Controle de Almoxarifado na Construção Civil

Jailmária Marques e Andreza Pereira.

Computação aplicada à engenharia.

1.Introdução

- O controle de almoxarifado desempenha um papel fundamental na eficiência e sucesso dos projetos na construção civil. No contexto da construção civil, o controle de almoxarifado refere-se ao processo de rastrear, organizar e controlar os materiais utilizados em uma obra

2.Objetivos

- Objetivo Geral: analisar práticas de controle de almoxarifado e desenvolver um modelo referência com base nisso.
- Objetivos específicos: controle de entrada e saída de materiais, gestão de estoque e reposição facilitar a rastreabilidade e controle de validade, otimizar o tempo de obra.

3.Benefícios do Controle de Almoxarifado

- Redução de desperdícios
- Melhor aproveitamento dos recursos
- Evita paralisações de obra por falta de materiais
- Planejamento mais eficiente
- Redução de custos e aumento da lucratividade

4.Metodologia de pesquisa

- A pesquisa foi realizada por meio de uma revisão bibliográfica que abordou tópicos como:
- Conceitos chave em relação à administração de almoxarifados;
- Práticas mais adequadas de organização e planejamento;
- Métodos recomendados de controle e estoque.

5.Funcionamento do código Python

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)
```

```
            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)
```

```
            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 catálogo")
            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()
```