Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

**Documentação de Software**

**Food Manager**

Autores:

Vitor Cristiano Fellizatti

Thiago dos Santos Rodrigues

Thiago Barros Gomes

Vinicius Augusto Nigra

Vitor de Carvalho

Documento de especificação de requisitos do projeto

" Food Manager." 

Versão: 1.2

Projeto Interdisciplinar do 3º semestre do curso de Desenvolvimento de Software Em Multiplataforma desenvolvido na Faculdade de Tecnologia de Araras (FATEC), apresentado como Trabalho de Conclusão.

**Faculdade de Tecnologia de Araras**

**DESENVOLVIMENTO DE SOFTWARE EM MULTIPLATAFORMA**

**TRABALHO DE CONCLUSÃO DO 3ºSEMESTRE.**

Orientadores:

Prof. **RENATO CIVIDINI MATTHIESEN**

Gestão Ágil de Projeto de Software

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Orlando Saraiva

Desenvolvimento Web III

Prof. **THIAGO GONCALVES MENDES**

Banco de Dados Não Relacional

**SUMÁRIO**

**1.0 INTRODUÇÃO5**

**1.1 Objetivo5**

**1.2 Escopo6**

**1.3Visão geral7  
1.4 DESCRIÇÃO GERAL8**

**2.0 Requisitos Funcionais** **9**

**2.1 Requisitos Não Funcionais** **9**

**2.2 Requisitos de interface** **10**

**2.3 Atributos de qualidade11**

**2.4 Características dos usuários12**

**3.0 Anexo** **13**

**4.0 Ferramentas** **14**

**4.1 Linguagens** **14**

**5.0 Diagrama Caso de Uso15  
5.1 Diagrama caso de uso - instituição17****6.0 Banco de dados19**

**1.0 INTRODUÇÃO**

Este documento tem por propósito especificar os requisitos necessários da página de web do Food Manager para o seu desenvolvimento. Além de servir de referência para a manutenção do software. Foram utilizados diagramas UML (diagrama de casos de uso e diagrama de sequência), requisitos funcionais e requisitos não funcionais para ajudar no entendimento das funcionalidades do sistema.

* 1. **OBJETIVO**

Nosso sistema permite a organizações de caridades e banco alimentos.

Registrando e rastreando doações, estoques e distribuições de alimentos.

Várias instituições recebem produtos com variedades de validades e como o gerenciamento é humano acaba ocorrendo falhas pois sendo manual via papel de controle os alimentos acaba sendo mal distribuídos. Através do nosso sistema vai ser mais fácil o controle dos produtos os que tem prioridade de saída e os tem que maior necessita podem ser pegos e distribuídos primeiros

1.2 **ESCOPO**

Gerenciador de Armazenamento de alimentos para pessoas com necessidades- F.M

Executável: Nesse projeto faremos a ponte entre usuário que precisam de alimentos com os doadores.

Benefícios: Agilizar informações necessárias, compartilhar de forma rápida o estado.

Objetivos: Agilizar a conexão do usuário que precisa com o produto e quem disponibiliza o mesmo,

Meta: Acelerar o encontro entre essas duas entidades.

**1.3 VISÃO GERAL**

* Muitas instituições lidam com uma variedade de produtos com datas de validade diversas. O gerenciamento manual desses alimentos resulta frequentemente em falhas, levando a distribuições inadequadas. Através do nosso sistema, buscamos resolver esse problema, proporcionando:​
* **Controle de Inventário Eficiente**​
* **Priorização de Alimentos**​
* **Agilidade no gerenciamento da distribuição**​
  1. **DESCRIÇÃO GERAL**

Os usuários conseguiram colocar as informações que serão necessárias para conseguimos fazer a ponte em quem precisa com aqueles que irão doar.

Terão janelas que vai ter essa informações de são necessária para a facilida.

**2.0 REQUISITOS FUNCIONAIS**

RF01: Cadastrar de informações - obrigatório

RF02: Compartilhar de informações entre usuários da plataforma - obrigatório;

RF03: Acrescentar informações - obrigatório;

**2.1 REQUISITOS NÃO FUNCIONAIS**

RNF01: tempo de resposta não deve exceder 5 segundos;

RNF02: Acesso de internet;

RNF03: Python como linguagens;

RNF04: Mongo DB como banco de dados;

RNF05: Utilização de framework.

**2.2 REQUISITOS DE INTERFACE**

RI01: Layout Intuitivo e fácil de navegar. Com elementos claros e organizados, permitindo que os usuários encontrem rapidamente as informações necessárias.

RI02: Design Responsivo: Acessível em diferentes dispositivos, como desktops, tablets e smartphones.

RI03: Segurança e Privacidade: A segurança dos dados do usuário. A interface deve implementar medidas robustas de segurança, como criptografia e autenticação.

RI04: Registro e Pesquisa Eficiente: A interface deve permitir o registro eficiente de informações.

**2.3 ATRIBUTOS DE QUALIDADE**

AQ01: Segurança e Privacidade: A segurança dos dados é fundamental. Implementação de medidas robustas de segurança, como criptografia e autenticação, para proteger as informações confidenciais dos pacientes contra acessos não autorizados.

OBRIGATÓRIO

AQ02: Design Responsivo: O Food Manager é acessível em diferentes dispositivos, como desktops, tablets e smartphones. O design responsivo garante que a interface se ajuste automaticamente ao tamanho da tela, proporcionando uma experiência consistente e adaptável em todos os dispositivos.

DESEJÁVEL

**2.4 CARACTERISTICAS DOS USUÁRIOS**

Este *software* destina-se a facilitar o encontro com a instituição que tem o produto para quem necessita do mesmo.

INSTITUIÇÃO: Será a entidade que seria empresas e comércio quem excedente de produto que queira doa esse excedente.

PESSOAS: Essa entendida será os usuários que receberam as doações.

**3.0 ANEXO**

**Métodos utilizados:**

*Brainstorming*: Utilizamos para levantar ideias e dar início a uma discussão que ainda não está formada. Assim rapidamente se levanta uma grande quantidade de ideias.

Para sabermos a necessidade de quem vai utilizar o produto ou saber detalhes sobre o funcionamento de um processo qualquer. Coletando assim as informações como são, permitindo aferir a usabilidade de um processo.

Pensando em grupo, com participantes que podem ser internos ou externos ao projeto. No *brainstorming*, os participantes falam livremente, sem coibição.

Protótipos: Utilizamos para coletar um *feedback* mais concreto das partes interessadas. Sendo atualizado diversas vezes, até solidificar os requisitos.

Um subproduto da solução principal que deve mostrar as funcionalidades que se desejem avaliar durante o levantamento de requisitos de um projeto.

**4.0 FERRAMENTAS**

FER01: *Visual Studio Code* – codificação

FER02: *MongoDB* – banco de dados

FER03: *Zeal* – documentação

FER04: MongoDB - documentação

FER05: *Draw.io*

**4.1 LINGUAGENS**

LING01: *JavaScript*

LING02: HTML

LING03: CSS

LING04: PHP

LING5: NOSQL

LING6: Python

5.0 DIAGRAMA DE CASO DE USO

**5.1 INSTITUIÇÃO**

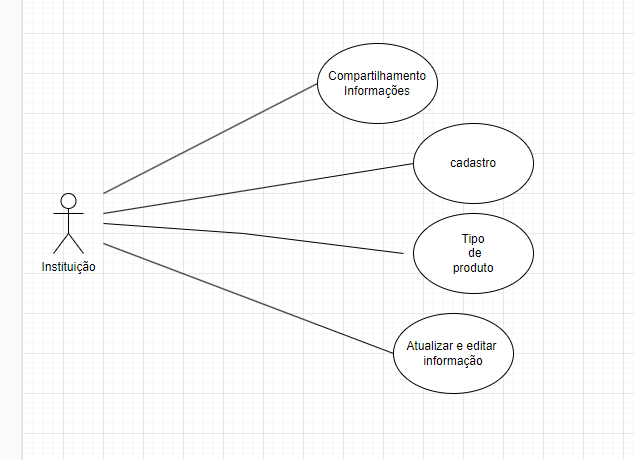
****

Figura 01 – Diagrama de Caso de Uso

**6.0 BANCO DE DADOS**

MongoDB oferece flexibilidade, desempenho e escalabilidade, tornando-se uma escolha atraente para aplicativos que lidam com dados variados e têm requisitos de consulta complexos, como no caso de sistemas de gerenciamento de doações e distribuição de alimentos

Usar o MongoDB nesse caso específico, onde você está lidando com informações relacionadas a doações, estoque de alimentos e distribuição, oferece vários benefícios devido às características e funcionalidades do MongoDB.

O MongoDB oferece suporte a consultas complexas e aninhadas. Isso significa que você pode executar consultas para recuperar dados específicos dentro dos documentos, como buscar todas as doações de um doador específico ou encontrar todas as distribuições para uma determinada organização.

NoSQL permite que você adicione campos de dados de forma flexível sem a necessidade de esquemas rígidos. Isso é importante, pois os tipos de alimentos, informações do doador e outras informações podem variar amplamente.

Alinhamento de Dados: Usando documentos JSON, você pode aninhar informações relacionadas. Por exemplo, uma doação pode conter informações sobre o doador e os alimentos doados em um único documento, tornando a recuperação de dados eficiente.

Escalabilidade: NoSQL é altamente escalável horizontalmente, o que é crucial quando se lida com grandes volumes de dados em um sistema de banco de alimentos. Conforme o sistema cresce, você pode adicionar mais servidores para acomodar a carga.

Consultas e Indexação: Bancos de dados NoSQL geralmente oferecem recursos de consulta e indexação, permitindo que você pesquise e filtre dados de maneira eficiente.

 Desempenho: A estrutura de documentos JSON é eficiente para leitura e gravação de dados, especialmente quando os dados estão relacionados e frequentemente acessados juntos.

Evolução de Dados: Como a estrutura de dados é flexível, é mais fácil adaptar-se a mudanças nos requisitos sem alterar esquemas ou causar interrupções no sistema.