Simulador de Métricas para Escolas Online: Decisões Baseadas em Dados com Python

Contextualizando - métricas KPIs

A decisão de crescer e se destacar num mundo cada vez mais competitivo está cada vez mais presente no cotidiano das pessoas e muito mais necessário na compreensão dos dados gerados no ambiente dos negócios.

Tratar os dados de maneira profissional e eficiente faz parte da boa gestão empresarial e não podem ficar em segundo plano pelos administradores.

As métricas (KPIs - Key Performance Indicators) vem de encontro a essa necessidade. Elas são a chave para entender como as coisas estão andando de acordo com o planejado.

Este artigo tem como objetivo apresentar o conceito das KPIs no contexto prático de uma escola online que oferece diferentes produtos e recursos educacionais, fornecendo uma ferramenta que permita simular diferentes cenários de negócios, calculando métricas de desempenho e gerando relatórios detalhados em formato PDF através de um simulador construído em Python.

Mas afinal, o que são e o que medem as KPIs?

Nada mais são que indicadores que ajudam você a medir o sucesso de aspectos importantes do seu negócio. Por exemplo:

- Qual é o lucro da sua escola?
- Quantos alunos você perdeu no último mês (churn rate)?
- O quanto um aluno gera de receita ao longo do tempo (LTV)?

- E quanto custa adquirir novos alunos (CAC)?

A resposta pode vir das métricas. E foi justamente analisando essa necessidade que este projeto nasceu.

A Ideia: Unir Tecnologia e Gestão Educacional

Criar um simulador simples e eficaz em Python que ajudasse administradores de escolas online (foco principal) a entenderem melhor a performance de sua empresa, oferecendo uma ferramenta que:

1. Permita simular diferentes cenários de negócio com base nos dados fornecidos.
2. Calcule automaticamente métricas importantes (KPIs) para a gestão.
3. Gere relatórios visuais em PDF para facilitar a análise.
4. Simplifique a tomada de decisão, transformando dados em informações úteis.

Para desenvolver o projeto, estudei como as métricas funcionam na prática e direcionei esse conhecimento para um contexto educacional. O resultado? Um simulador que combina cálculos automatizados com relatórios visuais completos.

Conceitos Técnicos das KPIs utilizadas

1. Lucro: Receita total menos custos totais.

2. Margem de Lucro: Percentual do lucro em relação à receita total.

3. Taxa de Conversão: Percentual de alunos grátis que se tornam premium.

4. Churn Rate: Percentual de alunos que cancelam suas assinaturas.

5. LTV (Lifetime Value): Valor médio gerado por aluno durante o tempo de assinatura.

6. CAC (Custo de Aquisição por Cliente): Custo médio para adquirir um novo aluno.

7. NPS (Net Promoter Score): Métrica de satisfação dos alunos, dividida em promotores, neutros e detratores.

8. Crescimento Mensal de Alunos: Taxa de crescimento do número de alunos de um mês para o outro.

Montagem Simples e Bem Dividida da Estrutura

O simulador foi dividido em quatro arquivos Python, cada um com uma responsabilidade clara. Isso facilita a manutenção e torna o código mais organizado:

1. simulacao.py:

Função principal: *calcular_kpis*.

Fornece toda a lógica para calcular os KPIs, como lucro, margem de lucro, churn rate, LTV, CAC, entre outros. Ele recebe os dados de entrada do usuário e retorna os resultados como um dicionário organizado.

2. interface_grafica.py:

Através da biblioteca *tkinter*, cria a interface gráfica onde o usuário deverá inserir os dados.

Coleta os valores de entrada e os passa para a função *calcular_kpis*.

3. gerar_relatorio.py:

Função principal: *gerar_relatorio_pdf*.

Este arquivo pega os resultados calculados e gera um relatório PDF contendo as métricas e gráficos. Ele usa as bibliotecas *fpdf* para PDFs e *matplotlib* para criar gráficos visuais.

4. main.py:

Função principal: Iniciar o programa.

Este é o ponto de entrada do script. Ele chama a interface gráfica e conecta todos os outros arquivos.

Relação Entre os Arquivos

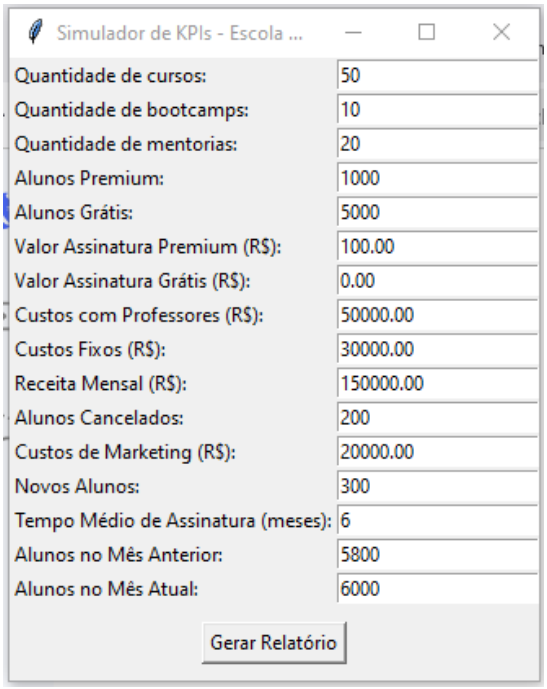
- O arquivo **main.py** inicia a execução do programa, chamando a função *coletar_dados* do arquivo **interface_grafica.py**.

- Por sua vez, a interface gráfica coleta os dados do usuário e os passa para a função *calcular_kpis* do arquivo **simulacao.py**.

- Após o cálculo das KPIs, os resultados são passados para a função *gerar_relatorio_pdf* do arquivo **gerar_relatorio.py**, que gera o relatório final em PDF.

Funcionamento do Script

1. Entrada do Usuário:



- O usuário insere os dados na interface gráfica, como:

- Quantidade de cursos, bootcamps e mentorias.

- Número de alunos premium e grátis.

- Valores das assinaturas premium e grátis.

- Custos com professores, custos fixos e custos de marketing.

- Receita mensal, número de alunos cancelados, novos alunos e tempo médio de assinatura.

2. Cálculo dos KPIs:

Com os dados fornecidos, o script calcula automaticamente métricas como:

- Lucro e margem de lucro.
- Taxa de conversão e churn rate (taxa de cancelamento).
- LTV (Lifetime Value) e CAC (Custo de Aquisição de Cliente).
- ROI (Retorno sobre Investimento), entre outros.

3. Processamento:

- Os dados são passados para a função `calcular_kpis`, que calcula todas as métricas.

- Os resultados são então enviados para a função `gerar_relatorio_pdf`.

4. Geração do Relatório:

- A função **gerar_relatorio_pdf** cria gráficos utilizando *matplotlib* e os insere no PDF.

- O relatório em PDF é salvo com o nome *relatorio_kpis.pdf*.

5. Tomada de decisão:

O administrador usa o relatório para entender a performance da escola e tomar decisões mais embasadas.

Resultados

O relatório gerado inclui:

1. KPIs Calculados:

- Lucro, margem de lucro, taxa de conversão, churn rate, LTV, CAC, NPS e crescimento mensal de alunos.

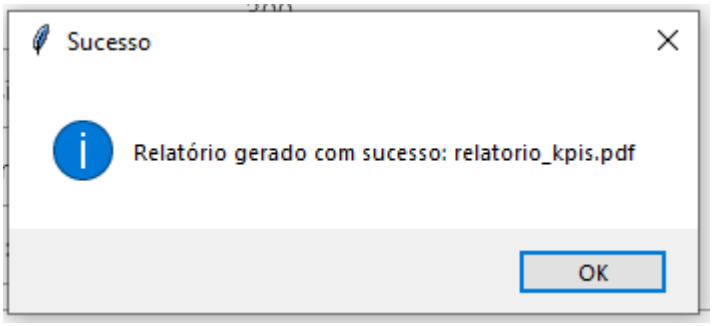
2. Gráficos:

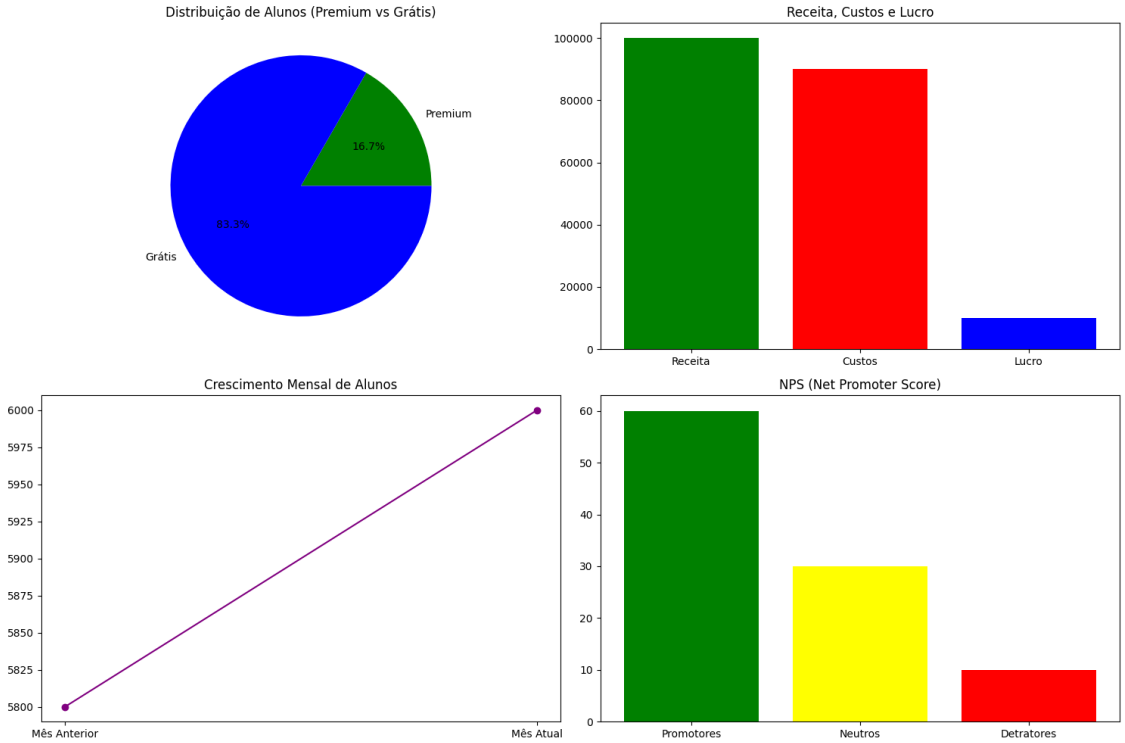
- Distribuição de alunos (Premium vs Grátis).

- Comparação de receita, custos e lucro.

- Crescimento mensal de alunos.

- Distribuição do NPS (Promotores, Neutros, Detratores).





Conclusão

Este projeto provou que a tecnologia pode ser uma grande aliada na gestão de negócios, mesmo para iniciantes.

Usando Python, conseguimos desenvolver um simulador simples, mas poderoso, que ajuda escolas online a transformarem dados em insights valiosos. Através de uma estrutura modular enxuta, o desenvolvimento se tornou prático com possibilidades de expandir no futuro. A ferramenta possui uma interface gráfica amigável disponibilizando ainda um relatório visual em PDF

economizando tempo e oferecendo uma visão clara da performance.

ANEXO – Os Códigos

a) Simulação.py

```
def calcular_kpis(cursos, bootcamps, mentorias, alunos_premium,
alunos_gratis, valor_premium, valor_gratis, custos_professores,
custos_fixos, receita_mensal, alunos_cancelados, custos_marketing,
novos_alunos, tempo_medio_assinatura, alunos_mes_anterior,
alunos_mes_atual):
    # Cálculo de KPIs
    total_alunos = alunos_premium + alunos_gratis
    receita_total = (alunos_premium * valor_premium) + (alunos_gratis *
valor_gratis)
    custos_totais = custos_professores + custos_fixos + custos_marketing
    lucro = receita_total - custos_totais
    margem_lucro = (lucro / receita_total) * 100 if receita_total != 0
else 0

    # Novas KPIs
    taxa_conversao = (alunos_premium / alunos_gratis) * 100 if
alunos_gratis != 0 else 0
    custo_por_aluno = custos_totais / total_alunos if total_alunos != 0
else 0
    receita_por_aluno = receita_total / total_alunos if total_alunos != 0
else 0
    roi = (lucro / custos_totais) * 100 if custos_totais != 0 else 0
    churn_rate = (alunos_cancelados / total_alunos) * 100 if total_alunos
!= 0 else 0
    ltv = (receita_total / total_alunos) * tempo_medio_assinatura if
total_alunos != 0 else 0
    cac = custos_marketing / novos_alunos if novos_alunos != 0 else 0
    crescimento_mensal = ((alunos_mes_atual - alunos_mes_anterior) /
alunos_mes_anterior) * 100 if alunos_mes_anterior != 0 else 0

    # Simulação de NPS
    nps_promotores = 60  # 60% promotores
    nps_neutros = 30     # 30% neutros
    nps_detratores = 10  # 10% detratores
```

```
    kpis = {
        "total_alunos": total_alunos,
        "receita_total": receita_total,
        "custos_totais": custos_totais,
        "lucro": lucro,
        "margem_lucro": margem_lucro,
        "taxa_conversao": taxa_conversao,
        "custo_por_aluno": custo_por_aluno,
        "receita_por_aluno": receita_por_aluno,
        "roi": roi,
        "churn_rate": churn_rate,
        "ltv": ltv,
        "cac": cac,
        "crescimento_mensal": crescimento_mensal,
        "nps_promotores": nps_promotores,
        "nps_neutros": nps_neutros,
        "nps_detratores": nps_detratores,
        "cursos": cursos,
        "bootcamps": bootcamps,
        "mentorias": mentorias,
        "alunos_premium": alunos_premium,
        "alunos_gratis": alunos_gratis,
        "valor_premium": valor_premium,
        "valor_gratis": valor_gratis,
        "custos_professores": custos_professores,
        "custos_fixos": custos_fixos,
        "receita_mensal": receita_mensal,
        "alunos_cancelados": alunos_cancelados,
        "custos_marketing": custos_marketing,
        "novos_alunos": novos_alunos,
        "tempo_medio_assinatura": tempo_medio_assinatura,
        "alunos_mes_anterior": alunos_mes_anterior,
        "alunos_mes_atual": alunos_mes_atual
    }

    return kpis
```

b) gerar_relatorio.py

```
from fpdf import FPDF
import matplotlib.pyplot as plt
import locale

# Configurar o locale para o padrão brasileiro
locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')
```

```python
def gerar_relatorio_pdf(kpis):
    # Criar gráficos
    plt.figure(figsize=(15, 10))

    # Gráfico de Pizza: Distribuição de Alunos (Premium vs Grátis)
    plt.subplot(2, 2, 1)
    labels = ["Premium", "Grátis"]
    sizes = [kpis["alunos_premium"], kpis["alunos_gratis"]]
    plt.pie(sizes, labels=labels, autopct="%1.1f%%", colors=["green",
"blue"])
    plt.title("Distribuição de Alunos (Premium vs Grátis)")

    # Gráfico de Barras: Receita, Custos e Lucro
    plt.subplot(2, 2, 2)
    categorias = ["Receita", "Custos", "Lucro"]
    valores = [kpis["receita_total"], kpis["custos_totais"],
kpis["lucro"]]
    plt.bar(categorias, valores, color=["green", "red", "blue"])
    plt.title("Receita, Custos e Lucro")

    # Gráfico de Linhas: Crescimento Mensal de Alunos
    plt.subplot(2, 2, 3)
    meses = ["Mês Anterior", "Mês Atual"]
    alunos = [kpis["alunos_mes_anterior"], kpis["alunos_mes_atual"]]
    plt.plot(meses, alunos, marker="o", color="purple")
    plt.title("Crescimento Mensal de Alunos")

    # Gráfico de Barras: NPS (Promotores, Neutros, Detratores)
    plt.subplot(2, 2, 4)
    nps_categorias = ["Promotores", "Neutros", "Detratores"]
    nps_valores = [kpis["nps_promotores"], kpis["nps_neutros"],
kpis["nps_detratores"]]
    plt.bar(nps_categorias, nps_valores, color=["green", "yellow",
"red"])
    plt.title("NPS (Net Promoter Score)")

    plt.tight_layout()
    plt.savefig("graficos.png")

    # Gerar PDF
    pdf = FPDF()
    pdf.add_page()
    pdf.set_font("Arial", size=12)

    # Título do Relatório
    pdf.cell(200, 10, txt="Relatório de KPIs - Escola Online", ln=True,
align="C")
    pdf.ln(10)
```

```
    # Adicionar KPIs ao PDF (formatados em R$)
    pdf.cell(200, 10, txt=f"Total de Alunos: {kpis['total_alunos']}",
ln=True)
    pdf.cell(200, 10, txt=f"Receita Total:
{locale.currency(kpis['receita_total'], grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"Custos Totais:
{locale.currency(kpis['custos_totais'], grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"Lucro: {locale.currency(kpis['lucro'],
grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"Margem de Lucro:
{kpis['margem_lucro']:.2f}%", ln=True)
    pdf.cell(200, 10, txt=f"Taxa de Conversão:
{kpis['taxa_conversao']:.2f}%", ln=True)
    pdf.cell(200, 10, txt=f"Custo por Aluno:
{locale.currency(kpis['custo_por_aluno'], grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"Receita por Aluno:
{locale.currency(kpis['receita_por_aluno'], grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"ROI: {kpis['roi']:.2f}%", ln=True)
    pdf.cell(200, 10, txt=f"Churn Rate: {kpis['churn_rate']:.2f}%",
ln=True)
    pdf.cell(200, 10, txt=f"LTV: {locale.currency(kpis['ltv'],
grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"CAC: {locale.currency(kpis['cac'],
grouping=True)}", ln=True)
    pdf.cell(200, 10, txt=f"Crescimento Mensal:
{kpis['crescimento_mensal']:.2f}%", ln=True)
    pdf.cell(200, 10, txt=f"NPS: Promotores {kpis['nps_promotores']}%,
Neutros {kpis['nps_neutros']}%, Detratores {kpis['nps_detratores']}%",
ln=True)
    pdf.ln(10)

    # Adicionar gráficos ao PDF
    pdf.image("graficos.png", x=10, y=pdf.get_y(), w=180)

    # Adicionar Anexo com Explicações Dinâmicas
    pdf.add_page()
    pdf.set_font("Arial", size=12)
    pdf.cell(200, 10, txt="ANEXO: Explicação das Fórmulas e Análise dos
Resultados", ln=True, align="C")
    pdf.ln(10)

    # Explicação das Fórmulas e Análise dos Resultados
    pdf.set_font("Arial", size=12, style="B")
    pdf.cell(200, 10, txt="1. Explicação das Fórmulas e Resultados",
ln=True)
    pdf.set_font("Arial", size=12)
    pdf.multi_cell(0, 10, txt=f"""
```

```
    - Receita Total: A receita total foi de
{locale.currency(kpis['receita_total'], grouping=True)}, sendo
{locale.currency(kpis['alunos_premium'] * kpis['valor_premium'],
grouping=True)} proveniente dos alunos premium e
{locale.currency(kpis['alunos_gratis'] * kpis['valor_gratis'],
grouping=True)} dos alunos grátis. Isso indica que a escola depende
principalmente dos alunos premium para gerar receita.

    - Custos Totais: Os custos totais foram de
{locale.currency(kpis['custos_totais'], grouping=True)}, incluindo
{locale.currency(kpis['custos_professores'], grouping=True)} com
professores, {locale.currency(kpis['custos_fixos'], grouping=True)} com
custos fixos e {locale.currency(kpis['custos_marketing'], grouping=True)}
com marketing. A escola deve avaliar a eficiência desses custos.

    - Lucro: O lucro foi de {locale.currency(kpis['lucro'],
grouping=True)}, resultante da diferença entre a receita total e os
custos totais. {"Um lucro positivo indica que a escola está operando com
eficiência financeira." if kpis['lucro'] >= 0 else "Um lucro negativo
indica que a escola está operando com prejuízo, o que é preocupante."}

    - Margem de Lucro: A margem de lucro foi de
{kpis['margem_lucro']:.2f}%. {"Isso é considerado saudável, mas há espaço
para melhorias, como a redução de custos ou o aumento da receita." if
kpis['margem_lucro'] >= 0 else "Uma margem de lucro negativa indica que a
escola está gastando mais do que arrecada, o que é insustentável a longo
prazo."}

    - Taxa de Conversão: A taxa de conversão foi de
{kpis['taxa_conversao']:.2f}%, indicando que, para cada 5 alunos grátis,
1 se torna premium. A escola pode melhorar essa taxa com estratégias de
marketing mais eficazes.

    - Custo por Aluno: O custo por aluno foi de
{locale.currency(kpis['custo_por_aluno'], grouping=True)}, o que é
{"razoável, mas pode ser otimizado." if kpis['custo_por_aluno'] <= 50
else "alto, indicando que a escola precisa reduzir custos para se tornar
mais eficiente."}

    - Receita por Aluno: A receita por aluno foi de
{locale.currency(kpis['receita_por_aluno'], grouping=True)}, um valor
{"baixo devido ao grande número de alunos grátis. A escola deve focar em
aumentar a base de alunos premium." if kpis['receita_por_aluno'] <= 50
else "adequado, mas ainda pode ser melhorado com estratégias de upsell."}

    - ROI: O ROI foi de {kpis['roi']:.2f}%, indicando que a escola está
{"gerando um retorno positivo sobre seus investimentos." if kpis['roi']
>= 0 else "operando com um retorno negativo, o que é preocupante."}
```

```
    - Churn Rate: O Churn Rate foi de {kpis['churn_rate']:.2f}%,
{"considerado baixo. Isso sugere que a maioria dos alunos está satisfeita
com os serviços da escola." if kpis['churn_rate'] <= 5 else "considerado
alto. Isso sugere que muitos alunos estão cancelando suas assinaturas, o
que pode ser um sinal de insatisfação."}

    - LTV: O LTV foi de {locale.currency(kpis['ltv'], grouping=True)},
indicando o valor médio que um aluno gera durante todo o tempo de
assinatura. A escola pode aumentar esse valor com estratégias de
retenção.

    - CAC: O CAC foi de {locale.currency(kpis['cac'], grouping=True)}, o
que é {"aceitável, mas pode ser reduzido com estratégias de marketing
mais eficientes." if kpis['cac'] <= 100 else "alto, indicando que a
escola está gastando muito para adquirir novos alunos."}

    - Crescimento Mensal: O crescimento mensal foi de
{kpis['crescimento_mensal']:.2f}%, indicando uma {"expansão constante da
base de alunos." if kpis['crescimento_mensal'] >= 0 else "redução na base
de alunos, o que é preocupante."}

    - NPS: O NPS foi de {kpis['nps_promotores'] -
kpis['nps_detratores']}, com {kpis['nps_promotores']}% de promotores,
{kpis['nps_neutros']}% de neutros e {kpis['nps_detratores']}% de
detratores. Isso indica que a maioria dos alunos está satisfeita com a
escola.
    """)

    # Parecer Geral do Analista
    pdf.set_font("Arial", size=12, style="B")
    pdf.cell(200, 10, txt="2. Parecer Geral do Analista", ln=True)
    pdf.set_font("Arial", size=12)

    if kpis['lucro'] < 0:
        parecer = f"""
        A escola online está operando com prejuízo de
{locale.currency(kpis['lucro'], grouping=True)} e uma margem de lucro
negativa de {kpis['margem_lucro']:.2f}%. Isso indica que os custos estão
muito altos em relação à receita gerada. Além disso, o Churn Rate de
{kpis['churn_rate']:.2f}% sugere que muitos alunos estão cancelando suas
assinaturas, o que pode ser um sinal de insatisfação.

        Recomenda-se:
        - Reduzir drasticamente os custos, especialmente com professores
e marketing.
        - Revisar a estratégia de preços para aumentar a receita.
        - Implementar programas de retenção para reduzir o Churn Rate.
        - Avaliar a eficácia das campanhas de marketing, já que o CAC
está alto em relação ao LTV.
```

```
        A situação atual é crítica, e ações imediatas são necessárias
para evitar prejuízos maiores.
        """
    else:
        parecer = f"""
        A escola online apresenta um desempenho financeiro
{"satisfatório" if kpis['lucro'] > 0 else "neutro"}, com um lucro de
{locale.currency(kpis['lucro'], grouping=True)} e uma margem de lucro de
{kpis['margem_lucro']:.2f}%. No entanto, há oportunidades de melhoria,
como:
        - Aumentar a taxa de conversão de alunos grátis para premium.
        - Reduzir os custos de marketing para melhorar o ROI.
        - Implementar estratégias de retenção para aumentar o LTV.
        - Acelerar o crescimento mensal de alunos com campanhas mais
eficientes.

        No geral, a escola está no caminho certo, mas pode alcançar
resultados ainda melhores com ajustes estratégicos.
        """

    pdf.multi_cell(0, 10, txt=parecer)

    pdf.output("relatorio_kpis.pdf")
    print("Relatório gerado com sucesso: relatorio_kpis.pdf")
```

c) interface_grafica.py

```
import tkinter as tk
from tkinter import messagebox
from src.simulacao import calcular_kpis
from gerar_relatorio import gerar_relatorio_pdf
import locale

# Configurar o locale para o padrão brasileiro
locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')

def coletar_dados():
    def gerar_relatorio():
        try:
            # Coletar dados dos campos
            cursos = int(entry_cursos.get())
            bootcamps = int(entry_bootcamps.get())
            mentorias = int(entry_mentorias.get())
            alunos_premium = int(entry_alunos_premium.get())
```

```
            alunos_gratis = int(entry_alunos_gratis.get())
            valor_premium = float(entry_valor_premium.get())
            valor_gratis = float(entry_valor_gratis.get())
            custos_professores = float(entry_custos_professores.get())
            custos_fixos = float(entry_custos_fixos.get())
            receita_mensal = float(entry_receita_mensal.get())
            alunos_cancelados = int(entry_alunos_cancelados.get())
            custos_marketing = float(entry_custos_marketing.get())
            novos_alunos = int(entry_novos_alunos.get())
            tempo_medio_assinatura =
float(entry_tempo_medio_assinatura.get())
            alunos_mes_anterior = int(entry_alunos_mes_anterior.get())
            alunos_mes_atual = int(entry_alunos_mes_atual.get())

            # Calcular KPIs
            kpis = calcular_kpis(
                cursos, bootcamps, mentorias, alunos_premium,
alunos_gratis,
                valor_premium, valor_gratis, custos_professores,
custos_fixos,
                receita_mensal, alunos_cancelados, custos_marketing,
novos_alunos,
                tempo_medio_assinatura, alunos_mes_anterior,
alunos_mes_atual
            )

            # Gerar relatório
            gerar_relatorio_pdf(kpis)
            messagebox.showinfo("Sucesso", "Relatório gerado com sucesso:
relatorio_kpis.pdf")
        except ValueError:
            messagebox.showerror("Erro", "Por favor, insira valores
válidos.")

    # Configuração da interface gráfica
    root = tk.Tk()
    root.title("Simulador de KPIs - Escola Online")

    # Campos de entrada
    campos = [
        ("Quantidade de cursos:", 0),
        ("Quantidade de bootcamps:", 1),
        ("Quantidade de mentorias:", 2),
        ("Alunos Premium:", 3),
        ("Alunos Grátis:", 4),
        ("Valor Assinatura Premium (R$):", 5),
        ("Valor Assinatura Grátis (R$):", 6),
        ("Custos com Professores (R$):", 7),
        ("Custos Fixos (R$):", 8),
```

```
        ("Receita Mensal (R$):", 9),
        ("Alunos Cancelados:", 10),
        ("Custos de Marketing (R$):", 11),
        ("Novos Alunos:", 12),
        ("Tempo Médio de Assinatura (meses):", 13),
        ("Alunos no Mês Anterior:", 14),
        ("Alunos no Mês Atual:", 15)
    ]

    entries = []
    for label_text, row in campos:
        tk.Label(root, text=label_text).grid(row=row, column=0,
sticky="w")
        entry = tk.Entry(root)
        entry.grid(row=row, column=1)
        entries.append(entry)

    # Atribuir cada campo a uma variável
    (
        entry_cursos, entry_bootcamps, entry_mentorias,
entry_alunos_premium,
        entry_alunos_gratis, entry_valor_premium, entry_valor_gratis,
        entry_custos_professores, entry_custos_fixos,
entry_receita_mensal,
        entry_alunos_cancelados, entry_custos_marketing,
entry_novos_alunos,
        entry_tempo_medio_assinatura, entry_alunos_mes_anterior,
entry_alunos_mes_atual
    ) = entries

    # Botão para gerar relatório
    tk.Button(root, text="Gerar Relatório",
command=gerar_relatorio).grid(row=16, column=0, columnspan=2, pady=10)

    # Encerrar o programa corretamente ao fechar a janela
    root.protocol("WM_DELETE_WINDOW", root.quit)

    root.mainloop()

if __name__ == "__main__":
    coletar_dados()
```

d) main.py

```python
from src.interface_grafica import coletar_dados

def main():
    print("Iniciando o Simulador de KPIs para Escola Online...")
    coletar_dados()

if __name__ == "__main__":
    main()
```

Siga-me no LinkedIn: www.linkedin.com/comm/mynetwork/discovery-see-all?usecase=PEOPLE_FOLLOWS&followMember=izairton-oliveira-de-vasconcelos-a1916351

Minha Newsletter, o link para assinar:https://www.linkedin.com/build-relation/newsletter-follow?entityUrn=7287106727202742273

Link do artigo no Linkedin: https://www.linkedin.com/pulse/simulador-de-m%25C3%25A9tricas-para-escolas-online-decis%25C3%25B5es-em-vasconcelos-oeayf

https://github.com/IOVASCON/simulador_kpi_escola.git