Ponto de Venda

Documento de Arquitetura de Software

Versão <1.0>

Histórico da Revisão

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 18/05/22 | 1.0 | Elaboração do Documento | Jair de Lima, Thiago R. da Motta, Victor Verdan |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Índice Analítico

1. Introdução 4

1.1 Finalidade 4

1.2 Escopo 4

1.3 Definições, acrônimos e abreviações 4

1.4 Visão Geral 4

2. Metas e Restrições da Arquitetura 4

3. Suposições e Dependências 4

4. Requisitos Arquiteturalmente Significantes 5

5. Decisões, Restrições e justificativas 5

6. Mecanismos Arquiteturais 5

7. Camadas da Arquitetura 5

8. Visões da Arquitetura 5

8.1 Visão Lógica 5

8.1.1 Visão Geral 5

8.1.2 Estrutura de Pacotes Significativos 5

8.1.3 Visão Lógica 5

8.2 Casos de Uso 5

9. Qualidade 6

Documento de Arquitetura de Software

# Introdução

Este documento tem como objetivo descrever o documento de arquitetura do projeto Ponto de Venda. Esse projeto tem como propósito a implementação de um sistema de caixa para mercado que seja de fácil adoção e customização para poder ser adotado a diferentes tipos de negócio.

## 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.

## Escopo

Este documento é voltado para o auxílio dos envolvidos desenvolvimento no projeto Ponto de Venda, captando aspectos arquiteturais do sistema.

## Definições, acrônimos e abreviações

* MVC – Padrão de arquitetura de software que consiste em M – Modelo, V – Visualização e C – Controlador. O modelo é responsável pelas regras de negócio, visualização é responsável a parte gráfica da aplicação e controladores pelo controle de dados e eventos.
* DAO – Padrão de acesso a dados Data Access Object que tem como objetivo separar a lógica de negócios da lógica de persistência de dados.
* CRUD – Conjunto de operações básicas de um banco de dados relacional. C – Create, R – Read, U – Update e D – Delete, ou traduzindo, criar, ler, atualizar e deletar, respectivamente.

## Visão Geral

Serão apresentados neste documento diferentes visões arquiteturais de como o sistema deverá se comportar em diferentes processos, como deverá ser implementado, justificações pelas escolhas feitas nesse projeto junto a como elas contribuem para todos os recursos.

# Metas e Restrições da Arquitetura

Algumas das restrições de requisito e de sistema terão uma relação fundamental com a arquitetura do projeto sendo elas:

* Sistema deverá ser multiplataforma
* Utilização do paradigma de desenvolvimento Orientado a Objetos;
* Padrão de estrutura MVC ou Orientado a Mensagens;
* Linguagem de programação *Python*;
* Framework *Tkinter* ou *Kivy*;
* Padrão DAO para operações CRUD;
* Banco de dados *MySQL* ou *Postgres*.

[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 ]

# 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

[Liste os mecanismos arquiteturais, como mecanismos de persistência, comunicação e tratamento de erros, por exemplo, e descreva ocorrente estado de cada um. Inicialmente, cada mecanismo pode ser somente um nome e uma breve descrição. Eles evoluirão até que o mecanismo se torne um padrão ou uma colaboração de elementos de projeto que possam ser aplicados diretamente em algum aspecto do projeto.]

**Mecanismo Arquitetural 1**

[Descreva a finalidade, os atributos e funções do mecanismo arquitetural.]

**Mecanismo Arquitetural 2**

[Descreva a finalidade, os atributos e funções do mecanismo arquitetural.]

# Camadas da Arquitetura

[Descreva os padrões de arquitetura utilizados e como a arquitetura se manterá consistente e uniforme. Isto pode ser uma simples referência para um conhecido padrão arquitetural, como o padrão de divisão em camadas e uma descrição de como os componentes do sistema podem ser colocados juntos.]

# Visões da Arquitetura

## Visão Lógica

### Visão Geral

A visão lógica define a estrutura da arquitetura. Abaixo será especificado o padrão MVC que foi selecionado para o desenvolvimento do sistema com seus pacotes principais.

### Estrutura de Pacotes Significativos

<imagem>

Diagrama de Classe geral do sistema

### Visão Lógica

## Casos de Uso

Os seguintes diagramas descrevem os casos de uso do sistema.

<imagens>

# Qualidade

O padrão de arquitetura adotado para esse projeto tem como principal objetivo garantir uma boa organização do código fonte, manutenibilidade e possibilidade de extensão das funcionalidades ou modificação de elementos como a interface gráfica ou banco de dados sem drasticamente afetar o funcionamento do sistema.