In [1]:
```
print("Ambiente funcionando!")
```

```
Ambiente funcionando!
```

In [2]:
```
# 01_analise_exploratoria.ipynb
# Projeto: Paradoxo da Produtividade
# Autor: Lelis Murakami
# Descrição: Análise exploratória inicial dos dados de produtividade
# nas ferramentas No-Code e Low-Code.
print("Notebook carregado com sucesso ✅")
```

```
Notebook carregado com sucesso ✅
```

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# Configurações visuais
plt.style.use('seaborn-v0_8-whitegrid')
pd.set_option('display.max_columns', None)
```

In [4]:
```
import pandas as pd

# Caminho correto do dataset
caminho_arquivo = "../data/processed/produtividade.csv"

try:
    df = pd.read_csv(caminho_arquivo)
    print("✅ Dados carregados com sucesso!")
    display(df.head())
except FileNotFoundError:
    print("⚠️ Arquivo não encontrado. Verifique o caminho:", caminho_arquivo)
```

```
✅ Dados carregados com sucesso!
```

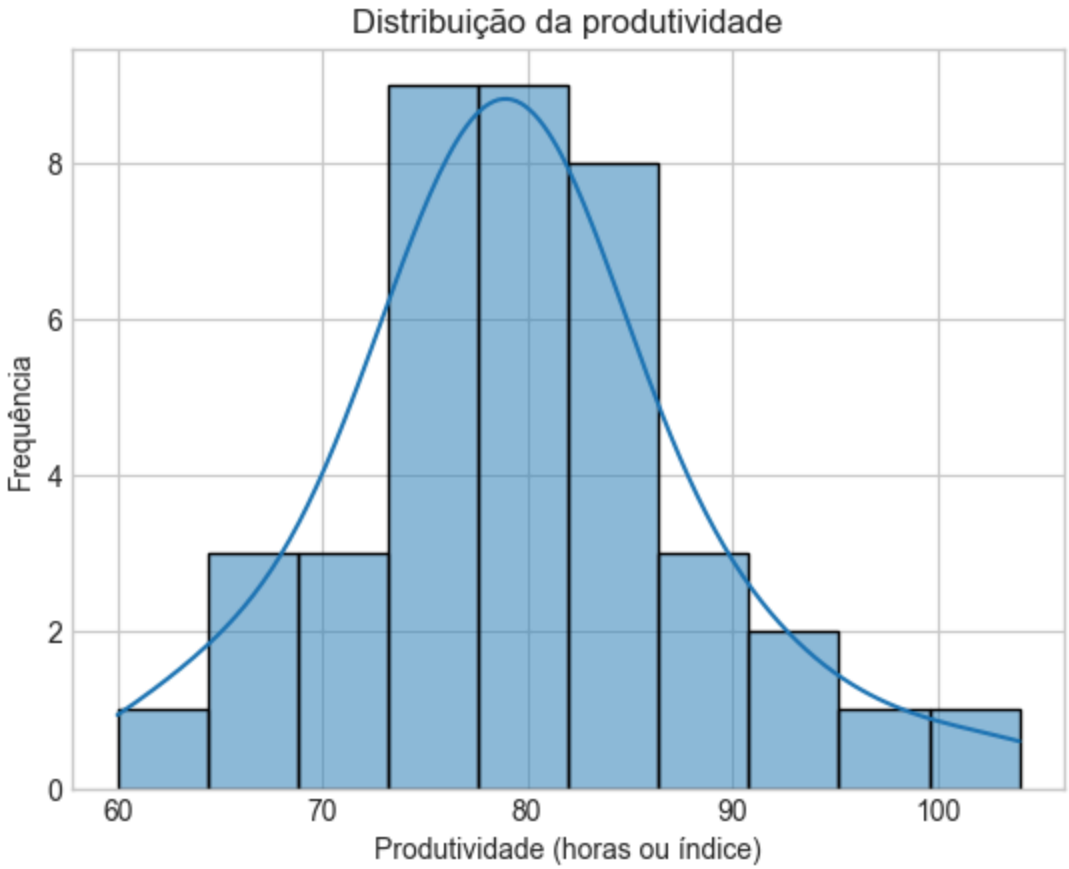
	ferramenta	complexidade_tarefa	tempo_execucao_min	produtividade
0	AppGyver	3	142	77
1	Airtable	2	125	83
2	Python	4	116	94
3	AppGyver	4	110	74
4	AppGyver	3	75	71

In [5]:
```
print("Dimensões do dataset:", df.shape)
print("\nTipos de dados:")
print(df.dtypes)

print("\nValores ausentes:")
print(df.isna().sum())
```

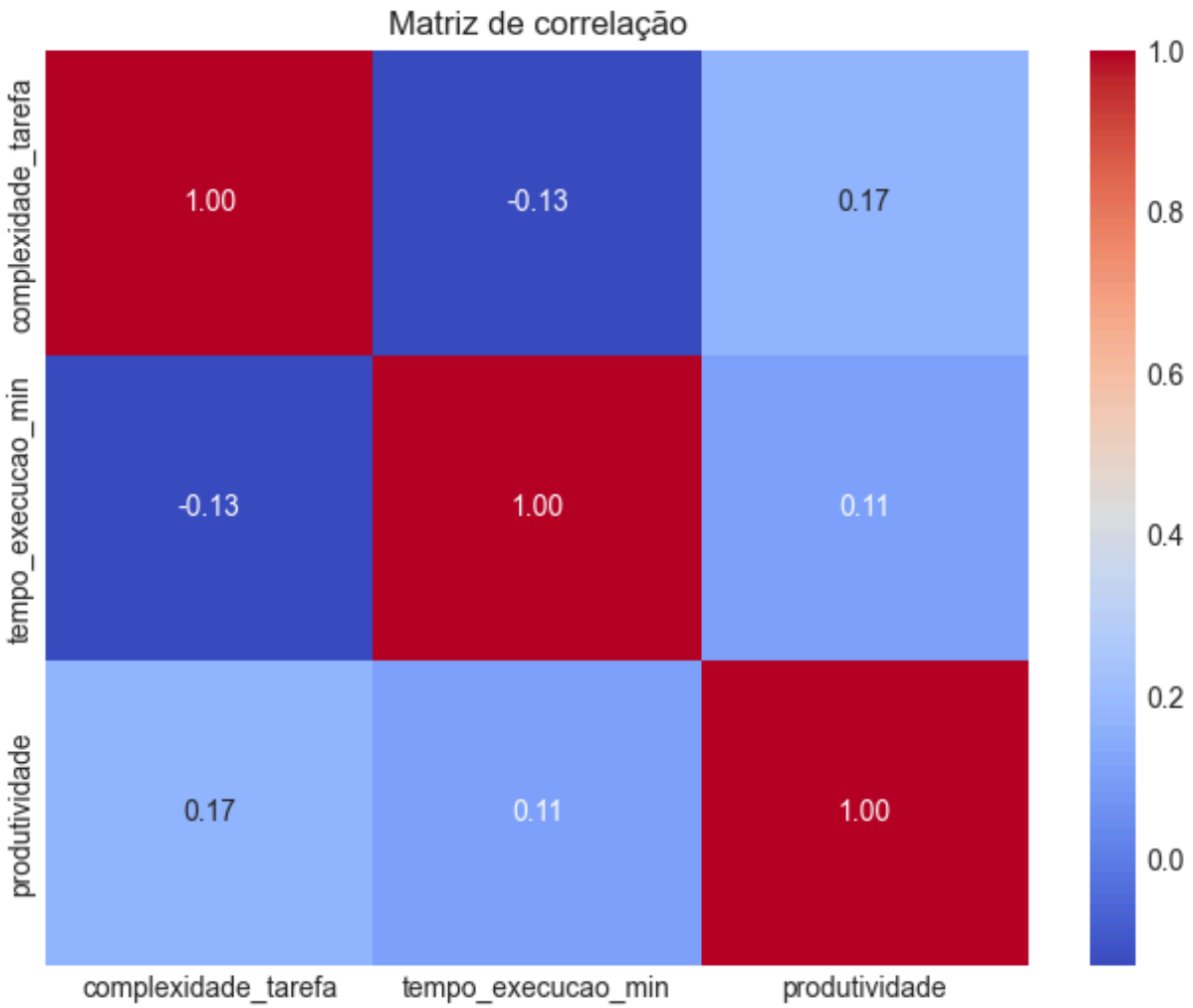
```
Dimensões do dataset: (40, 4)

Tipos de dados:
ferramenta             object
complexidade_tarefa     int64
tempo_execucao_min      int64
produtividade           int64
dtype: object

Valores ausentes:
ferramenta             0
complexidade_tarefa    0
tempo_execucao_min     0
produtividade          0
dtype: int64
```

In [6]:
```
display(df.describe())

# Distribuição de produtividade
sns.histplot(df['produtividade'], kde=True)
plt.title("Distribuição da produtividade")
plt.xlabel("Produtividade (horas ou índice)")
plt.ylabel("Frequência")
plt.show()
```

	complexidade_tarefa	tempo_execucao_min	produtividade
count	40.000000	40.000000	40.000000
mean	3.075000	118.575000	79.525000
std	1.288758	28.901147	8.667616
min	1.000000	41.000000	60.000000
25%	2.000000	103.750000	74.750000
50%	3.000000	120.000000	79.000000
75%	4.000000	142.500000	83.000000
max	5.000000	166.000000	104.000000

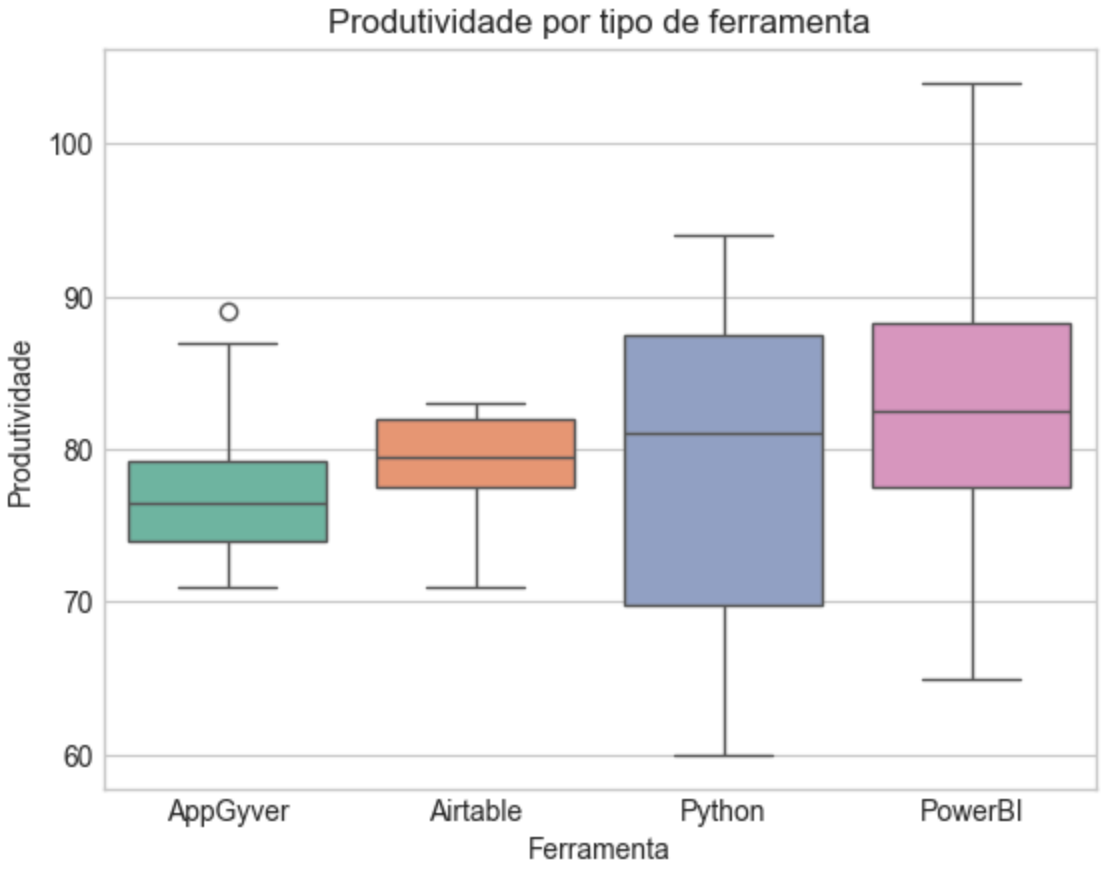


In [7]:
```
corr = df.corr(numeric_only=True)
plt.figure(figsize=(8, 6))
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Matriz de correlação")
plt.show()
```



In [8]:
```python
# Exemplo: comparar produtividade por tipo de ferramenta
if 'ferramenta' in df.columns:
    sns.boxplot(x='ferramenta', y='produtividade', data=df, palette='Set2')
    plt.title("Produtividade por tipo de ferramenta")
    plt.xlabel("Ferramenta")
    plt.ylabel("Produtividade")
    plt.show()
else:
    print("⚠️ A coluna 'ferramenta' não existe no dataset.")
```

```
C:\Users\Lelis\AppData\Local\Temp\ipykernel_17596\1635297684.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='ferramenta', y='produtividade', data=df, palette='Set2')
```



In [9]:
```
# Exportar resumo estatístico para CSV
resumo = df.describe()
resumo.to_csv("dados/processado/resumo_estatistico.csv", index=False)
print("📁 Resumo salvo em dados/processado/resumo_estatistico.csv")
```

```
📁 Resumo salvo em dados/processado/resumo_estatistico.csv
```

In [10]:
```
print("""
✅ Análise exploratória concluída!
- Dados verificados e estatísticas básicas geradas
- Correlações e distribuições avaliadas
- Resultados exportados para a pasta /dados/processado/
""")
```

```
✅ Análise exploratória concluída!
- Dados verificados e estatísticas básicas geradas
- Correlações e distribuições avaliadas
- Resultados exportados para a pasta /dados/processado/
```

In [11]:
```
import os

# Caminho para salvar as figuras
pasta_figuras = "../reports/figures"
os.makedirs(pasta_figuras, exist_ok=True)

# Exemplo: salvar gráfico de correlação
plt.figure(figsize=(6, 4))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')
```

```
plt.title("Correlação entre variáveis")
plt.tight_layout()
plt.savefig(os.path.join(pasta_figuras, "correlacao_variaveis.png"))
plt.close()

# Exemplo: salvar gráfico de produtividade por ferramenta
if 'ferramenta' in df.columns:
    plt.figure(figsize=(6, 4))
    sns.boxplot(x='ferramenta', y='produtividade', data=df, palette='Set2')
    plt.title("Produtividade por tipo de ferramenta")
    plt.tight_layout()
    plt.savefig(os.path.join(pasta_figuras, "produtividade_por_ferramenta.png"))
    plt.close()

print(f"📊 Gráficos salvos em: {pasta_figuras}")
```

```
📊 Gráficos salvos em: ../reports/figures
C:\Users\Lelis\AppData\Local\Temp\ipykernel_17596\2887691876.py:18: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='ferramenta', y='produtividade', data=df, palette='Set2')
```

In [ ]: