UNIVERSIDADE DO ESTADO DA BAHIA - UNEB
DEPARTAMENTO DE CIÊNCIAS EXATAS E DA TERRA I

CONTROLE DE ALMOXARIFADO NA CONSTRUÇÃO CIVIL

Trabalho apresentado a UNEB para obtenção de nota na disciplina Computação Aplicada à Engenharia

Discente: Andreza Pereira; Jailmária Marques
Docente: Robson Marinho

Salvador - BA
2024

Sumário

1 RESUMO

Este estudo tem como objetivo aprofundar a análise da gestão de almoxarifados na indústria da construção civil, com o propósito de identificar e propor melhorias eficientes que otimizem tanto o armazenamento quanto o controle de materiais. Reconhecendo a importância crucial do gerenciamento adequado do almoxarifado, busca-se assegurar não apenas a disponibilidade oportuna de materiais, mas também a minimização de desperdícios e o aumento da produtividade no cenário da construção civil. É notório que a falta de materiais pode acarretar atrasos significativos nas obras, sendo essencial sanar essa questão por meio de estratégias eficazes de gestão de almoxarifados.

Neste trabalho, serão cuidadosamente examinados os desafios enfrentados e as práticas comuns adotadas na organização de almoxarifados na indústria da construção. Com base nessa análise, serão apresentadas não apenas recomendações, mas também estratégias concretas que visam aprimorar o fluxo de materiais e fortalecer o controle de estoque. Tais melhorias propostas têm o potencial não apenas de resolver problemas imediatos, como a escassez de materiais, mas também de promover uma gestão mais eficiente e sustentável dos recursos, contribuindo assim para a otimização dos processos e para o sucesso geral dos projetos na construção civil.

Além dos aspectos relacionados à gestão de almoxarifados na construção civil, este estudo também explora o potencial da linguagem de programação Python para otimizar processos de controle de estoque e gestão de materiais.

Python que é uma linguagem aberta de programação que oferece uma ampla gama de bibliotecas e ferramentas que podem ser aplicadas no contexto da construção civil, desde a automação de tarefas rotineiras até a análise de dados para previsão de demanda e otimização de inventário.

Dessa forma o python consegue ser utilizado para análise de dados do estoque, que permite identificar padrões de consumo e tendências que podem nos auxiliar a prever demandas do almoxarifado. Além disso, conseguimos realizar dashboards para visualizar a operação dos almoxarifado

Palavras-chave: Almoxarifado, Construção Civil, Organização, Gerenciamento de Materiais, Controle de Estoque.

2 INTRODUÇÃO

O setor da construção civil requer um eficiente sistema de organização de almoxarifados para garantir que os materiais necessários estejam disponíveis quando necessário, minimizar desperdícios, evitar atrasos e melhorar a eficiência geral do projeto. No entanto, muitas empresas enfrentam desafios na gestão de seus almoxarifados, como dificuldades de localização de materiais, falta de controle de estoque e perdas financeiras decorrentes de má gestão. Portanto, este estudo visa identificar e analisar as práticas comuns de organização de almoxarifados na construção civil, bem como propor estratégias de melhoria para otimizar o gerenciamento de materiais.

Uma das principais características do almoxarifado na construção civil é a diversidade de materiais e equipamentos armazenados. Desde materiais básicos, como cimento, areia e tijolos, até elementos mais complexos, como tubulações, ferragens e sistemas elétricos, é preciso gerenciar uma ampla variedade de itens. Além disso, a gestão de estoque também deve considerar as quantidades necessárias para cada etapa da obra, a fim de evitar excessos ou falta de materiais.

Outro aspecto importante no almoxarifado da construção civil é o controle rigoroso dos itens recebidos e utilizados. É necessário registrar e acompanhar a entrada e saída de materiais, verificando a conformidade com as especificações técnicas, a qualidade e a quantidade. A adoção de sistemas de inventário, etiquetagem e código de barras pode facilitar o controle e a rastreabilidade dos materiais, permitindo uma gestão mais precisa e eficiente.

Por meio de um planejamento eficiente, é possível otimizar a logística de recebimento, armazenamento e distribuição dos materiais, garantindo um fluxo contínuo e adequado de suprimentos. Isso contribui para reduzir desperdícios, evitar a falta de materiais em momentos críticos e assegurar a qualidade das construções.

3 OBJETIVO

O objetivo geral deste projeto é analisar as práticas atuais de controle de almoxarifado na construção civil e desenvolver um modelo de referência para a gestão desses dados na área. Os objetivos especícos são:

- Avaliar as estratégias de armazenamento de dados utilizadas atualmente no setor;
- Propor um modelo referência para catalogação dos itens presentes no almoxarifado na construção civil.

4 METODOLOGIA DA PESQUISA

A pesquisa será realizada por meio de uma abordagem mista, envolvendo revisão bibliográfica, observação direta em canteiros de obras e entrevistas com profissionais da área. A revisão bibliográfica abordará conceitos-chave relacionados à gestão de almoxarifados, boas práticas de organização, métodos de controle de estoque e tecnologias utilizadas. A observação direta permitirá a coleta de dados sobre a atual organização do almoxarifado em

diferentes canteiros de obras. As entrevistas com profissionais da área, como gestores de projetos e almoxarifes, fornecerão informações sobre desafios enfrentados e possíveis melhorias.

5 PYTHON AUXILIANDO NO CONTROLE DE ALMOXARIFADO

Python, uma linguagem de programação versátil e de alto nível, pode desempenhar um papel significativo no controle de almoxarifado na construção civil, oferecendo uma série de benefícios e recursos para otimizar a gestão dos materiais. Através de suas bibliotecas e recursos, o Python pode auxiliar em várias etapas do controle de estoque, desde o registro de entrada e saída de materiais até a análise de dados e a automação de processos.
Uma das formas pelas quais o Python pode ajudar é na criação de um sistema de inventário digitalizado para o almoxarifado. Sendo possivel desenvolver uma estrutura de dados capaz de registrar e acompanhar os materiais em estoque. Isso simplifica a tarefa de controlar as quantidades, especificações técnicas e datas de validade dos itens, facilitando a identificação de necessidades de reposição e evitando a falta ou o excesso de estoque.
Outra funcionalidade do Python é a integração com sistemas de gestão empresarial (ERP). Através de bibliotecas e APIs, é possível conectar o almoxarifado ao sistema central da empresa, garantindo uma troca de informações fluida e precisa. Isso possibilita o compartilhamento de dados em tempo real, agiliza o fluxo de informações entre diferentes setores e facilita a tomada de decisões estratégicas baseadas em informações atualizadas.

Comp cód. 2.py

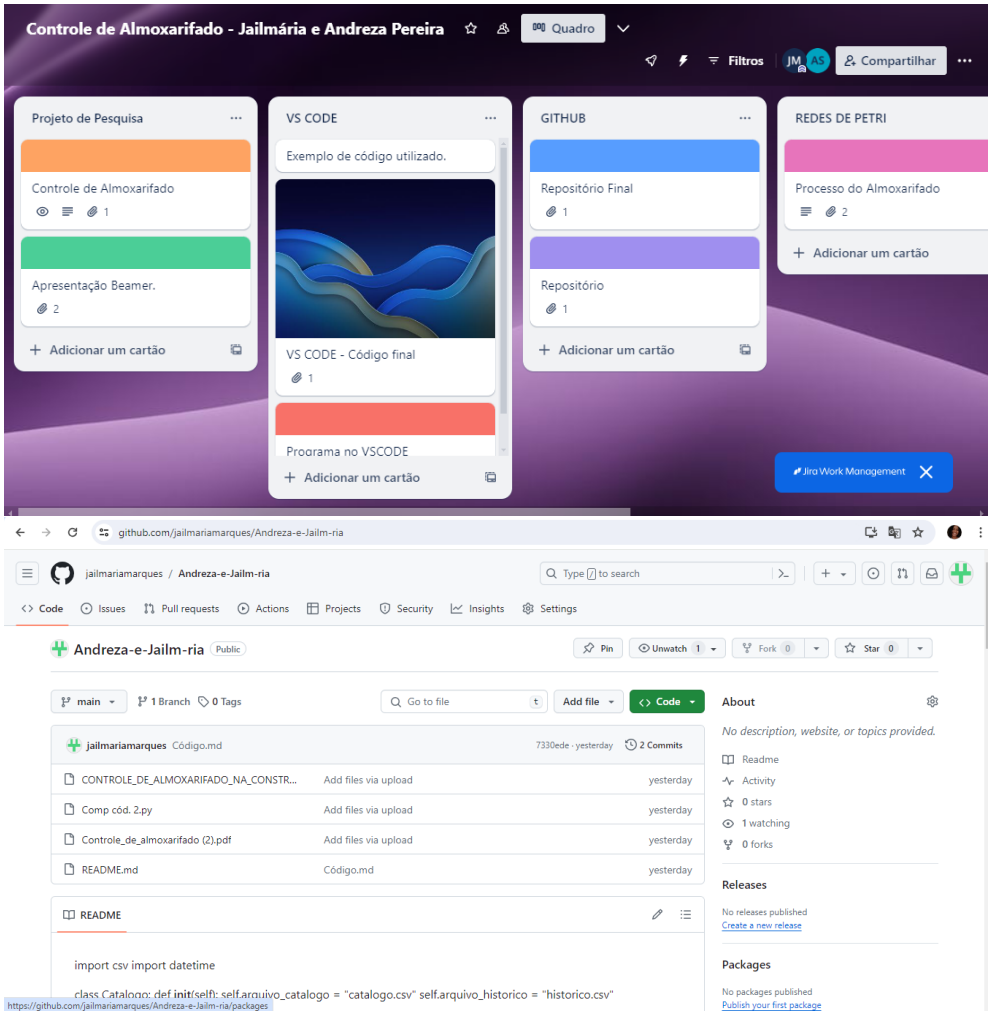
```python
import csv
import datetime

class Catalogo:
    def __init__(self):
        self.arquivo_catalogo = "catalogo.csv"
        self.arquivo_historico = "historico.csv"

    def adicionar_item(self):
        codigo = input("Digite o código do item: ")
        descricao = input("Digite a descrição do item: ")
        quantidade = input("Digite a quantidade disponível: ")

        with open(self.arquivo_catalogo, "a", newline="") as arquivo:
            writer = csv.writer(arquivo)
            writer.writerow([codigo, descricao, quantidade])

        print("Item adicionado com sucesso!")

    def pesquisar_item(self):
        codigo = input("Digite o código do item: ")

        try:
            with open(self.arquivo_catalogo, "r") as arquivo:
                reader = csv.reader(arquivo)
                for linha in reader:
                    if linha[0] == codigo:
```

```
                for linha in reader:
                    if linha[0] == codigo:
                        print("Descrição:", linha[1])
                        print("Quantidade disponível:", linha[2])
                        return
            print("Item não encontrado.")
        except FileNotFoundError:
            print("Arquivo do catálogo não encontrado.")
        except Exception as e:
            print("Ocorreu um erro:", e)

    def atualizar_quantidade(self):
        codigo = input("Digite o código do item: ")
        nova_quantidade = input("Digite a nova quantidade disponível: ")

        try:
            linhas_atualizadas = []
            with open(self.arquivo_catalogo, "r") as arquivo:
                reader = csv.reader(arquivo)
                for linha in reader:
                    if linha[0] == codigo:
                        linha[2] = nova_quantidade
                    linhas_atualizadas.append(linha)

            with open(self.arquivo_catalogo, "w", newline="") as arquivo:
                writer = csv.writer(arquivo)
                writer.writerows(linhas_atualizadas)

            print("Quantidade atualizada com sucesso!")
        except FileNotFoundError:
            print("Arquivo do catálogo não encontrado.")
        except Exception as e:
            print("Ocorreu um erro:", e)

    def exibir_catalogo(self):
        try:
            with open(self.arquivo_catalogo, "r") as arquivo:
                reader = csv.reader(arquivo)
                for linha in reader:
                    print("Código:", linha[0])
                    print("Descrição:", linha[1])
                    print("Quantidade disponível:", linha[2])
                    print()
        except FileNotFoundError:
            print("Arquivo do catálogo não encontrado.")
        except Exception as e:
            print("Ocorreu um erro:", e)

    def exibir_historico(self):
        try:
            with open(self.arquivo_historico, "r") as arquivo:
                reader = csv.reader(arquivo)
                for linha in reader:
                    data = datetime.datetime.strptime(linha[0], "%Y-%m-%d %H:%M:%S")
                    descricao = linha[1]
                    quantidade = linha[2]
                    print("Data e hora:", data)
                    print("Descrição:", descricao)
                    print("Quantidade:", quantidade)
                    print()
        except FileNotFoundError:
            print("Arquivo do histórico não encontrado.")
        except Exception as e:
            print("Ocorreu um erro:", e)

    def menu(self):
        while True:
            print("1. Adicionar item")
            print("2. Pesquisar item")
            print("3. Atualizar quantidade")
            print("4. Exibir catálogo")
```

```
            print("4. Exibir catalogo")
            print("5. Exibir histórico")
            print("6. Sair")

            opcao = input("Digite a opção desejada: ")

            if opcao == "1":
                self.adicionar_item()
            elif opcao == "2":
                self.pesquisar_item()
            elif opcao == "3":
                self.atualizar_quantidade()
            elif opcao == "4":
                self.exibir_catalogo()
            elif opcao == "5":
                self.exibir_historico()
            elif opcao == "6":
                break
            else:
                print("Opção inválida. Tente novamente.")

            print()

# Cria uma instância da classe Catalogo
catalogo = Catalogo()
```

DESCRIÇÃO DO CÓDIGO PYTHON: 1. As primeiras linhas importam os módulos csv e datetime. O csv é usado para trabalhar com arquivos CSV, enquanto o datetime é usado para manipular datas e horas. 2. Em seguida é definida a classe 'Catalogo'. A classe é um tipo de estrutura que agrupa dados e funções relacionadas. 3. O método 'init' é um método especial, que é executado automaticamente ao criar um objeto da classe. Neste caso, ele inicializa os atributos 'arquivo catalogo' e 'arquivo historico' com os nomes dos arquivos CSV que serão usados para armazenar o catálogo de itens ao histórico de consumo, respectivamente. 4. O método 'adicionar item' permite ao usuário adicionar um novo item ao catálogo. Em seguida, ele abre o arquivo 'arquivo catalogo' em modo de append ("a") e utiliza o objeto csv.writer para escrever uma nova linha com os dados fornecidos pelo usuário. 5. O método 'pesquisar item' permite ao usuário pesquisar um item no catálogo. Ele solicita ao usuário que insira o código do item a ser pesquisado. Em seguida, abre o arquivo 'arquivo catalogo' em modo de leitura ("r") e utiliza o objeto csv.reader para percorrer as linhas do arquivo. 6. O método 'atualizar quantidade' permite ao usuário atualizar a quantidade disponível de um item. Ele solicita ao usuário que insira o código do item e a nova quantidade disponível. 7. O método 'exibir catalogo' exibe o catálogo de itens. Ele abre o arquivo 'arquivo catalogo' em modo de leitura ("r") e utiliza o objeto csv.reader para percorrer as linhas do arquivo. Em seguida, exibe o código, a descrição e a quantidade disponível de cada item. 8. O método 'exibir historico' exibe o histórico de consumo. Para cada linha, ele converte a data e a hora (que estão no formato de string) em um objeto datetime usando o método strptime. Em seguida, exibe a data e a hora, a descrição e a quantidade do consumo. 9. Esse método 'menu' exibe um menu de opções e permite ao usuário interagir com o programa. Ele utiliza um loop 'while True' para exibir continuamente o menu até que a opção "6"seja escolhida para sair. 10. As linhas finais criam uma instância da classe Catalogo chamada 'catalogo' e, em seguida, chamam o método menu() dessa instância para iniciar a execução do programa.

6 DESENVOLVIMENTO DO PROJETO (Quadros do Trello e Github)



7 CONCLUSÃO

O código apresentado propôs a otimização e a melhora da eficiência em operações de armazenamento, controle de estoque e materiais. O funcionamento do sistema gera uma melhor gestão do fluxo de materiais, facilitando o rastreamento. Além disso, a precisão dos registros no estoque é aprimorada, ocasionando na redução de erros e minimização de perdas.

Referências

[1] N. B. d. OLIVEIRA, B. S. d. SILVA, C. C. d. SOUZA, I. C. d. S. PEREIRA, and D. T. FRANCO, "Gestão de estoques: otimização no almoxarifado de empresas de materiais de construção," 2022.

[2] E. D. Sandrini, "Controle de almoxarifado através da otimização da gestão de estoques em uma construtora," B.S. thesis, Universidade Tecnológica Federal do Paraná, 2017.

[3]
(1) (2)
(3)