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.Gestão do Projeto

- A ferramenta Trello foi utilizada para gerenciar o projeto de pesquisa de maneira eficiente e colaborativa.
- O GitHub foi fundamental para o gerenciamento e a colaboração no projeto de pesquisa, especialmente no controle de versões de documentos e códigos.

5.Evolução

- Nesse momento foi adicionado a possibilidade do nosso programa disparar emails automáticos para equipe de compras, informando um baixo estoque, dessa forma, aumentamos a integração entre os funcionários responsáveis pela gestão do almoxarifado e o time de engenharia montante.

6.Funcionamento do código Python

```python
import csv
import datetime
import imaplib
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.parser import BytesParser
from email.policy import default


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: ")
```

```python
    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:
                        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)

```

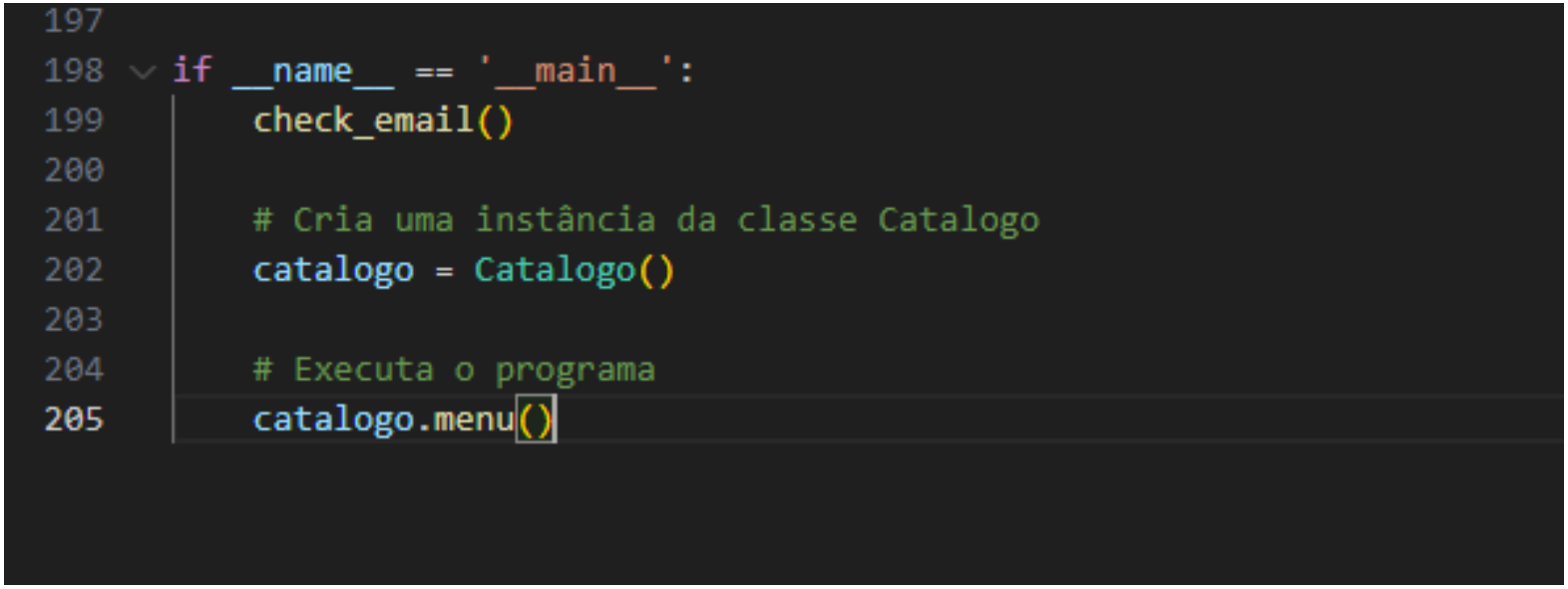
```
            with open(self.arquivo_catalogo, "w", newline="") as arquivo:
                writer = csv.writer(arquivo)
                writer.writerows(linhas_atualizadas)

            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 remover_item(self):
        codigo = input("Digite o código do item a ser removido: ")
        item_encontrado = False

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

            if item_encontrado:
                with open(self.arquivo_catalogo, "w", newline="") as arquivo:
                    writer = csv.writer(arquivo)
```

```
                    writer.writerows(linhas_atualizadas)
                print("Item removido com sucesso!")
            else:
                print("Item não encontrado.")
        except FileNotFoundError:
            print("Arquivo do catálogo 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("5. Exibir histórico")
            print("6. Remover item")
            print("7. 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":
```

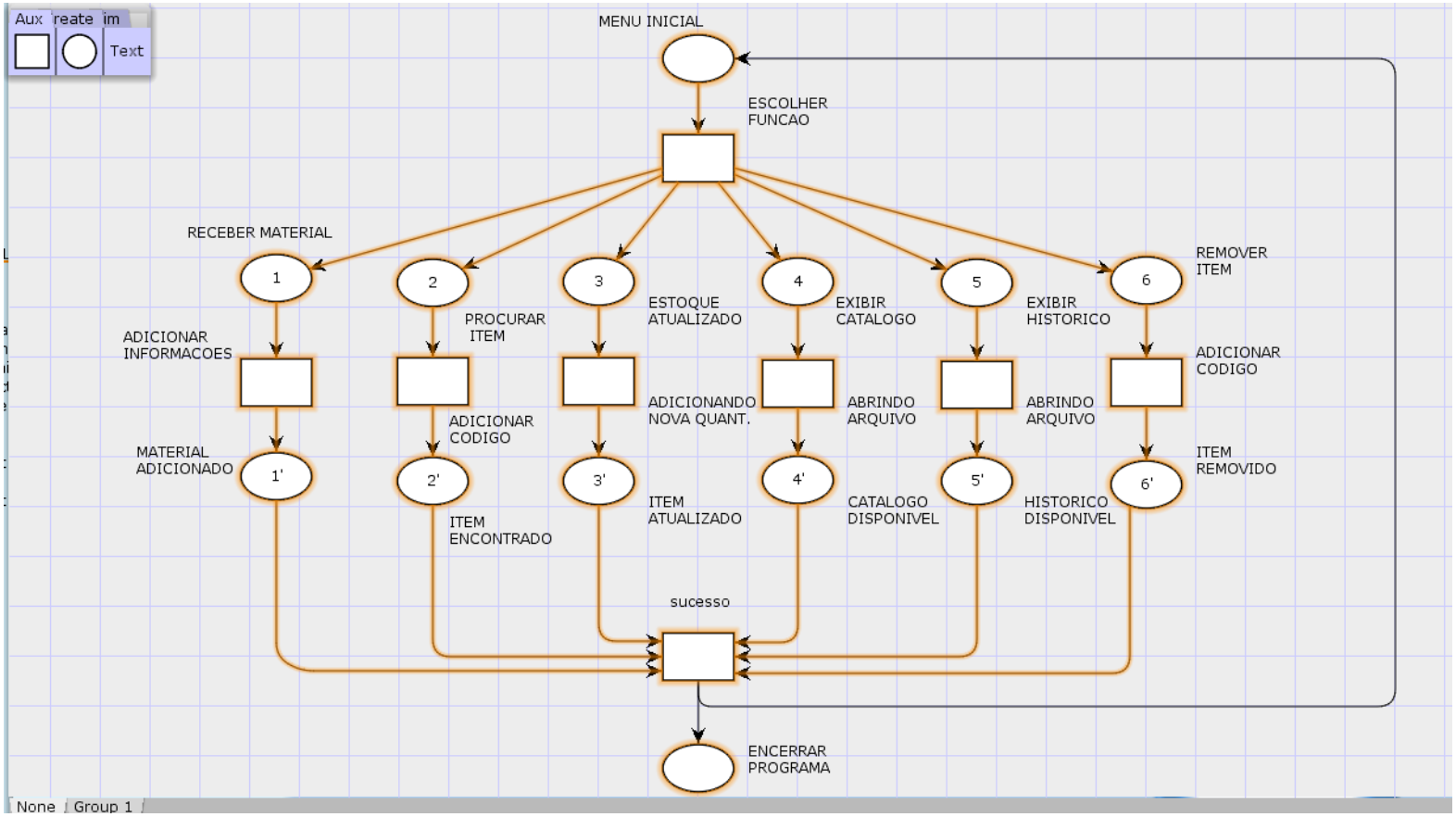
```
124         def menu(self):
143                     self.exibir_catalogo()
144                 elif opcao == "5":
145                     self.exibir_historico()
146                 elif opcao == "6":
147                     self.remover_item()
148                 elif opcao == "7":
149                     break
150                 else:
151                     print("Opção inválida. Tente novamente.")
152 
153                 print()
154 
155 
156 # Configurações do e-mail
157 IMAP_SERVER = 'imap.exemplo.com'
158 IMAP_PORT = 993
159 SMTP_SERVER = 'smtp.exemplo.com'
160 SMTP_PORT = 587
161 EMAIL_ACCOUNT = 'seu_email@exemplo.com'
162 PASSWORD = 'sua_senha'
163 FORWARD_TO_EMAIL = 'destinatario@exemplo.com'
164 AVISO = "Aviso: Almoxarifado com capacidade menor que 30%"
165 
166 
167 def check_email():
168     mail = imaplib.IMAP4_SSL(IMAP_SERVER, IMAP_PORT)
169     mail.login(EMAIL_ACCOUNT, PASSWORD)
170     mail.select('inbox')
171     status, messages = mail.search(None, '(UNSEEN)')
```

```python
def check_email():
    mail = imaplib.IMAP4_SSL(IMAP_SERVER, IMAP_PORT)
    mail.login(EMAIL_ACCOUNT, PASSWORD)
    mail.select('inbox')
    status, messages = mail.search(None, '(UNSEEN)')
    email_ids = messages[0].split()

    for email_id in email_ids:
        status, msg_data = mail.fetch(email_id, '(RFC822)')
        raw_email = msg_data[0][1]
        msg = BytesParser(policy=default).parsebytes(raw_email)

        forward_email(msg)

    mail.logout()


def forward_email(original_msg):
    msg = MIMEMultipart()
    msg['From'] = EMAIL_ACCOUNT
    msg['To'] = FORWARD_TO_EMAIL
    msg['Subject'] = "Fwd: " + original_msg['Subject']
    body = AVISO + "\n\n" + original_msg.get_body(preferencelist=('plain',)).get_content()

    msg.attach(MIMEText(body, 'plain'))
    with smtplib.SMTP(SMTP_SERVER, SMTP_PORT) as server:
        server.starttls()
        server.login(EMAIL_ACCOUNT, PASSWORD)
        server.sendmail(EMAIL_ACCOUNT, FORWARD_TO_EMAIL, msg.as_string())

```

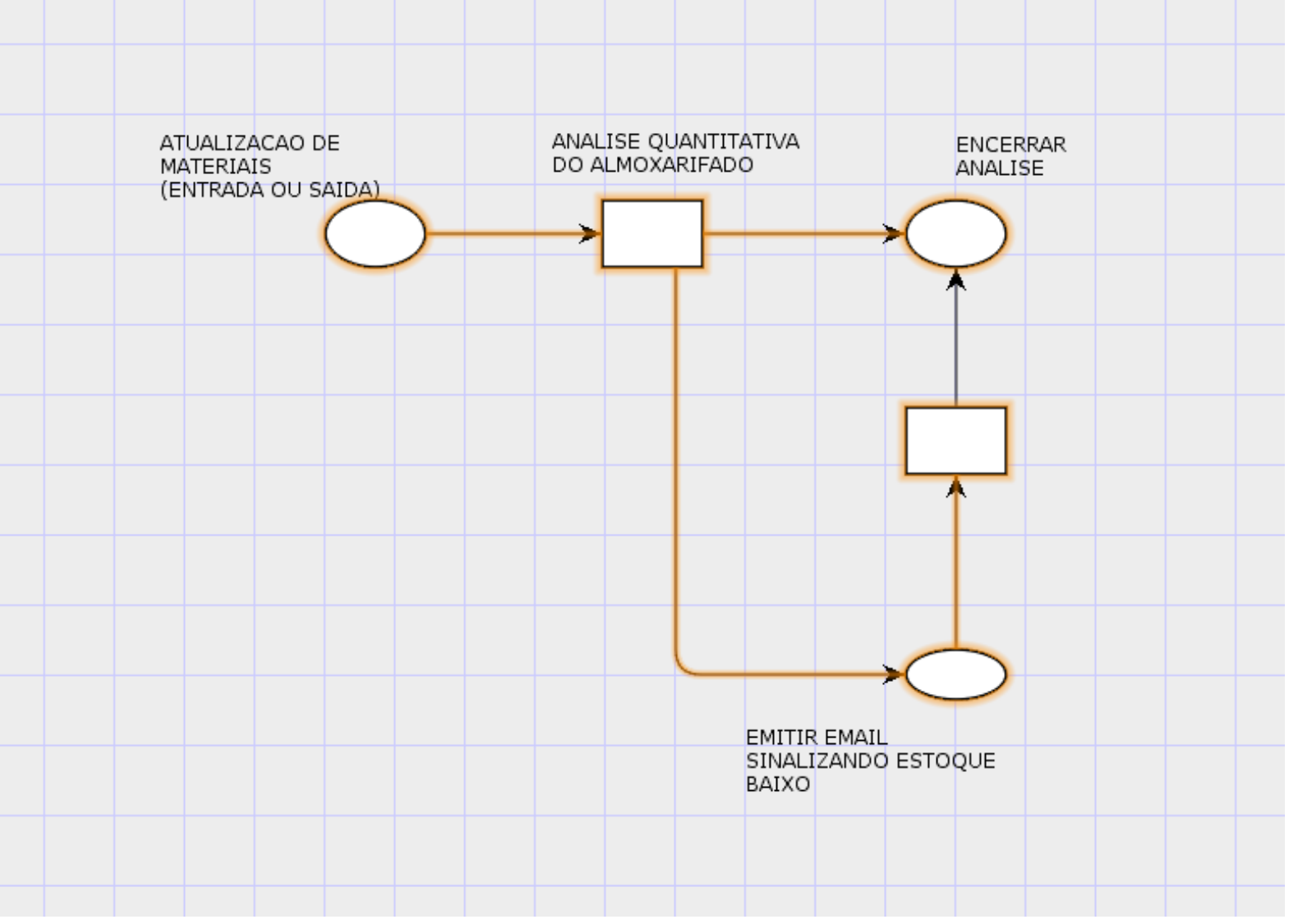

197
198 if __name__ == '__main__':
199 check_email()
200
201 # Cria uma instância da classe Catalogo
202 catalogo = Catalogo()
203
204 # Executa o programa
205 catalogo.menu()

7. Rede de Petri

- A modelagem através de redes de Petri ofereceu uma representação clara e detalhada do fluxo de materiais, ajudando a identificar gargalos e a analisar a eficiência das operações.



7. Rede de Petri



Conclusões Finais

Este estudo aprofundou-se na análise das práticas atuais de gerenciamento de estoque, identificando desafios comuns e propondo melhorias que visam otimizar tanto o armazenamento quanto o controle de materiais. A adoção de tecnologias como a linguagem de programação Python e a rede de Petri mostrou-se promissora na automação e análise de processos, permitindo uma gestão mais precisa e eficiente.

Referências

- E. D. Sandrini, “Controle de almoxarifado atrav es da otimiza c ao da gest ao de estoques em uma construtora,” B.S. thesis, Universidade Tecnol ogica Federal do Paran a, 2017.
- N. B. d. OLIVEIRA, B. S. d. SILVA, C. C. d. SOUZA, I. C. d. S. PEREIRA, and D. T.FRANCO, “Gest ao de estoques: otimiza c ao no almoxarifado de empresas de materiais de constru c ao,” 2022