso em: 22 jun. 2016.

GUIA DE ESTILOS. **Apresentação de elementos importantes na elaboração de um documento de Guia de Estilo como, por exemplo, cores, tipografia, forma, exemplos de uso e tamanho mínimo.** Disponível em: <http://tableless.com.br/guia-de-estilos/>. Acesso em: 22 jun. 2016.

OBJECT MANAGEMENT GROUP BUSINESS PROCESS MODEL AND NOTATION. **BPMN - fornece às empresas a capacidade de compreender os seus procedimentos internos de negócios em uma notação gráfica e da as organizações a capacidade de comunicar procedimentos de uma forma padrão.** Disponível em: <http://www.bpmn.org/>. Acesso em: 22 jun. 2016.

MOOD BOARD. **Passa a passo do desenvolvimento de um Painel Semântico.** Disponível em: <http://chocoladesign.com/como-criar-um-painel-semantico-ou-mood-board>. Acesso em: 22 jun. 2016.

STORY BOARD. **O que é um storyboard.** Disponível em: <http://modelosdestoryboards.blogspot.com.br/p/o-que-e-um-storyboad.html>. Acesso em: 22 jun. 2016.

PERSONAS. **Persona: Como e por que criar uma para sua empresa.** Disponível em: <http://resultadosdigitais.com.br/blog/persona-o-que-e/>. Acesso em: 22 jun. 2016.

SWAGGER. **Site oficial Swagger**. Disponível em: <<http://swagger.io/>>. Acesso em 15 de mar. 2016.

RESTFUL. **Steps toward the glory of REST**. Disponível em: <<http://martinfowler.com/articles/richardsonMaturityModel.html>>. Acesso em: 20 mai. 2016.

ANICHE, Mauricio. **Test Driven-Develpment**. São Paulo: Casa do Código, 2012.