**Documento de Visão de Projeto**



**DrinkIt**

**Histórico de Versões**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** | **Revisor** |
| 05/03/11 | 1.0 | Modelagem e desenvolvimento | Gabriel Lima Gomes e Gustavo da Rosa | - |

|  |  |
| --- | --- |
| **Cliente** | FATEC - Interno |
| **Documento** | Documento de Visão de Projeto: *DrinkIt* |
| **Data** | 30 de setembro de 2019 |
| **Autor** | **Gabriel Lima Gomes**  alexmarcolongo@gmail.com |

**Página de Assinaturas**

|  |  |  |
| --- | --- | --- |
| Revisado e Aprovado por: |  |  |
|  |  | 27/11/10 |

1. Objetivo

Este documento trata principalmente da especificação das necessidades de negócios, da justificativa do projeto, do entendimento atual das necessidades do cliente e descreve resumidamente o novo produto, serviço ou resultado que deve satisfazer esses requisitos.

Tem o objetivo de alinhar as expectativas dos interessados para formalizar o início do projeto. Apresentar uma visão arquitetural do sistema **DrinkIt**. O intuito é salientar diferentes aspectos deste produto, obtidos a partir de decisões arquiteturais realizadas no âmbito do sistema **DrinkIt**.

* 1. Escopo

Esse sistema e-commerce, tem como objetivo permitir que pessoas possam realizar compras de bebidas por meio da internet, além de abranger um módulo administrativo para controle da gestão das bebidas, dashboards interativos, gerenciamento de trocas e cupons promocionais.

* 1. Referências

Para a construção deste documento foram utilizadas as seguintes referências:

* + Documento de requisitos *DRS\_LES\_2\_2019* – Documento de especificação de requisitos funcionais, não funcionais e regras de negócios;
  + Documento de Especificação de Casos de Uso
  + E-mail explicativo *LES\_SEGUNDA\_1\_ENTREGA* dos Sr.’s Gabriel Lima Gomes e Gustavo da Rosa, para com o Sr. Rodrigo Rocha Silva;

Este documento influencia os seguintes documentos:

* Proposta Técnica Comercial – DrinkIt

Necessidades de Negócio

É necessário um sistema e-commerce que possibilite, a qualquer tipo de cliente, a compra de bebidas pela internet, disponibilizando informações detalhadas de cada bebida, possibilitando também a troca de produto caso haja alguma irregularidade. É de relevância que existe um perfil administrativo para realizar a gestão completa do sistema, além de possibilitar gerar cupons promocionais para os clientes.

Objetivo do Projeto

Desenvolver uma plataforma e-commerce capaz de:

* armazenar informações em uma base de dados
* utilizar o protocolo HTTP
* autenticar usuários
* ser executado no navegador *Google Chrome*

Declaração Preliminar de Escopo

* 1. Descrição

O projeto consiste em uma plataforma e-commerce, que tem como foco ser um facilitador da venda de bebidas para pessoas por meio da internet.

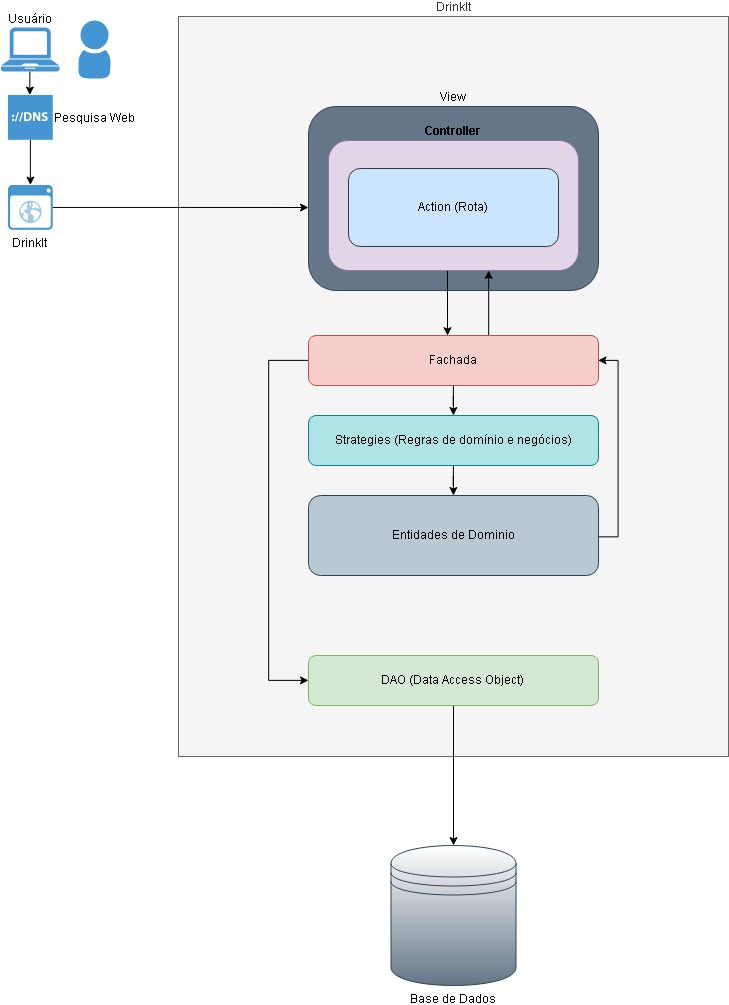
* 1. Produtos a serem entregues
* Sistema e-commerce *DrinkIt*, implementado de acordo com a especificação feita na fase de análise.
* Documentos de especificação do sistema, concebido na fase de elaboração.
* Hospedagem do sistema em ambiente 24 x 7.
  1. Requisitos
     1. Requisitos Funcionais
* O sistema deve manter um cadastro único para bebidas.
* O sistema deve possibilitar que bebidas sejam inativadas.
* O sistema deve possibilitar a alteração de dados cadastrais para as bebidas.
* O sistema deve possibilitar que uma bebida seja consultada com base em um filtro definido pelo usuário. Todos os campos utilizados para identificação da bebida podem ser utilizados como filtro, tanto de forma combinada como de forma isolada.
* Deve ser possível ativar o cadastro de um livro.
* Grupo: Cadastro de Clientes
* O sistema deve possibilitar o cadastro de clientes.
* O sistema deve possibilitar a alteração de dados cadastrais de clientes.
* O sistema deve possibilitar que clientes sejam inativados.
* O sistema deve possibilitar que um cliente seja autenticado por meio do e-mail e senha cadastrados previamente.
* O sistema deve disponibilizar no cadastro de clientes a consulta de todas transações já realizadas pelo mesmo.
* Deve ser possível associar diversos endereços de entrega ao cadastro de um cliente. Cada cadastro de endereço deve ser identificado com um nome composto de uma frase curta, chamado de “Descrição”.
* Deve ser possível associar diversos cartões de crédito ao cadastro de um cliente. Deve haver um cartão de crédito configurado como preferencial.
* O sistema deve possibilitar que a senha do usuário seja alterada sem que seja necessária a alteração de todos os dados cadastrais.
* Grupo: Gerenciar Vendas Eletrônicas
* O sistema deve permitir que produtos sejam colocados em um repositório temporário para futura compra (carrinho de compra). Deve ser possível adicionar, alterar e excluir itens de compra no carrinho. Também deve ser possível visualizar os itens no carrinho.
* Deve ser possível editar a quantidade de cada item ao adicionar um produto no carrinho. Também deve ser possível editar a quantidade de itens de um carrinho na visualização dos itens já adicionados.
* Deve ser possível a partir de um carrinho de compra realizar uma compra.
* O sistema deve calcular o frete da compra com base nos itens selecionados e o endereço apontado pelo cliente.
* O cliente pode selecionar qualquer endereço de entrega previamente cadastrado em seu perfil ou um novo endereço de entrega pode ser cadastrado. Caso um novo endereço de entrega seja inserido, deve-se dar a possibilidade que o mesmo seja incorporado ao perfil do cliente.
* O cliente pode selecionar qualquer cartão de crédito previamente cadastrado em seu perfil ou um novo cartão de crédito pode ser cadastrado. Caso um novo cartão de crédito seja cadastrado, deve-se dar a possibilidade que o mesmo seja incorporado ao perfil do cliente.
* O cliente também poderá utilizar um cupom de troca ou um cupom promocional válido.
* Deve-se possibilitar que o pagamento seja feito utilizando tanto cupons de troca, promocionais e cartão de crédito.
* Uma compra deve ser finalizada após a seleção da forma de pagamento e endereço de entrega. Após a finalização o status da compra deve ser EM PROCESSAMENTO.
* O sistema deve possibilitar que um usuário com perfil de administrador selecione vendas já aprovadas para serem entregues. Assim o status deve ficar EM TRANSITO.
* O sistema deve possibilitar que um usuário com perfil de administrador confirme entrega de uma compra. Assim o status deve ficar ENTREGUE.
* O sistema deve possibilitar que um item de uma compra seja trocado por um cliente através da visualização de pedidos do mesmo.
* O sistema deverá possibilitar que o administrador autorize pedidos ou compra com status EM TROCA. Assim o pedido passa ficar com status TROCA AUTORIZADA.
* O sistema deverá possibilitar que o administrador visualize todos pedidos de troca ou compra com status EM TROCA.
* O sistema deverá possibilitar que o administrador confirme o recebimento de pedidos de troca ou compra com status EM TROCA.
* Nesta confirmação o administrador deverá informar se os itens trocados deverão retornar ao estoque. Em caso positivo deve-se dar entrada no estoque dos respectivos itens.
* O sistema deverá gerar um cupom de troca quando o administrador informar que os itens a serem trocados chegaram. Este cupom deverá ser disponibilizado para o cliente para ser utilizado em futuras compras.
* Grupo: Controle de estoque
* O sistema deve permitir que seja possível realizar entrada de itens de livros em estoque.
* No registro de cada item, deve ser indicado o livro já previamente cadastrado e a quantidade de itens do livro.
* O sistema deve calcular o valor de venda com base no valor de custo e o grupo de precificação. Sendo que o valor de venda será o valor de compra mais o percentual definido no grupo de precificação relacionado ao livro.
* Para cada venda realizada deve-se dar baixa no estoque do total de itens vendidos.
* O sistema deve realizar a reentrada de um item em estoque a partir da troca de um produto.
  + 1. Requisitos Não Funcionais
* Utilizar linguagem C#
* Utilizar o banco de dados MSSQL Server
* A arquitetura da solução deve obedecer ao design pattern MVC.
* O sistema deve rodar nos seguintes browsers:
  + Google Chrome
* Para toda operação de escrita (Inserção ou Alteração) deve ser registado data, hora, usuário responsável além de manter os dados alterados.
* Toda bebida cadastrada deve receber um código único no sistema.
* Deve haver um script de implantação do sistema que insere todos registros de tabelas de domínio necessárias por exemplo: tipos de bebida, bandeiras de cartão, etc.
* A senha cadastrada pelo usuário deve ser composta de pelo menos 8 caracteres, ter letras maiúsculas e minúsculas além de conter caracteres especiais.
* O usuário obrigatoriamente deve digitar duas vezes a mesma senha no momento do registro da mesma.
* A senha deve ser criptografada
* O sistema deve possibilitar que endereços de entrega ou cobrança possam ser alterados ou adicionados de forma simples sem a necessidade da edição dos demais dados cadastrais.
* Todo cliente cadastrado deve receber um código único no sistema.
* Deve ser apresentado na listagem de itens do carrinho os produtos removidos por atingirem o prazo determinado para finalização da compra (apresentar o tempo conforme parâmetro do sistema). Assim a opção comprar deve ser desabilitada e o itens deverão ser adicionados novamente no carrinho.
  + 1. Regras de Negócio

Premissas

O projeto será orientado pelo professor Rodrigo Rocha.

Representação Arquitetural

O sistema será desenvolvido tendo como base a arquitetura ilustrada na Figura 1. Toda a arquitetura será baseada nos padrões de projetos tradicionais do GoF.



**Figura 1 - Modelo Arquitetural Genérico**

As Classes de Domínio são as classes que representam a abstração do negócio no estilo *anêmico* (somente com os atributos).

As Classes de Negócio representam as classes responsáveis por aplicar as regras de negócio do sistema como, por exemplo, Cadastrar Bebida. Constarão também nas classes de negócio o relacionamento com os DAOs responsáveis por persistir e recuperar os objetos no banco de dados.

Como mostra a figura, nem todas as camadas conversão entre si, gerando menor acoplamento, e aumentando a manutenção e escalabilidade do sistema, obedecendo a arquitetura MVC.

* 1. Restrições Arquiteturais
* Utilização do framework .NET 4.5, ou versões sueriores;
* Utilização do servidor IIS;
* Utilização do framework ASP.NET MVC e da tecnologia Ajax para desenvolvimento da camada de apresentação;
* Utilização do framework de mapeamento objeto-relacional ADO.NET;
* Utilização do MSSQL Server.
  1. Objetivos e Restrições Arquiteturais

Alguns requisitos registrados que impactam diretamente a arquitetura do sistema *DrinkIt*, são:

* Utilização da Linguagem C#
* Considerar a utilização de software Livre, quando possível
* O Sistema de Gerenciamento de Banco de Dados a ser considerado em implementações de âmbito corporativo será o SGBD MSSQL Server.

1. Visão de Use Case
   1. Diagrama de Caso de Manter Dados de Atributos

Uma imagem contendo texto, mapa

Descrição gerada automaticamente

Figura 2 - Diagrama de Caso de Uso do DrinkIt

1. Visão de Lógica

Esta visão apresenta elementos de design significativos do ponto de vista da arquitetura, descrevendo a organização do Sistema DrinkIt em pacotes, bem como a organização desses pacotes em camadas.



Figura 3 – Diagrama de camadas do DrinkIt

**Apresentação**: Contém classes para as interfaces gráficas com os usuários (GUI). Através destas interfaces os usuários conseguem interagir com o DrinkIt, com o intuito de incluir, alterar e excluir produtos.

**Negócio**: Contém classes que controlam a execução das funcionalidades e detém as informações e regrad do negócio relacionado ao DrinkIt.

**Persistência**: Contém classes responsáveis por persistir as entidades de modelo. Por exemplo,

contém as classes que permitem ler e gravar os objetos no banco de dados relacional.

Neste momento, é importante ressaltar que a camada de apresentação envolve componentes que são executados na porção servidora e na porção cliente. Na porção servidora, são executados os componentes que montam as páginas html de resposta e controlam o fluxo de interação com o usuário. Assim, envolve interpretação de páginas com a extensão “*.cshtml*”, utilização de helpers para montagem das páginas de resposta, delegação de solicitação para a camada de negócio, obtenção de respostas da camada de negócio, dentre outros. Já na porção cliente, estão os códigos javascript que são executados nos navegadores dos usuários com o intuito de facilitar a interação. Por exemplo, podem fazer uma pré-verificação dos dados digitados (como o preenchimento de campos obrigatórios), solicitação de páginas via ajax, de modo a tornar mais dinâmica a interação com o usuário, dentre outros.

* 1. Camada de Apresentação

Nesta camada, temos o pacote form que contém todos os arquivos relacionados à exibição de informações para usuário, o que engloba páginas HTML, imagens, javascript, dentre outros. Já o pacote de controle desta camada, contém as Actions responsáveis pela comunicação com as classes da camada de negócio. Estes pacotes podem ser vistos na Figura 5.



Figura 5: Camada de Apresentação

* 1. Camada de Negócio

Nesta camada, temos o pacote controle que contém as classes responsáveis por controlar as regras de negócio da aplicação. O pacote model, contém as classes que representam o modelo, ou seja, aquelas que contém as informações sobre o Sistema DrinkIt. O pacote controller contém as classes de controle do negócio. O pacote complementador contém classes que auxiliam na complementação de informações e o pacote filtro contém as classes que auxiliam na filtragem de informações.

* + 1. Pacote Controller



Figura 5.5: Classes de controle

* + 1. Pacote Model

Uma imagem contendo texto, mapa

Descrição gerada automaticamente

Figura 5.6: Classes do Modelo

1. Camada de Persistência

Nesta camada temos o pacote dao que contém as classes e interfaces responsáveis por persistir as informações do XXXX no BD relacional. O pacote hibernate contido em dao, possui as classes que dependem diretamente do Hibernate, que é o framework utilizado para realizar o mapeamento objeto relacional.

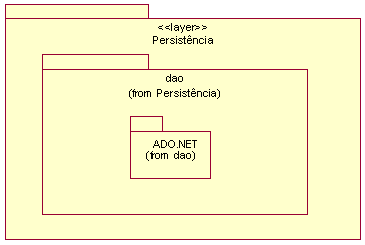


Figura 5.7: Camada de Persistência.

1. Visão de Implementação

A estrutura geral de implementação para o DrinkIt é baseada na estrutura da Visão Lógica, assim, não há necessidade de detalhar os diagramas de camadas e pacotes de implementação, uma vez que são fortemente baseados naqueles desenvolvidos para Visão Lógica.

1. Visão de Dados

O mecanismo de persistência utilizado no sistema DrinkIt utiliza-se o banco de dados Relacional MSSQL Server, juntamente com o framework de persistência de dados, ADO.NET. O controle de transações adotado envolve a utilização do Framework ASP.NET MVC em conjunto com o ADO.NET.

A figura abaixo apresenta a visão lógica e física da base de dados do DrinkIt.

Uma imagem contendo texto

Descrição gerada automaticamente

Figura XXX – Modelo Lógico

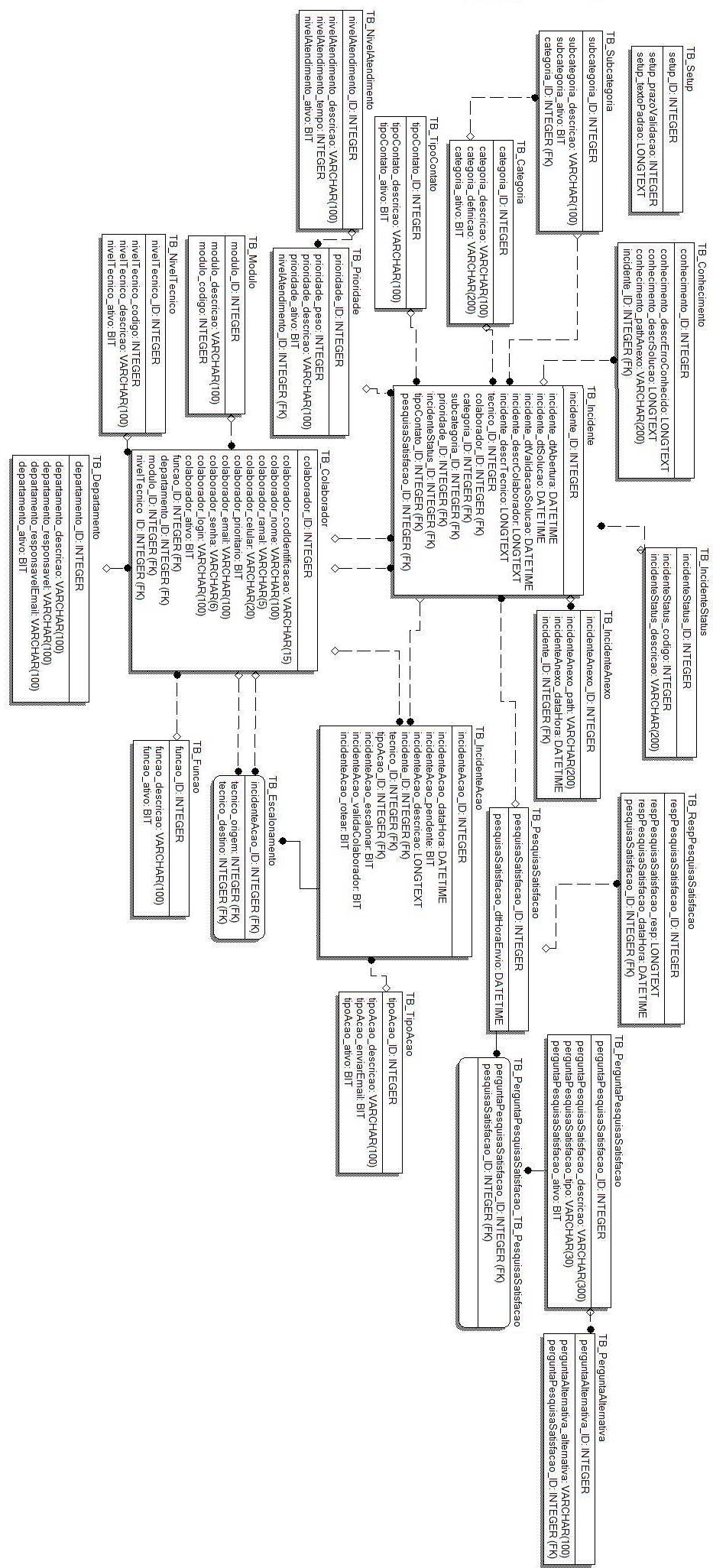


Figura XXX – Modelo Físico

A Tabela 2 define o mapeamento das principais classes de modelo para entidades do modelo lógico do BD.

1. Tamanho e Performance

O sistema DrinkIt será usado para vendas online e consequentemente terá uma grande base.

Seus servidores provavelmente irão passar por períodos de picos de utilização (por exemplo, próximo a datas comemorativas, ou períodos de festividades).

As estimativas do número de usuários e de carga de utilização em períodos de pico de utilização, bem como maiores informações sobre questões relacionadas ao tamanho e desempenho do sistema DrinkIt, podem ser obtidas no documento de requisitos não funcionais.

1. Qualidade

O sistema DrinkIt será usado para vendas online e, consequentemente tratando de altos volumes financeiros e grande número de operações de vendas diariamente.

Eventuais erros e/ou falhas na sua operação podem levar a prejuízos significativos tanto em termos financeiros, portanto na fase de design deve-se levar em consideração como fatores prioritários a confiabilidade e robustez do sistema.

Adicionalmente, o sistema DrinkIt pode ser alvo de ataques de “hackers” para roubar ou simplesmente corromper informações, possibilidade aumentada pela interface do sistema disponível na Internet, para evitar que tais ataques tenham sucesso, uma infra-estrutura de segurança deve ser especificada e projetada.

Maiores informações sobre questões relacionadas aos requisitos de qualidade do sistema DrinkIt podem ser obtidas no documento de requisitos não funcionais.

1. Cronograma Macro.

* **05/08/2019**– E-mail com grupo, tema do e-commerce, tecnologias a serem utilizadas e primeira versão do documento de requisitos;
* **07/08/2019**– Protótipos de telas Funcional (Execução no Browser);
* **19/08/2019**– Testes funcionais dos protótipos de telas do CRUD a ser entregue dia 26/08/2019;
* **26/08/2019**– CRUD na arquitetura definida – EM JAVA OU C# - CONTEMPLANDO REQUISITOSEQUIVALENTES AOS REQUISITOS DE CLIENTE NO DOCUMENTO DE REFERÊCIA, DEVERÁ JUNTOCOM A APRESENTAÇÃO TER UMA MATRIZ QUE DEMONSTRE QUAIS REQUISITOS (RF, RNF e rns) CUMPRIDOS;
* **09/09/2019**– Testes funcionais dos protótipos de telas da CONDUÇÃO a ser entregue dia16/09/2019;É esperado que todos os cenários da condução sejam implementados, INSERÇÃO DEPEDIDO, COM PAGAMENTO COM DIFERENTES FORMAS DE PAGAMENTO INCLUINDO DOISCARTÕES E CUPOM DE TROCA, ADIÇÃO DE NOVO ENDEREÇO E NOVO CARTÃO NO ATO DACOMPRA, DEFINIÇÃO DE FRETE, CONSULTA DE PEDIDOS, SOLICITAÇÃO DE TROCA ECANCELAMENTO, E MUDANÇAS DE STATUS PELO ADMINISTRADOR.
* **16/09/2019**– Testes funcionais da CONDUÇÃO;É esperado que todos os cenários da condução sejam implementados, INSERÇÃO DEPEDIDO, COM PAGAMENTO COM DIFERENTES FORMAS DE PAGAMENTO INCLUINDO DOISCARTÕES E CUPOM DE TROCA, ADIÇÃO DE NOVO ENDEREÇO E NOVO CARTÃO NO ATO DACOMPRA, DEFINIÇÃO DE FRETE, CONSULTA DE PEDIDOS, SOLICITAÇÃO DE TROCA ECANCELAMENTO, E MUDANÇAS DE STATUS PELO ADMINISTRADOR.
* **23/09/2019**– Proposta Técnica Comercial
* **30/09/2019**– DVP – Documento de Visão de Sistema
* **07/10/2019**– DAS – Documento de Arquitetura de Sistemas
* **14/10/2019**– Versão final da CONDUÇÃO, junto com a implementação deverá ser apresentado uma lista dos requisitos implementados e não implementados;
* **21/10/2019**– Especificação do caso de uso de condução;
  + -Testes funcionais da ANÁLISE (Gráfico de linhas com consulta por período e pelo menos dois indicadores - linhas);

Obs: Os prazos apresentados são uma estimativa inicial considerando as informações disponíveis nesta etapa do projeto. Um cronograma detalhado será elaborado na fase de planejamento e, eventualmente, estes prazos podem ser modificados.

1. Referências

Unified Modeling Language: <http://www.omg.org/technology/documents/formal/uml.htm>

RUP. Rational Unified Process.

ASP.NET MVC - <https://docs.microsoft.com/pt-br/aspnet/>

GOF -

<https://www.devmedia.com.br › design-patterns-padroes-gof>

MVC -

[https://tableless.com.br › mvc-afinal-e-o-que](https://tableless.com.br › mvc-afinal-e-o-que )