SISTEMA DE MANUTENÇÃO INDUSTRIAL - ALMOXARIFADO

1. Escolha do Estilo Arquitetural

O sistema de manutenção industrial — almoxarifado será desenvolvido utilizando o padrão arquitetural **MVC (Model-View-Controller).**

- **Model (Modelo):** Representa os dados do sistema e regras de negócio, incluindo fornecedores, materiais, notas de entrada e requisições.
- **View (Visão):** Responsável pela apresentação no terminal, exibindo menus, prompts e mensagens de confirmação/erro.
- **Controller (Controlador):** Intermediário entre View e Model, coordenando o fluxo do sistema, recebendo entradas do usuário e atualizando o Model conforme necessário.

2. Justificativa Técnica

A escolha do MVC se justifica pelos seguintes fatores:

- Separação de responsabilidades:
  - Permite que a lógica de negócio, a interface do usuário e o controle do fluxo sejam desenvolvidos de forma independente.
  - Facilita alterações futuras em qualquer camada sem impactar as outras.

- Manutenibilidade:
  - Mudanças na apresentação (menus e mensagens no terminal) não afetam as regras de negócio.
  - Novas funcionalidades podem ser adicionadas com menor risco de quebrar o sistema existente.

- Alinhamento com requisitos não funcionais:
  - O padrão MVC suporta usabilidade, fornecendo feedback claro ao usuário (View).

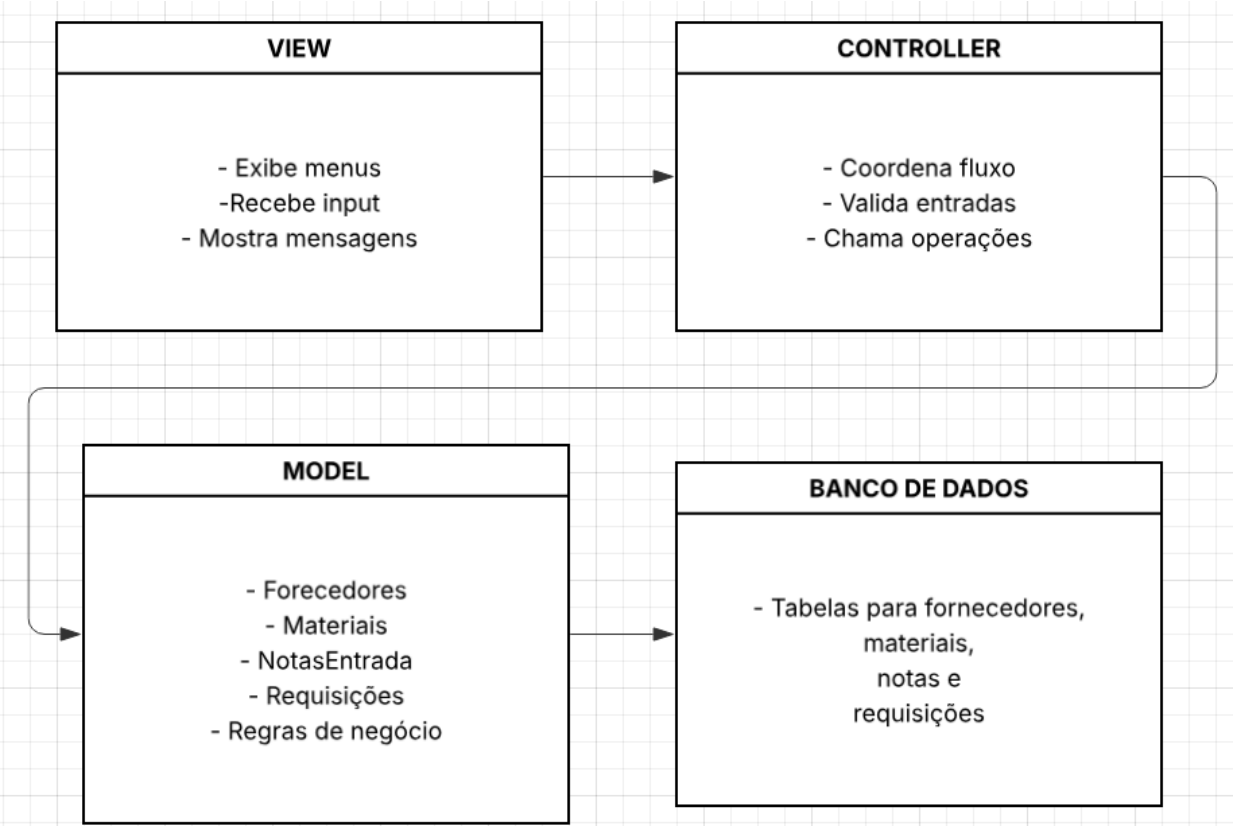
    - Permite confiabilidade e consistência, garantindo que operações críticas no Model sejam executadas corretamente antes de atualizar o View.
    - Facilita manutenibilidade, uma vez que cada camada pode ser organizada em módulos separados.

- Escalabilidade futura:
    - Caso o sistema evolua para interfaces gráficas ou web, o MVC permite reaproveitar grande parte do Model e Controller.

3. Rationale Arquitetural (Trade-offs e Benefícios)

Aspecto	**Benefício**	**Trade-off / Limitação**
Separação de camadas	Facilita manutenção, teste e evolução do sistema	Requer disciplina no design para evitar acoplamento
Reutilização de componentes	Reaproveitamento de Model e Controller em diferentes interfaces	Inicialmente pode gerar mais arquivos e classes
Simplicidade	Adequado para protótipos em terminal	Não é ideal para microsserviços ou alta complexidade
Evolução futura	Facilita integração com sistemas ERP ou interfaces gráficas	Pode precisar de adaptação caso migre para web MVC ou API

4. Visão Geral da Arquitetura Proposta



Explicação do fluxo:

- O usuário interage com o View através do terminal.
- O Controller recebe a entrada, valida dados e chama métodos do Model.
- O Model atualiza o banco de dados e aplica as regras de negócio.
- O View exibe mensagens de sucesso ou erro para o usuário.