DOCUMENTO DE VISÃO DE PROJETO

Random Things

FATEC
Faculdade de Tecnologia de Mogi das Cruzes
Tecnologia em Análise e Desenvolvimento de Sistemas | Tecnologia em Agronegócio | Tecnologia em Recursos Humanos

Histórico de Versões

Data	Versão	Descrição	Autor	Revisor
18/03/19	1.0	Modelagem e desenvolvimento	Daniel	-
10/06/19	2.0	Revisão final	Daniel	-

Cliente FATEC - Interno

Documento Documento de Visão de Projeto: *Random Things*

Data 18 de março de 2019

Autor **Daniel Dias de Souza**
daniel160598@hotmail.com

Página de Assinaturas

Revisado e
Aprovado por: _____

18/03/19

Índice

1. Objetivo

Este documento trata principalmente da documentação 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.

1.1. Escopo

O escopo deste documento trata do desenvolvimento de um sistema que atenda as principais necessidades de um e-commerce abrangendo qualquer domínio.

O escopo deste documento é documentar as partes significativas do ponto de vista da arquitetura do modelo de design, como sua divisão em subsistemas e pacotes. Além disso, mostra sua divisão em classes e utilitários de classe.

1.2. Referências

Para a construção deste documento foram utilizadas as seguintes referências:

- Proposta Técnica Comercial Random Things
- Documento de Requisitos

2. Necessidades de Negócio

Um sistema informatizado para gestão de um comércio eletrônico é necessário para possibilitar um aumento de desmanta de todo o país. Com o sistema também é possível controlar todas as transações da plataforma, e gerenciamento do estoque.

3. Objetivo do Projeto

Desenvolver uma plataforma para soluções web capaz de:

- armazenar informações em uma base de dados
- utilizar o protocolo HTTP
- ser executado em qualquer navegador

4. Declaração Preliminar de Escopo

Esta seção descreve, em alto nível, o escopo do projeto. Os requisitos serão melhor detalhados nos documentos de Requisitos.

4.1. Descrição

Um sistema de vendas online com a possibilidade de cadastrar e manter produtos e clientes, além de possibilitar realizar transações online e gerar relatórios.

4.2. Produtos a serem entregues

Os seguintes itens são considerados produtos do projeto, na sua etapa 1.

- Documentos de especificação do sistema, concebido na fase de elaboração.
- Sistema do módulo de gestão de produtos, cliente e pedido.

4.3. Requisitos

Os seguintes requisitos descrevem em alto nível o sistema. Maiores detalhamentos podem ser acessados no Documento de Requisitos

4.3.1. Requisitos Funcionais

O sistema deve ser capaz de efetuar o cadastro, exclusão, alteração e consulta de produtos.
O sistema deve ser capaz de efetuar o cadastro, exclusão, alteração e consulta de clientes.
O sistema deve ser capaz de efetuar o controle de pedidos.
O sistema deve ser capaz de produzir uma listagem com os produtos que são os mais visualizados.
O sistema deve ser capaz de produzir relatórios referentes as operações do sistema.
O sistema deve ser capaz de adicionar novos produtos para o estoque.
O sistema deve ser capaz de realizar a consulta a históricos de pedidos.

4.3.2. Requisitos Não Funcionais

Utilizar linguagem Java
Utilizar o banco de dados MySQL.
O sistema deve ser responsivo.
O sistema deve rodar nos seguintes browsers:

- IE
- Google Chrome
- Firefox
- Cronograma de Marcos Sumariado

4.3.3. Regras de Negócio

Um produto deve estar associado com ao menos uma categoria.
O cliente deve receber um ranking numérico com base no seu perfil de compra.
Somente itens de pedidos com status ENTREGUE poderão receber solicitação de troca.

5. Premissas

- O sistema devera gerenciar produtos, clientes e vendas.
- O sistema deverá ser acessado através da internet.
- O projeto será desenvolvido pelo Daniel Dias de Souza
- O projeto será orientado pelo professor Rodrigo Rocha.

6. Influência das Partes Interessadas

- Daniel Dias: autor responsável pela especificação do sistema, definição do design, definição do conteúdo e desenvolvimento do site.
- Rodrigo Rocha: Cliente interessado no projeto.

7. Representação Arquitetural

Os sistemas serão desenvolvidos tendo como base a arquitetura ilustrada na Figura 1. Toda a arquitetura será baseada nos padrões de projetos tradicionais do GoF e também nos padrões J2EE e ECMAScript sendo executados dentro de um Servidor de Aplicações e um Servidor de interface.

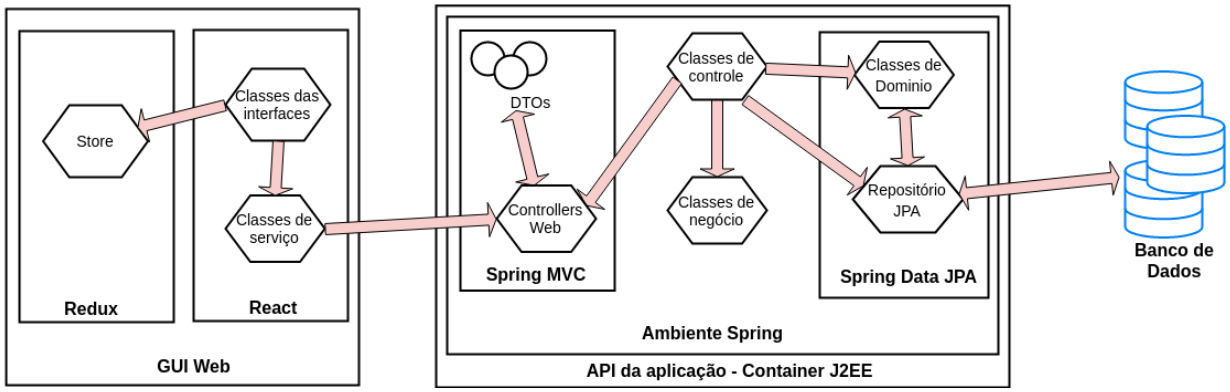


Figura 1 - Modelo Arquitetural Genérico

O React abrigará os componentes responsáveis pela interface do usuário e comunicação com o servidor de aplicações, enquanto o Redux será responsavel por manter os dados atualizados dos componentes

O Spring MVC abrigará os componentes da arquitetura responsáveis pela camada de comunicação com as requisições e transferência de dados.

O Spring Data JPA será responsável por gerenciar a comunicação com o banco de dados, e criar suas respectivas tabelas de acordo com as classes de domínio.

As Classes de Domínio são as classes que representam os Value Object, contendo somente os atributos e os métodos getters/setters.

As Classes de Negócio representam as classes responsáveis por aplicar as regras de negócio do sistema como, por exemplo, Cadastrar Produto.

As classes de controle representam as classes responsáveis por gerenciar e manipular os dados da aplicação, seguindo o padrão Command e Facade.

7.1. Restrições Arquiteturais

Foram identificadas algumas orientações / restrições pertinentes ao desenvolvimento deste subsistema:

- Utilização do JDK 1.8 ou superior do Java;
- Utilização do ambiente Spring para desenvolvimento da aplicação;
- Utilização do Ract para o desenvolvimento das interfaces gráfica;
- Utilização do SGBD MySQL.

7.2. Objetivos e Restrições Arquiteturais

Considerando premissas definidas para o Sistema Random Things pode-se citar as seguintes restrições:

- Utilização da Linguagem Java
- 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 MySQL.

8. Visão de Use Case

Esta seção apresenta os Casos de Uso arquiteturalmente significativos, que foram selecionados considerando-se o pacote do Modelo de Casos de Uso que representa o sistema Random Things.

8.1. Diagrama de Caso de Uso Arquiteturalmente Significativos

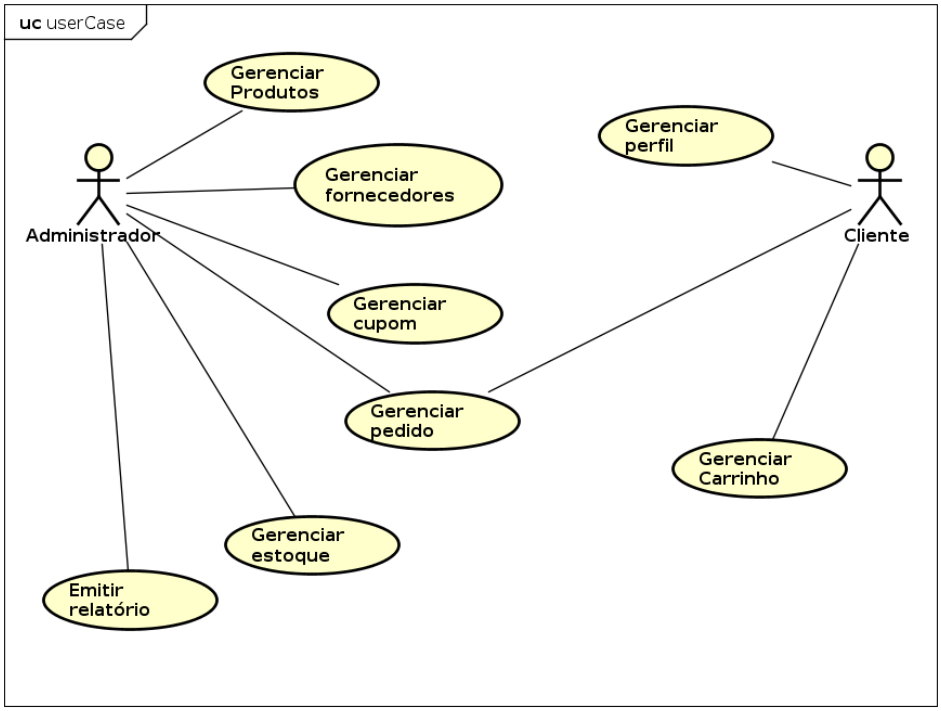


Figura 2 Diagrama de Caso de Uso Arquiteturalmente significativos.

8.2. Descrição dos Casos de Uso Arquiteturalmente Significativos

Gerenciar Produtos

Este caso de uso se inicia opcionalmente após a consulta de produtos pelo nome e o usuário solicita manutenção (incluir, alterar, excluir logicamente) os dados de Atributos.

Gerenciar Cupom

Este caso de uso inicia opcionalmente após o administrador consultar um cupom pelo nome e solicitar a manutenção (alterar ou excluir) os dados de atributos.

Gerenciar Perfil

Esse caso de uso se inicia após o cliente se logar e entrar no meu perfil. Possibilitando o usuário gerenciar seu perfil (incluir, alterar, excluir) dados.

Gerenciar Carrinho

Esse caso de uso se inicia opcionalmente após a adicionar um produto no carrinho, podendo remove-lo, alterar a quantidade e ter uma prévia do subtotal da compra.

Gerenciar Pedido

Esse caso de uso se inicia após um cliente finalizar um pedido, e o administrador separar e enviar os produtos. Além de receber os produtos em caso de trocas e criar cupons com o valor do produto ou pedido.

Gerenciar Estoque

Esse caso de uso se inicia opcionalmente após o administrador incluir uma nova entrada no estoque.

Gerenciar Fornecedores

Esse caso de uso se inicia opcionalmente após o administrador desejar incluir um novo fornecedor no sistema.

Emitir Relatórios

Esse caso de uso se inicia opcionalmente após o administrador selecionar um tipo de relatório como vendas, e a faixa de tempo.

9. Visão de Lógica

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

O Diagrama com as camadas do sistema Random Things é ilustrado na figura 3.1.

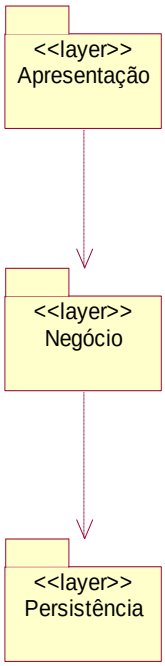


Figura 3.1 – Diagrama de camadas do Random Things

Apresentação: Contém classes para a comunicação das requisições. Através destas classes os usuários conseguem interagir com o Random Things, com o intuito de incluir, alterar e excluir produtos.

Negócio: Contém classes que controlam a execução das funcionalidades do Random Things.

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.

A figura 3.2 ilustra o diagrama de camadas com as tecnologias utilizadas no desenvolvimento, já descritas na figura 3.1.

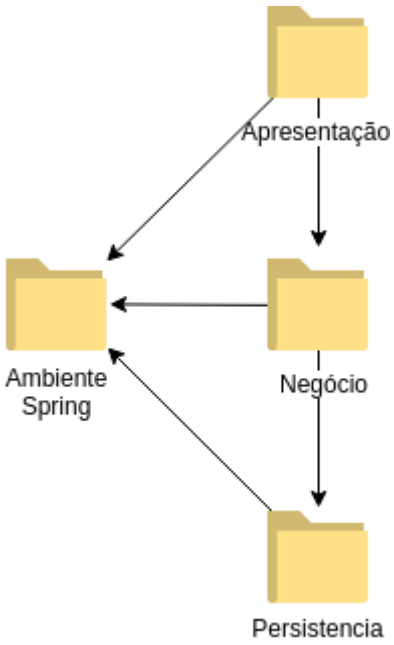


Figura 3.2: Camadas do Random Things com as dependências de tecnologia

9.1. Camada de Apresentação

9.1.1. Pacote Controller

A figura 3.3 ilustra as principais classes de controle.

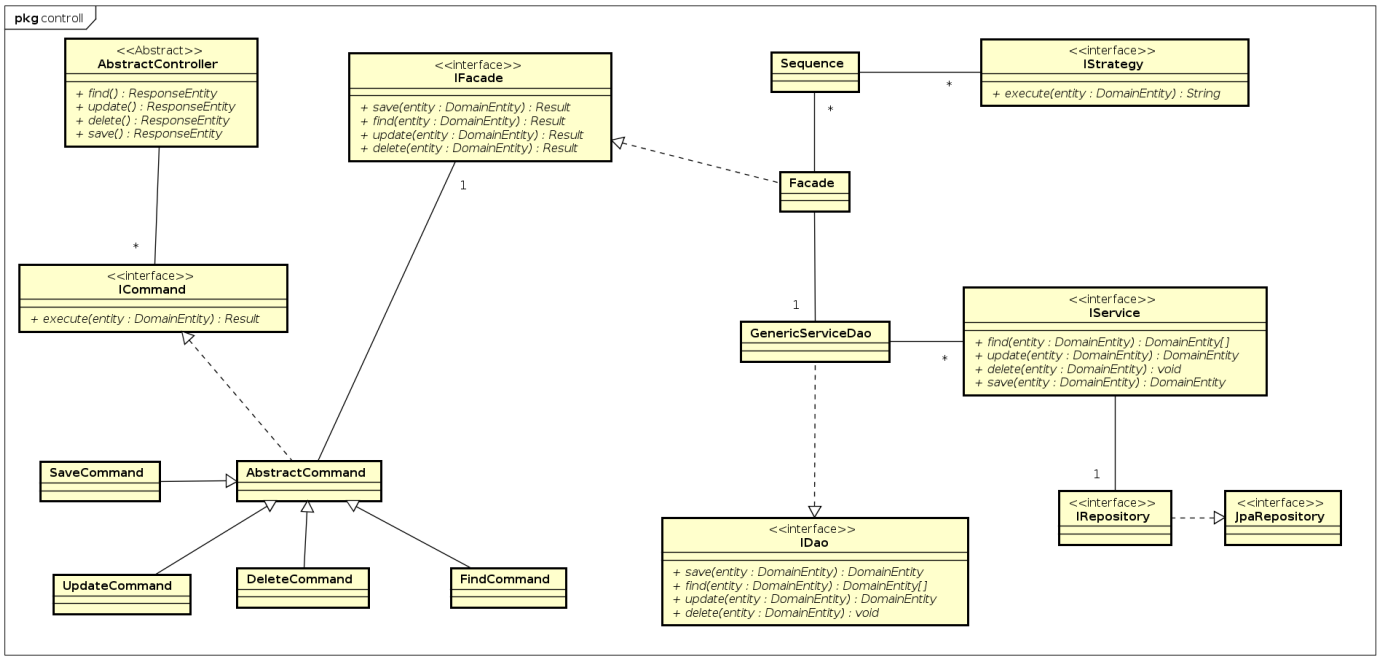


Figura 3.3: Classes de controle

9.1.1. Pacote Model

A figura 3.5. ilustra as classes do modelo.

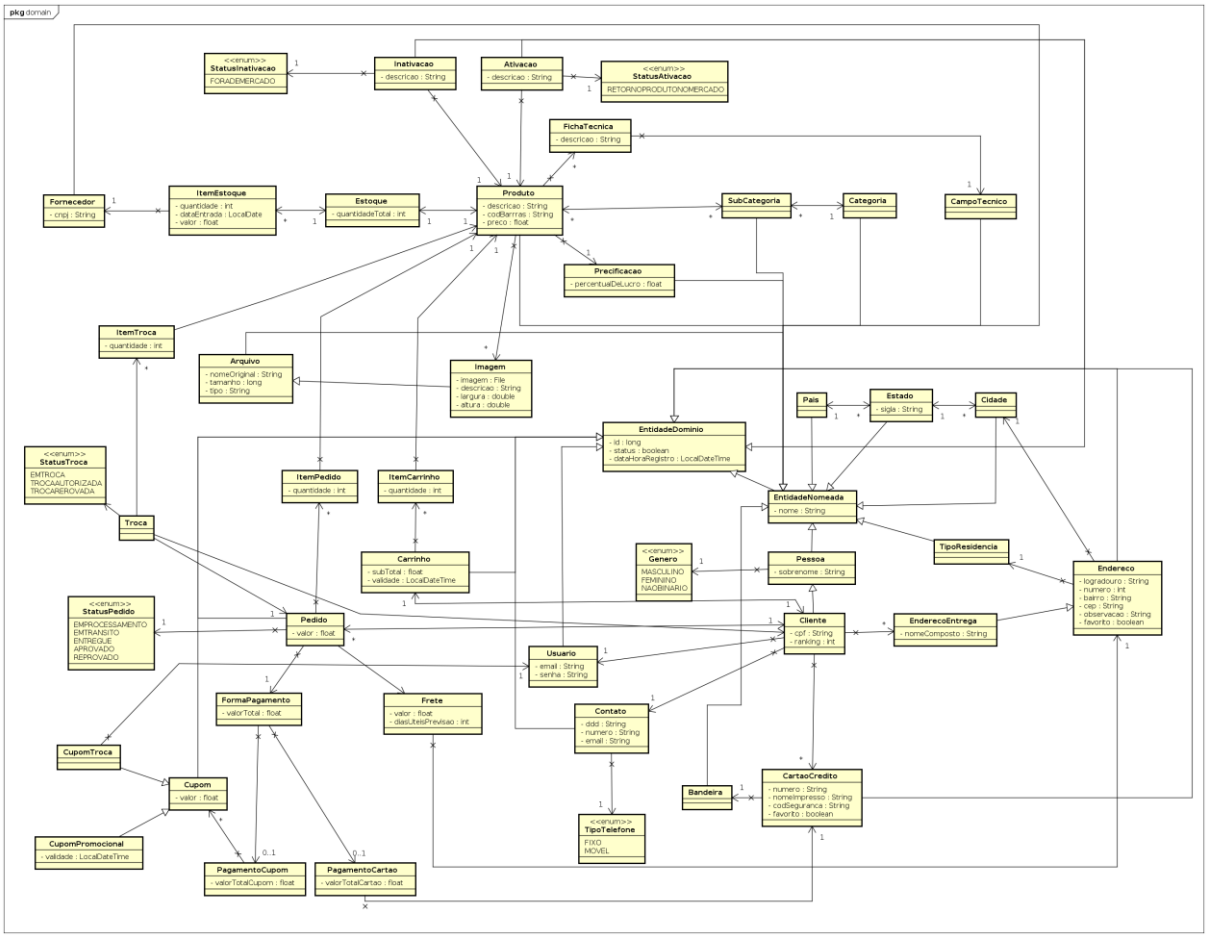


Figura 3.4: Classes do Modelo

9.1.2. Camada de Persistência

Nesta camada temos o pacote dao que contém as classes e interfaces responsáveis por persistir as informações do Random Things 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.

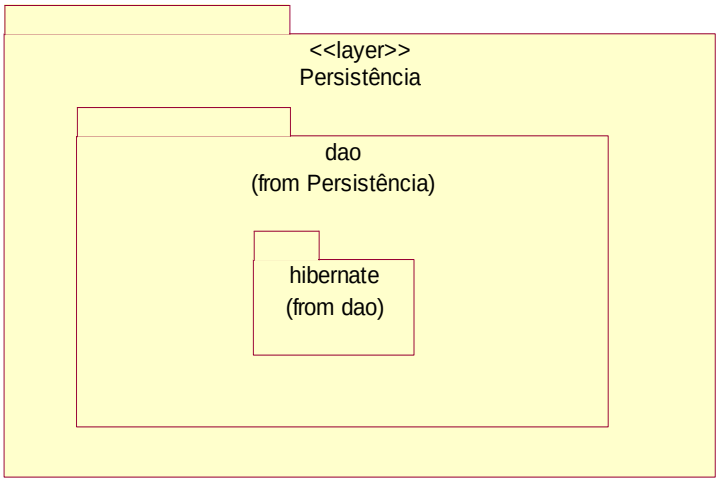
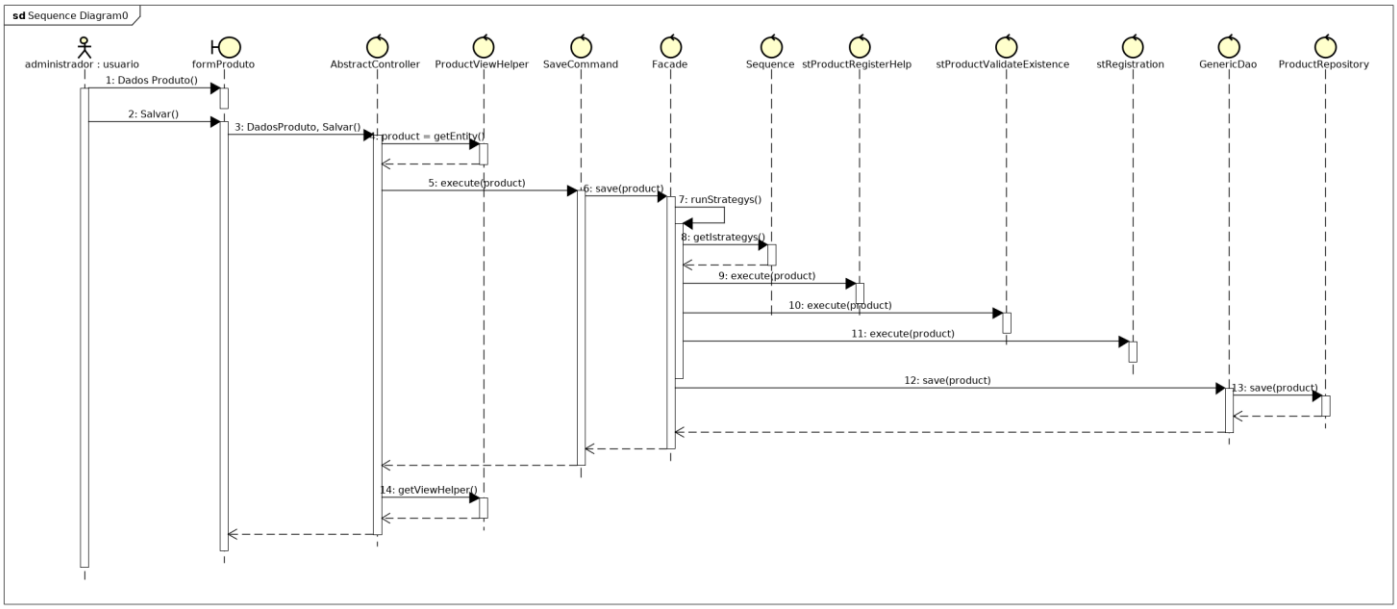


Figura 3.5: Camada de Persistência.

9.1.3. Realização dos Casos de Uso Significativos



10. Visão de Implantação

Esta seção descreve as configurações da rede física (hardware) na qual o Random Things será implantado e executado.

Trata-se de uma visão do Modelo de Implantação que, para a configuração em questão, indica os nós físicos (computadores, CPUs), que executarão o sistema, e as respectivas interconexões (barramento, LAN, etc). A figura 4 ilustra o modelo de implantação para o Random Things.

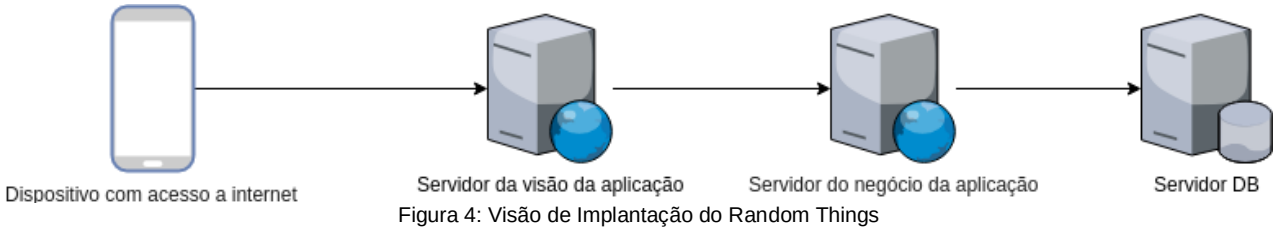


Figura 4: Visão de Implantação do Random Things

Na Figura 4 observa-se os seguintes nós físicos:

- **Web Server Application Cliente:** Aplicativos com interface de usuário via navegador, construídos com base no Framework ReactJS.
- **Web Server Aplication Domain:** Aplicação com as regras de negócio, classes de dominio e de gerenciamento das requisições, e persistência no banco, construídos com base no ambiente Spring.
- **Servidor DB**: Nó que contém o BD Central do Sistema Random Things.

11. Visão de Implementação

Esta visão descreve a estrutura geral de implementação, a decomposição do software em camadas de implementação.

A estrutura geral de implementação para o Random Things 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.

12. Visão de Dados

O mecanismo de persistência utilizado no sistema Random Things utiliza-se o banco de dados Relacional MySQL juntamente com o framework para mapeamento objeto-relacional, Hibernate. O controle de transações adotado envolve a utilização do Spring Framework em conjunto com o Hibernate.

As figuras 5 e 6, apresentam a visão lógica e física da base de dados relacionado a produto do Random Things.

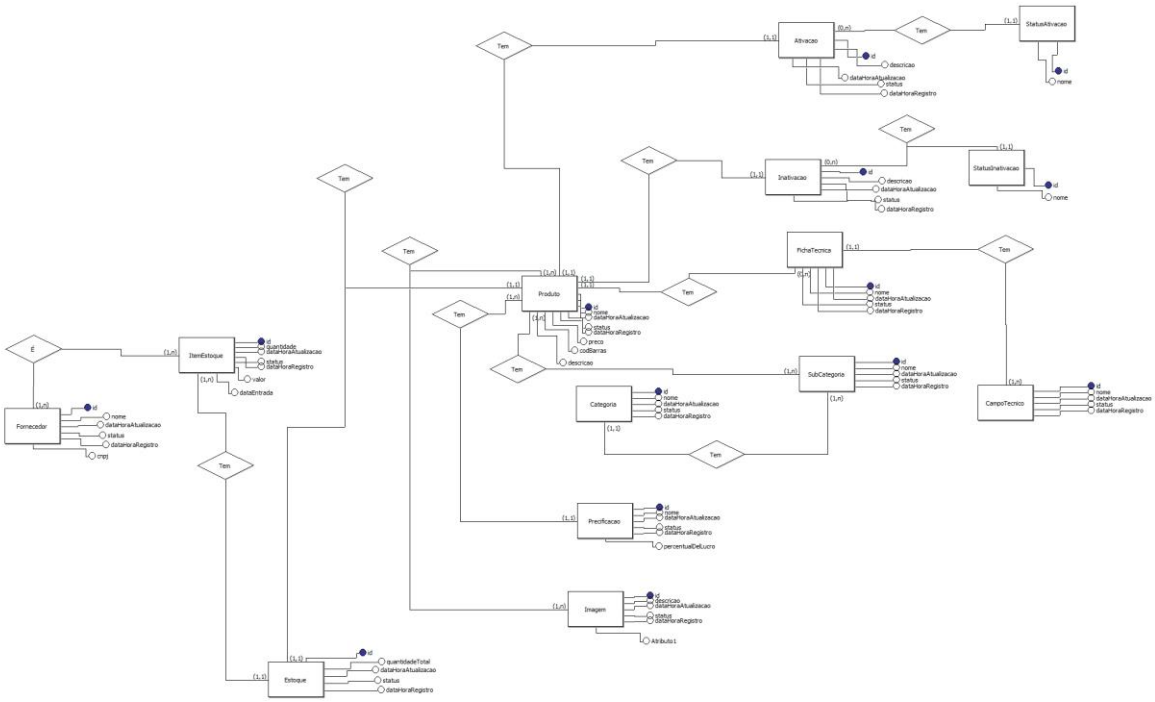


Figura 5 – Modelo Lógico Produto

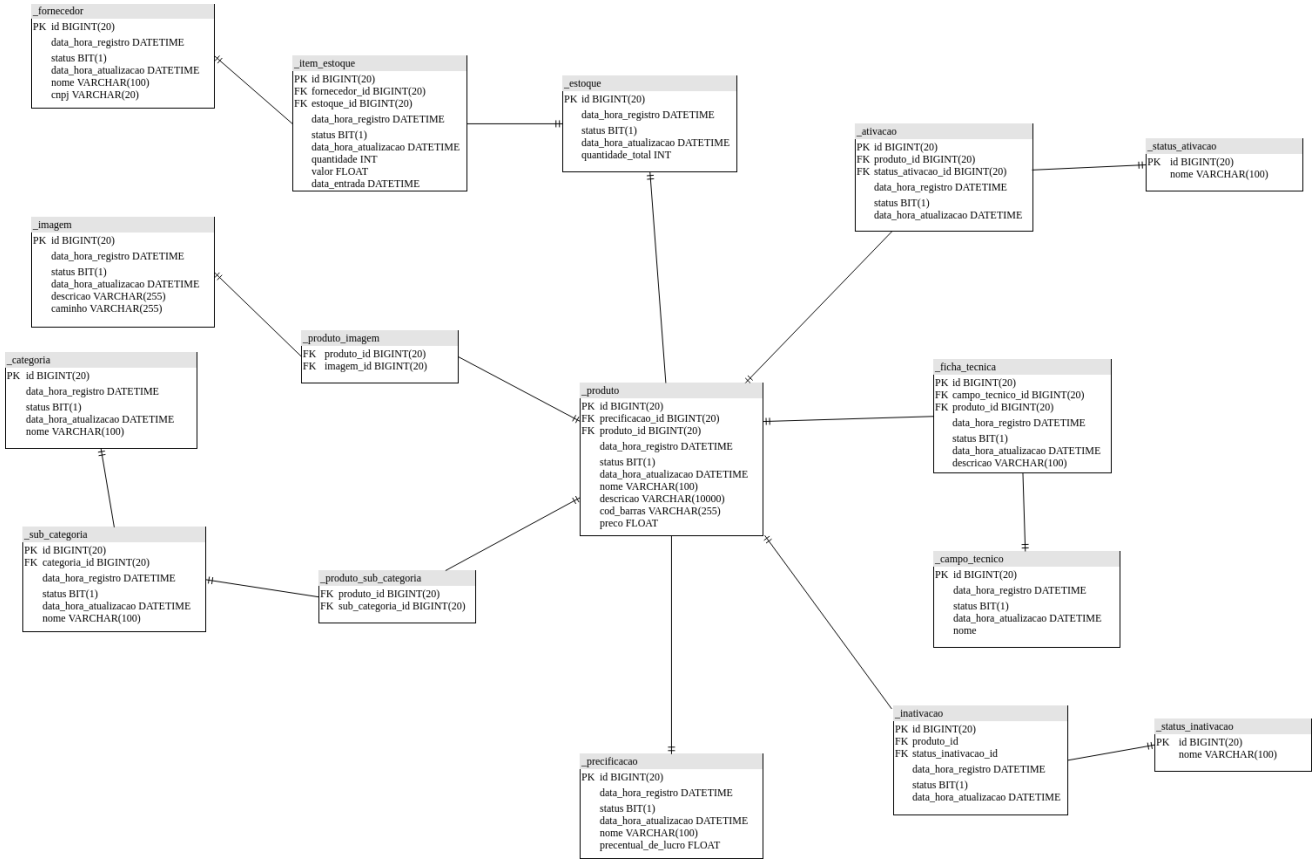


Figura 6 – Modelo Físico Produto

13. Qualidade

O sistema Random Things será usado para vendas online, conseqüentemente tratando de altos volumes de tráfego diariamente.

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

Adicionalmente, o sistema Random Things 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 sejam bem-sucedidos uma infra-estrutura de segurança deve ser especificada e projetada.

14. Cronograma Macro.

Resultado	
Apresentação do tema	18/02/2019
Documento Requisitos	25/02/2019
Proposta técnica comercial	04/03/2019
1º CRUD	11/03/2019
DVP e testes com Selenium	18/03/2019

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.

15. Referências

Unified Modeling Language: http://www.omg.org/technology/documents/formal/uml.htm
RUP. Rational Unified Process.