Documento de Projeto
Sistema de Estoque

Oaní da Silva da Costa
Dec 23, 2023

Introdução

Este documento tem como objetivo descrever o projeto de um sistema de estoque para uma rede de lojas. O sistema deve permitir o cadastro, alteração e exclusão de produtos no estoque central, bem como a consulta e movimentação dos mesmos entre as lojas. O sistema também deve fornecer relatórios e gráficos sobre o estado do estoque, as vendas realizadas e as notas fiscais emitidas.

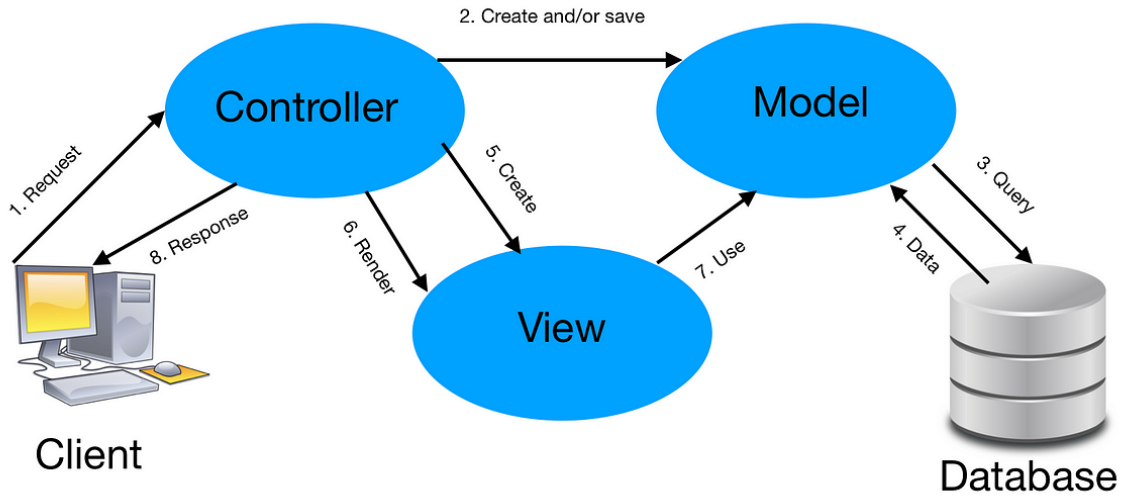
Requisitos Funcionais

Os requisitos funcionais do sistema são os seguintes:

- RF01: O sistema deve permitir o cadastro, alteração e exclusão de produtos no estoque central, informando o nome, a quantidade e o valor unitário de cada produto.
- RF02: O sistema deve permitir a consulta de produtos no estoque central

Arquitetura do Sistema

A arquitetura proposta para o sistema é baseada no padrão MVC (Model-View-Controller), que separa as camadas de apresentação, negócio e dados do sistema. A figura abaixo ilustra a arquitetura do sistema:



Retirado de: https://cdn-images-1.medium.com/v2/resize:fit:1200/1*YKM_9ifUTva23vBFHEgt3Q.png

A camada de apresentação é responsável por interagir com o usuário, exibindo as telas e os menus do sistema e recebendo as entradas e as saídas do usuário. A camada de apresentação será desenvolvida em ASP NET Core. Esse sistema parece, no atual estado do projeto, adequado devido à rapidez de desenvolvimento aliada a flexibilidade e escalabilidade do framework. No futuro, se a aplicação exigir mais do front, o front pode evoluir para aliar o uso de React ou outras tecnologias.

A camada de negócio é responsável por implementar as regras de negócio do sistema, validando as entradas do usuário, realizando os cálculos e as operações necessárias e fornecendo os dados para a camada de apresentação. A camada de negócio será desenvolvida em .Net 6, uma linguagem de programação orientada a objetos, portável e de alto nível.

A interface do sistema deve contemplar a história do usuário, permitindo que ele realize as operações de cadastro, alteração, exclusão e consulta de produtos no estoque. Para isso, é desejável que se desenvolva duas telas:

Uma para pesquisa dos dados e outra para inclusão/edição. A tela de pesquisa deve permitir que o usuário filtre os produtos por nome, categoria ou fornecedor, e selecione um produto para edição ou exclusão.

A tela de inclusão/edição deve permitir que o usuário informe ou altere os dados do produto, como nome, quantidade e valor unitário.

As telas devem ser simples, intuitivas e responsivas, seguindo as boas práticas de usabilidade e acessibilidade.

A camada de dados é responsável por persistir e recuperar os dados do sistema, utilizando um sistema de banco de dados relacional. A camada de dados será desenvolvida em SQL Server que tem fácil e robusta integração com .NET.

Entre as camadas de negócio e de dados, não será usado nenhum ORM pois entende-se que, no nível atual, o projeto não necessita. Porém o Entity Framework seria uma recomendação para quando o projeto crescer.

Para a geração de relatórios e gráficos, será utilizado o ReportViewer, um controle da própria Microsoft, integrado ao Visual Studio.NET que permite a geração de relatórios. O ReportViewer pode renderizar dados de forma eficiente, realizando operações como filtragem, ordenação, agrupamento e agregação. O ReportViewer pode criar documentos em diversos formatos, como PDF, HTML, XML, CSV, entre outros, a partir de dados provenientes de diversas fontes, como bancos de dados, arquivos e objetos .NET, etc.

Uma outra opção para a geração de relatórios e gráficos é o Crystal Reports, um framework de geração de relatórios que permite a criação de documentos em diversos formatos, como PDF, HTML, XML, CSV, entre outros, a partir de dados provenientes de diversas fontes.

Relevante é lembrar que O desenvolvimento do projeto deve seguir os princípios SOLID, que são um conjunto de boas práticas para a criação de código limpo, coeso, acoplado e fácil de manter. Os princípios SOLID são:

Single Responsibility Principle (SRP): cada classe deve ter uma única responsabilidade e um único motivo para mudar.

Open/Closed Principle (OCP): as classes devem ser abertas para extensão, mas fechadas para modificação.

Liskov Substitution Principle (LSP): as subclasses devem ser substituíveis pelas suas superclasses, sem quebrar o comportamento esperado.

Interface Segregation Principle (ISP): as interfaces devem ser pequenas e específicas, evitando dependências desnecessárias.

Dependency Inversion Principle (DIP): as classes devem depender de abstrações e não de implementações concretas.

O desenvolvimento do projeto também será baseado nos conceitos dos Objetos Elegantes, propostos pelo autor Yegor Bugayenko. Os Objetos Elegantes são um estilo de programação orientada a objetos que segue princípios como: não há setters, apenas construtores públicos e foco na imutabilidade. Esses princípios visam tornar o código mais coeso, encapsulado, expressivo e fácil de testar. Além disso, em horas oportunas,

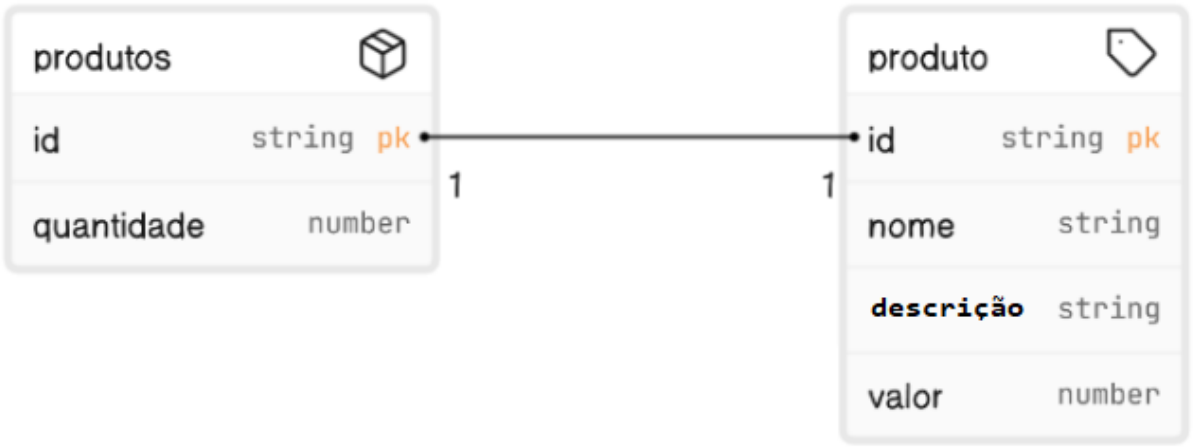
serão usados conceitos da programação funcional, que trata a computação como uma avaliação de funções matemáticas puras, evitando estados ou dados mutáveis e efeitos colaterais. A programação funcional pode trazer benefícios como maior legibilidade, modularidade, reusabilidade e paralelismo.

Modelo de Dados

O modelo de dados do sistema é composto pelas seguintes tabelas:

- Produtos: armazena os dados dos produtos do estoque central, como a quantidade de cada produto disponível. Possui uma chave primária composta (Id e ProdutoId) e uma chave estrangeira (ProdutoId), que referencia a tabela Produto.
- Produto: armazena os dados dos produtos, como nome e valor unitário. Possui uma chave primária (Id).

A figura abaixo ilustra o modelo de dados do sistema:



Considerações Finais

Este documento apresentou o projeto de um sistema de estoque para uma rede de lojas, descrevendo os requisitos funcionais, a arquitetura, o modelo de dados e as tecnologias utilizadas. O sistema visa facilitar o controle e a gestão do estoque. No futuro, das vendas e das notas fiscais, proporcionando maior eficiência e qualidade para o negócio. Esse documento é apenas uma proposta inicial e pode ser alterado conforme as necessidades e especificações do cliente.

Apêndice A: Mais entidades

Abaixo estão algumas considerações sobre possíveis novas entidades do projeto. Interessante lembrar de, conforme o projeto evolui, seria interessante considerar os custos de desenvolver um sistema tao completo ou contratar um ERP ou módulos de um dado a complexidade e custos que o projeto assumiria.

Produto: armazena os dados dos produtos do estoque, como nome, quantidade e valor unitário. Possui uma chave primária (Id) e duas chaves estrangeiras (CategoriaId e FornecedorId), que referenciam as tabelas Categoria e Fornecedor, respectivamente.

Categoria: armazena os dados das categorias de produtos, como nome e descrição. Possui uma chave primária (Id).

Fornecedor: armazena os dados dos fornecedores de produtos, como nome, endereço, telefone e e-mail. Possui uma chave primária (Id).

Estoque: armazena os dados do estoque de cada loja, como a quantidade de cada produto disponível. Possui uma chave primária composta (ProdutoId e LojaId) e duas chaves estrangeiras (ProdutoId e LojaId), que referenciam as tabelas Produto e Loja, respectivamente.

Loja: armazena os dados das lojas da rede, como nome, endereço, telefone e e-mail. Possui uma chave primária (Id).

Movimentacao: armazena os dados das movimentações de produtos entre o estoque central e as lojas, como a quantidade, a data e a origem e o destino da movimentação. Possui uma chave primária (Id) e três chaves estrangeiras (ProdutoId, OrigemId e DestinoId), que referenciam as tabelas Produto, Loja e Loja, respectivamente.

Venda: armazena os dados das vendas realizadas nas lojas, como a quantidade, o valor, a data e o cliente da venda. Possui uma chave primária (Id) e duas chaves estrangeiras (ProdutoId e LojaId), que referenciam as tabelas Produto e Loja, respectivamente.

Cliente: armazena os dados dos clientes das lojas, como nome, endereço, telefone e e-mail. Possui uma chave primária (Id).

NotaFiscal: armazena os dados das notas fiscais emitidas nas lojas, como os dados do produto, do cliente, da loja e da venda. Possui uma chave primária (Id) e uma chave estrangeira (VendaId), que referencia a tabela Venda.

Apêndice B: Requisitos funcionais futuros.

- RF03: O sistema deve permitir a seleção de um produto no estoque central para edição ou exclusão, exibindo uma tela específica para cada operação.
- RF04: O sistema deve permitir o cadastro de novos produtos no estoque central, exibindo uma tela específica para essa operação.
- RF05: O sistema deve permitir o cadastro, alteração e exclusão de categorias de produtos, informando o nome e a descrição de cada categoria.
- RF06: O sistema deve permitir o cadastro, alteração e exclusão de fornecedores de produtos, informando o nome, o endereço, o telefone e o e-mail de cada fornecedor.
- RF07: O sistema deve permitir a consulta de categorias e fornecedores de produtos, filtrando por nome ou descrição.
- RF08: O sistema deve permitir a seleção de uma categoria ou fornecedor de produtos para edição ou exclusão, exibindo uma tela específica para cada operação.
- RF09: O sistema deve permitir o cadastro de novas categorias ou fornecedores de produtos, exibindo uma tela específica para essa operação.
- RF10: O sistema deve permitir a consulta de produtos no estoque de cada loja, filtrando por nome, categoria ou fornecedor.
- RF11: O sistema deve permitir a movimentação de produtos entre o estoque central e as lojas, informando a quantidade, a data e a origem e o destino da movimentação.
- RF12: O sistema deve permitir a consulta de movimentações de produtos, filtrando por produto, data, origem ou destino.
- RF13: O sistema deve permitir a venda de produtos nas lojas, informando a quantidade, o valor, a data e o cliente da venda.
- RF14: O sistema deve permitir a consulta de vendas realizadas nas lojas, filtrando por produto, data, valor ou cliente.
- RF15: O sistema deve permitir a emissão de notas fiscais para as vendas realizadas nas lojas, informando os dados do produto, do cliente, da loja e da venda.
- RF16: O sistema deve permitir a consulta de notas fiscais emitidas nas lojas, filtrando por produto, cliente, loja ou data.
- RF17: O sistema deve permitir a geração de relatórios sobre o estoque, as vendas e as notas fiscais, informando os dados agregados e detalhados de cada item.
- RF18: O sistema deve permitir a geração de gráficos sobre o estoque, as vendas e as notas fiscais, exibindo os dados em forma de barras, linhas, pizza ou outros formatos.
- RF19: Filtrar na lista de produtos por por nome, categoria ou fornecedor.