Drugs & Dragons

Documento de Arquitetura de Software

Versão 2.0

*Alunos(a):*

Bruno Decnop

Hugo Bianquini

Rafaela Peçanha

Rafael Martins

Tiago Matos

Thomaz Falcão

Histórico da Revisão

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 27/09/2022 | 1.0 | Elaboração inicial, Introdução, Visão Lógica da arquitetura | Julia Ribas |
| 28/09/2022 | 1.0 | Elaboração da Camadas de Arquitetura e Mecanismos Arquiteturais | Rafaela Peçanha |
|  |  |  |  |
|  |  |  |  |

Índice Analítico

1. Introdução 4

1.1 Finalidade 4

1.2 Definições, Acrônimos e Abreviações 4

2. Metas e Restrições da Arquitetura 4

3. Suposições e Dependências 4

4. Requisitos Arquiteturalmente Significantes 4

5. Decisões, Restrições e justificativas 4

6. Mecanismos Arquiteturais 5

7. Camadas da Arquitetura 5

8. Visões da Arquitetura 5

9. Qualidade 6

Documento de Arquitetura de Software

# Introdução

O presente documento tem por finalidade apresentar as características da arquitetura do projeto de farmácia eletrônica Drugs & Dragons. O sistema visa proporcionar ambiente adequado para a busca e compra de remédios da parte de pacientes, de maneira a facilitar o acesso à medicação

## Finalidade

Este documento oferece uma visão geral arquitetural abrangente do sistema, usando diversas visões arquiteturais para representar diferentes aspectos do sistema. O objetivo deste documento é capturar e comunicar as decisões arquiteturais significativas que foram tomadas em relação ao sistema

## Definições, Acrônimos e Abreviações

MVC: Padrão de arquitetura de software.

M significa “Modelo”, porção responsável pelas regras de negócio,   
V significa “Visualização”, responsável pela parte de interfaces,   
C significa “Controle”, sendo a parte de controle de dados.

# Metas e Restrições da Arquitetura

*[Descreva a filosofia da arquitetura. Identifique as questões que dirigem a arquitetura, como “O sistema será dirigido por funcionalidades complexas, integração com sistemas legados ou questões de desempenho? O sistema precisa ser robusto durante um longo tempo de manutenção ?*

*[Formule um conjunto de metas que a arquitetura tem de cumprir em sua estrutura e comportamento. Identifique questões críticas que a arquitetura deve resolver, tais como “há dependências de hardware que precisam ser isoladas o resto do sistema?” e “O sistema precisa funcionar de forma eficiente em condições não-usuais?”]*

# Suposições e Dependências

*[Liste as suposições e dependências que dirigem as decisões arquiteturais. Isto pode incluir áreas sensíveis ou críticas, dependências e interfaces com sistemas legado, a habilidade e experiência da equipe, a disponibilidade de recursos importantes, e assim por diante ]*

1. **Requisitos Funcionais**

* **NC.001 - Usuários**

* **RF 1.01** - O sistema deve permitir o cliente realizar o cadastro, editar, ou excluir sua conta da plataforma.
* **RF 1.02** - O sistema deve possuir uma seção de gerenciamento de usuários, manipulada por um administrador.
* **RF 1.03** - O sistema deve possuir uma seção de login para usuários já cadastrados.
* **RF 1.04** - O cliente poderá acessar seu histórico de compras, e detalhamento de pedidos feitos anteriormente.
* **RF 1.05** - O sistema deve permitir que o funcionário bloqueie contas de clientes.
* **RF 1.06** - O sistema deve permitir que o funcionário valide a prescrição médica referente a determinado produto.
* **NC.002 - Produtos**

* **RF 2.01** - O sistema deve permitir que o funcionário gerencie produtos. Cada produto deve estar alocado a uma posição no depósito.
* **RF 2.02** - Um produto pode exigir ou não prescrição.
* **RF 2.03** - O sistema deve exibir uma página inicial com os produtos mais vendidos, sugestões para o usuário.
* **RF 2.04** - O sistema deve ter uma barra de pesquisa e um filtro avançado para busca de produtos.
* **RF 2.05** - O funcionário deve poder consultar a posição do produto no depósito.
* **NC.003 - Estoque**

* **RF 3.01** - Ao cadastrar um produto, deve-se inserir a quantidade de estoque deste produto.
* **RF 3.02** - O funcionário pode gerenciar o estoque dos produtos.
* **RF 3.03** - Após a confirmação de separação do produto de um pedido, o estoque precisa ser atualizado.
* **NC.004 – Carrinho**
  + **RF 4.01** - Um carrinho é composto por itens de produto. Um item é um produto associado a uma quantidade determinada pelo cliente.
  + **RF 4.02** - O cliente pode gerenciar o seu carrinho.
  + **RF 4.03** - Um cliente pode ter somente um carrinho ativo.
  + **RF 4.04** - O cliente pode adicionar produtos ao carrinho, caso estejam em estoque.
  + **RF 4.05** - O cliente deve selecionar uma forma de pagamento. As formas de pagamento são: cartão de crédito, débito, pix, transferência bancária ou boleto.
  + **RF 4.06** - O cliente deve selecionar um endereço de entrega. Após selecionar, deve ser calculado o frete.
  + **RF 4.07** - O carrinho deve exibir o subtotal do pedido.
* **NC.005 - Pedido**
* **RF 5.01** - Após a confirmação da compra pelo cliente, será gerado um novo pedido, sob status “Aguardando confirmação de pagamento”.
* **RF 5.02** - Após retorno positivo do pagamento, o pedido receberá o status de “Pagamento Confirmado”, o usuário receberá uma notificação de confirmação, e os produtos serão separados para entrega.
* **RF 5.03** - O funcionário pode atualizar o status do pedido de “Pagamento confirmado” para “Entrega em andamento”. Ao fazer isso, o estoque deve ser atualizado.
* **RF 5.04** - O funcionário pode cancelar pedidos.
* **RF 5.05** - O cliente pode cancelar seu pedido, até que este saia para a entrega.
* **RF 5.06** - Qualquer pedido cancelado o pagamento deve ser reembolsado. Qualquer cancelamento deve ser notificado por email ao usuário.
* **RF 5.07** - O sistema deve emitir uma nota fiscal para o cliente após a confirmação de pagamento de um pedido.
* **RF 5.08** - O cliente pode monitorar o status do seu pedido e receberá o código de rastreamento assim que sair para entrega.

1. **Requisitos Não Funcionais**

* **RNF 1.01** - O sistema deve utilizar o banco de dados postgresql v11 para armazenar seus dados.
* **RNF 1.02** - O sistema deve ser suportado por todos os browsers em suas versões mais recentes
* **RNF 1.03** - A plataforma poderá ser acessada em qualquer horário do dia, 24 horas de disponibilidade ininterrupta. Há possibilidades de paradas pré-programadas para manutenção, preferencialmente entre 22h e 6h.
* **RNF 1.04** - Um usuário comum deve ser capaz de comprar um produto em menos de 10 minutos.
* **RNF 1.05** - O sistema não deve demorar mais do que 30 segundos para que uma página seja totalmente carregada.
* **RNF 1.06** - O sistema deve realizar backup diariamente de forma automática.

# Requisitos Arquiteturalmente Significantes

*[Insira uma referência ou link para os requisitos que exploram aspectos relevantes da arquitetura.]*

# Decisões, Restrições e justificativas

*[Liste as decisões tomadas em relação às abordagens arquiteturais. Estas decisões servirão como guias para definir as partes arquiteturalmente significantes do sistema. Justifique cada decisão ou restrição para que os desenvolvedores compreendam a importância de construir o sistema dentro desse contexto. Isto pode incluir uma lista de “Faça” ou “Não Faça” para guiar os desenvolvedores no desenvolvimento do sistema]*

* Decision or constraint and justification
* Decision or constraint and justification

# Mecanismos Arquiteturais

**Mecanismo de Análise**

O Mecanismo de Análise tem a finalidade de mostrar padrões de estruturas, comportamento ou ambos. Serve para diminuir a complexidade da análise e aperfeiçoar sua consistência, o que acarreta aos designers uma representação resumida de um comportamento complicado. Dessa froma, ainda se tem o mecanismo de análise de persistência que permite ter objetos persistentes e pegar requisitos que são necessários cumprir o mecanismo de persistência, sem pensar em como ou o que esse mecanismo irá realizar.Além disso, tem se o mecanismo de análise de comunicação entre processos que permite os elementos de modelo se comunicar com componentes ou serviços executados em outros processos.

**Mecanism de Desenho**

O Mecanismo de Desenho tem como finalidade guiar a implementação e documentar a implementação, sendo o processo de transferência do espaço do problema para o espaço da solução. O desenho tem o intuito de lidar com complexidade, independência, inteligibilidade e adaptabilidade.

**Mecanism de Implementação**

O Mecanismo de Implementação tem a finalidade de solucionar o problema colocando em prática e tornando concreta a solução, por meia da implentação. No qual o sistema é desenvolvido e é possível a compilação e geração do código-executável para o desenvolvimento do projeto.

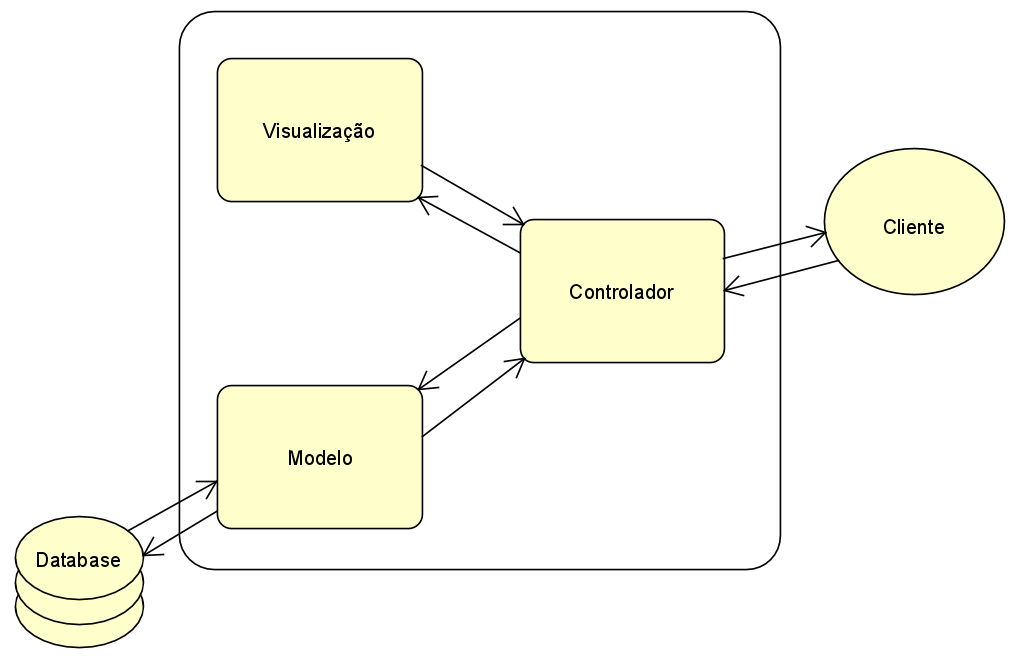
# Camadas da Arquitetura

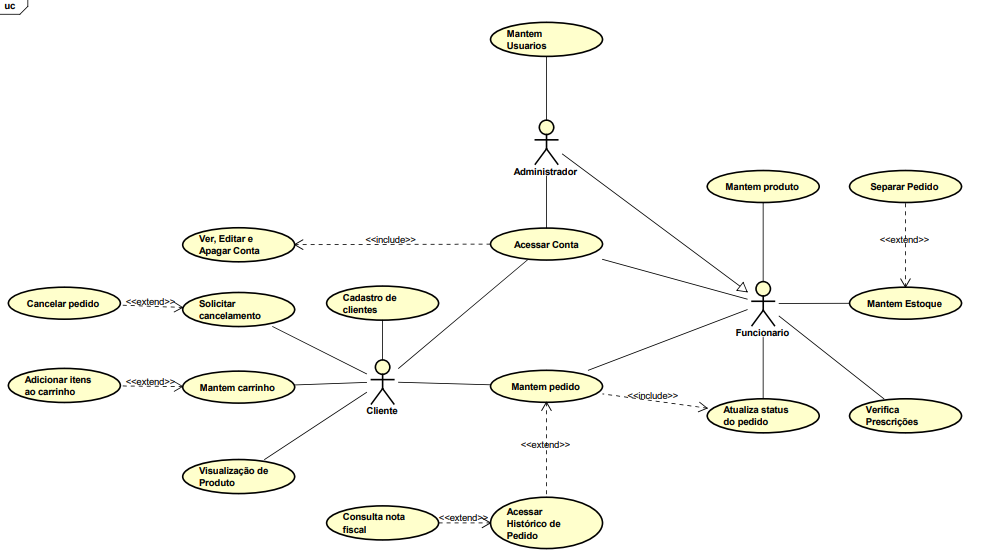
O padrão arquitetural MVC (Model-View-Controller) por possuir uma forma de implementação de interface gráfica, além de possuir o conceito de orientação a objetos se enquadra nos padrões do projeto a ser implementado. O MVC, é divido em visão, controladores e modelo, no qual a interface gráfica é formada pela visão e os controladores. Como o intuito do projeto é ser uma interface web, deverá ser utilizado o Ruby on Rails que é denomindado um framework MVC, no qual teremos um sistema em três partes: clientes (browser), aplicação web (visão, controladores e modelo) e banco de dados. Dessa froma, uma das vantagens de utilizar o MVC é que o mesmo favore a especialização do trabalho de desenvolvedor, e já que estaremos utilizando o Ruby on Rails, poderemos dividir em: back-end e front-end. Outras vantagens é que favorece a testabilidade e as classes de Modelo podendo ser utilizadas por diferentes Visões.

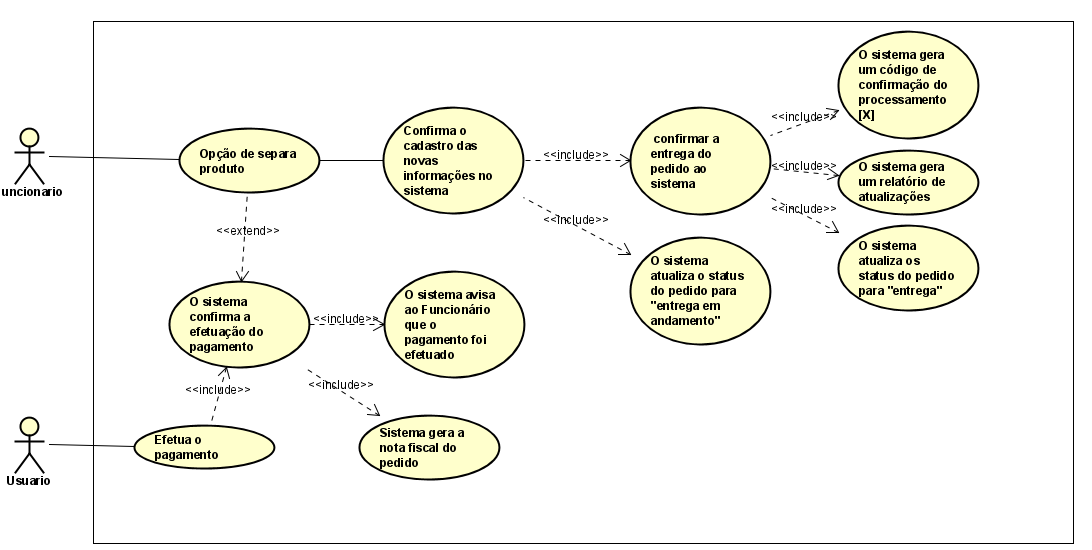
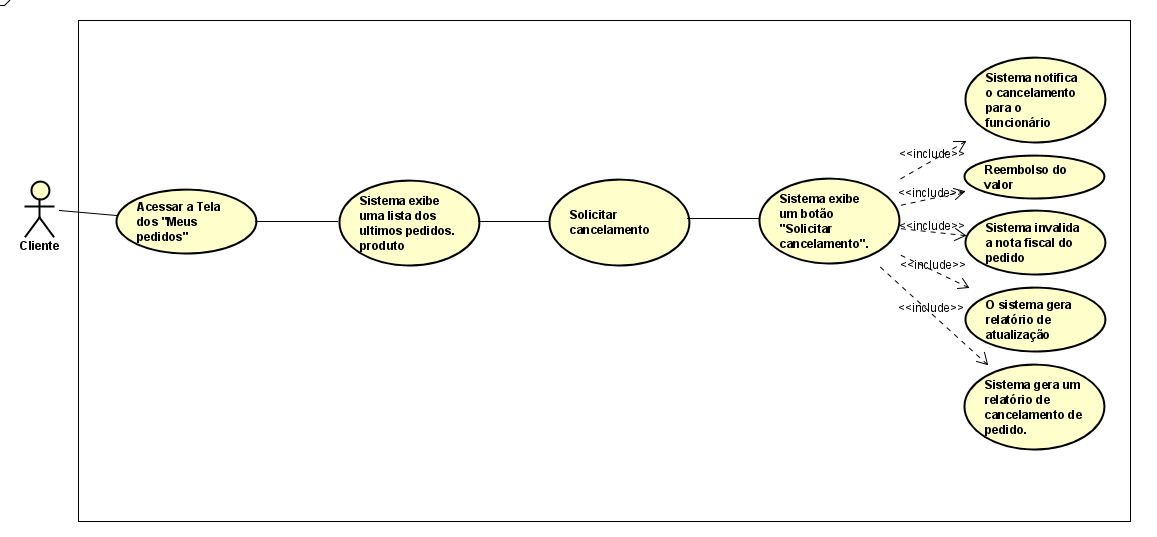
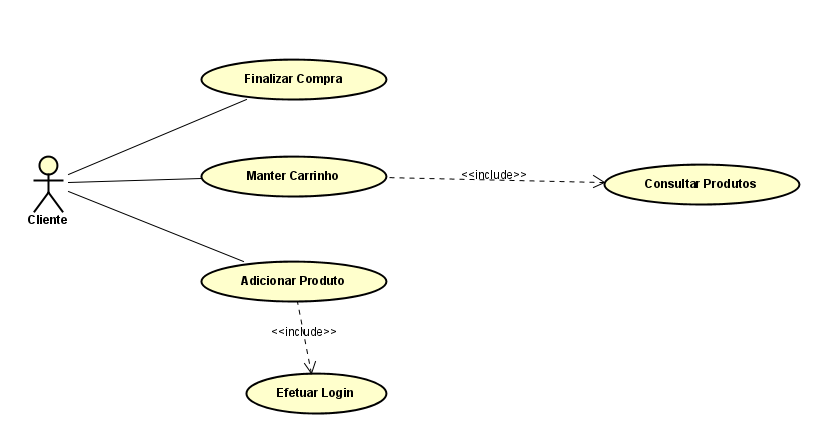
# Visões da Arquitetura

Visões Recomendadas:

* **Lógica:**



* **Operacional:** Descreva os nós físicos do sistema e os processos, threads e componentes que rodam em cada um desses nós. Esta visão não é necessária se o sistema roda num único processo e num único thread.
* **Casos de Uso:** 

****

**Implementação**

Foram criados os vies, controllers e models de cada uma das entidades de interesse do sisstema no framework Ruby on Rails que utiliza a linguagem Ruby na versão 3.0.

No exemplo abaixo vemos a implementação do controlador de carrinho que despacha as atividades do CRUD relacionadas ao carrinho.

**Texto

Descrição gerada automaticamente**

**Texto

Descrição gerada automaticamente**

Foram criadas as respectivas views para este controller

**Texto

Descrição gerada automaticamente**

E , como utilizamos o framework modelado como ORM (Object Relational Mapping) temos as entidades persistindo no banco de dados, no caso, POSTGRESQL.

Texto

Descrição gerada automaticamente

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

# Qualidade

*[Uma descrição de como a arquitetura do software contribui para todos os recursos (exceto a funcionalidade) do sistema: extensibilidade, confiabilidade, portabilidade e assim por diante. Se essas características possuírem significado especial, como implicações de segurança, garantia ou privacidade, elas deverão ser delineadas claramente.]*