# Documentação – Sistema de Controle de Estoque do Supermercado das Três Glórias

## 1. Introdução

Este documento apresenta o desenvolvimento de um sistema web para controle de estoque de um supermercado, cujo objetivo é facilitar a gestão de produtos, pedidos e reposições. O sistema busca otimizar a comunicação entre o administrador e os responsáveis por setores, oferecendo uma interface intuitiva para o registro de produtos em falta e o acompanhamento dos pedidos.

## 2. Objetivo

O projeto tem como objetivo desenvolver uma aplicação web que permita:  
- Ao administrador, gerenciar produtos, setores, responsáveis e acompanhar os pedidos realizados;  
- Aos responsáveis por setor, informar produtos em falta ou com baixo estoque, editar pedidos já feitos (mediante justificativa) e manter atualizado o status de seus setores.

## 3. Justificativa

O controle manual de estoque em supermercados é suscetível a erros e atrasos, especialmente quando há múltiplos setores e responsáveis. Este sistema visa centralizar as informações e automatizar o fluxo de comunicação entre os responsáveis e o administrador, tornando o processo mais rápido, organizado e confiável.

## 4. Escopo do Sistema

O sistema contempla as seguintes funcionalidades:  
- Login e autenticação de usuários (admin e responsáveis);  
- Gerenciamento de produtos, setores e responsáveis;  
- Registro de produtos em falta ou com baixo estoque;  
- Criação e edição de pedidos pelos responsáveis, com justificativa em caso de alterações;  
- Visualização dos pedidos realizados pelo administrador;  
- Controle de status e histórico de pedidos.  
  
Observação: o administrador não aprova nem recusa pedidos — ele apenas visualiza o que foi solicitado para então realizar manualmente os pedidos externos de reposição.

## 5. Tecnologias Utilizadas

HTML5 / CSS3 / JavaScript – Estrutura e interatividade da interface web  
PHP 8.x – Lógica de back-end e manipulação dos dados  
Zend Framework – Estrutura MVC e gerenciamento de rotas  
Doctrine ORM – Abstração da camada de persistência utilizando arquivos JSON para armazenar informações de produtos, setores e pedidos  
Git / GitHub – Controle de versão e colaboração entre os desenvolvedores

## 6. Arquitetura do Sistema

O sistema segue o padrão MVC (Model-View-Controller), utilizando o Zend Framework para separar as camadas e o Doctrine ORM para gerenciar a persistência de dados. Em vez de um banco de dados relacional, o sistema utilizará arquivos JSON como forma de armazenamento local, garantindo leveza e simplicidade. Esses arquivos serão manipulados por meio do Doctrine, que fornecerá uma interface de abstração para leitura e escrita estruturada.  
  
- Model: representa entidades como Produto, Setor, Responsavel e Pedido;  
- View: interface do usuário desenvolvida com HTML, CSS e JavaScript;  
- Controller: coordena as ações dos usuários e a comunicação entre o front-end e o back-end.

## 7. Casos de Uso

Atores:  
- Administrador  
- Responsável por Setor  
  
Principais casos de uso:  
1. Login no sistema  
2. Cadastrar, editar e excluir produtos (Admin)  
3. Cadastrar setores e responsáveis (Admin)  
4. Registrar produtos em falta ou com baixo estoque (Responsável)  
5. Criar pedidos de reposição (Responsável)  
6. Editar pedidos já criados, com justificativa da alteração (Responsável)  
7. Visualizar todos os pedidos e seus status (Admin)

## 8. Requisitos Funcionais

RF01 – O sistema deve permitir o login de administrador e responsáveis.  
RF02 – O administrador deve poder cadastrar, editar e excluir produtos.  
RF03 – O administrador deve poder cadastrar setores e responsáveis.  
RF04 – O responsável deve poder registrar produtos em falta ou com baixo estoque.  
RF05 – O responsável deve poder criar novos pedidos de reposição.  
RF06 – O responsável deve poder editar pedidos existentes, informando uma justificativa.  
RF07 – O administrador deve poder visualizar todos os pedidos criados e suas alterações.  
RF08 – O sistema deve registrar a data, hora e justificativa de edições em pedidos.

## 9. Requisitos Não Funcionais

RNF01 – O sistema deve ser acessível via navegador web.  
RNF02 – Deve haver autenticação segura para login.  
RNF03 – O sistema deve registrar logs de ações críticas (edição de pedidos).  
RNF04 – A interface deve ser responsiva e intuitiva.  
RNF05 – O sistema deve ser compatível com navegadores modernos.  
RNF06 – O código deve seguir o padrão MVC e boas práticas do Zend Framework.

## 10. Armazenamento de Dados

O sistema não utilizará um banco de dados relacional como MySQL. Em vez disso, será utilizado o armazenamento em arquivos JSON, que conterão os dados de usuários, setores, produtos e pedidos. O Doctrine ORM será configurado para realizar a leitura, escrita e atualização desses arquivos, mantendo a estrutura de entidades e facilitando a manipulação dos dados.

## 11. Considerações Finais

O sistema de controle de estoque proposto visa melhorar a comunicação entre setores e administração de um supermercado, centralizando pedidos e registros de produtos em falta. A opção por utilizar arquivos JSON em conjunto com o Doctrine ORM mantém a estrutura do padrão MVC sem a necessidade de um banco de dados completo, tornando o projeto leve e adequado ao contexto acadêmico.