



Instituto Infnet

GRADUAÇÃO EM BANCO DE DADOS

Projeto de Bloco: Ciência de Dados Aplicada [24E3_5]

WANDERSON RAFAEL MENDONÇA BATISTA

TESTE DE PERFORMANCE – TP1

PROF. DIEGO DA SILVA RODRIGUES

RIO DE JANEIRO, 2024

1. Identificar o Problema de Negócio:

Defina o problema de negócio que sua aplicação irá resolver.

Estabeleça as metas e os indicadores que determinarão o sucesso do projeto.

Escreva a qual ODS seu projeto atende e justifique sua resposta.

Estabeleça qual o público-alvo de sua aplicação.

O problema de negócio que este projeto visa resolver é a falta de acessibilidade e clareza na visualização e análise de dados de saúde pública provenientes do FTP do SUS, dificultando a elaboração de políticas públicas eficientes e o acesso do cidadão a informações sobre onde buscar atendimento. A meta do projeto é facilitar a tomada de decisões por gestores públicos e melhorar a experiência dos usuários na busca por serviços de saúde, com indicadores de sucesso como o número de acessos à aplicação, o tempo de resposta na extração de dados e a adoção das informações em planejamentos de saúde. O projeto atende ao ODS 3 (Saúde e Bem-Estar), pois busca melhorar a saúde pública ao fornecer dados claros e acessíveis que apoiam políticas e ações voltadas para o acesso equitativo a serviços de saúde de qualidade. O público-alvo da aplicação são gestores públicos de saúde, responsáveis pela criação de políticas, e cidadãos que necessitam localizar unidades de saúde e serviços oferecidos.

2. Organização do Projeto:

Estruture seu projeto utilizando os métodos CRISP-DM (Cross-Industry Standard Process for Data Mining) e TDSP (Team Data Science Process).

Compreenda as diferentes fases do ciclo de vida do TDSP e como elas se aplicam ao seu projeto.

	Business Understanding	Data Understanding	Modeling	Deployment	Acceptance
Project Leader	<div>Definir Problema de Negócio</div> <div>Project Charter</div>				
Data Scientist		<div>Data Exploration Data Summary Report</div> <div>Data Summary Report</div>	<div>Feature Engenieering Data Cleaning Dataviz</div> <div>Feature Engenieering scripts Dashboard</div>		
Project Manage	<div>Definir Metas e KPI</div> <div>Project Charter</div>				<div>Verificação das Metas e KPI</div>
Solution Architect			<div>Design da Solução</div> <div>Solution Architeture Document</div>	<div>Deploy do Dashboard</div>	

3. Organização de Diretórios e Artefatos Iniciais:

Crie uma organização de diretórios que reflita as diferentes fases do ciclo de vida do TDSP.

Desenvolva os artefatos iniciais do projeto, incluindo:

Project Charter: Documento que descreve o escopo, os objetivos e os stakeholders do projeto (Em anexo).

Data Summary Report: Relacione as fontes de dados que serão utilizadas, indicando o tipo de dados e o objetivo de uso. Este será o primeiro esboço do Data Summary Report.



O Data Summary Report foi incluído no visual do projeto, uma vez que o mesmo já apresenta informações sobre os dados analisados



←

→

↻

localhost:8501

ChatGPTEMSERHPE - EMSERHPaneis - EMSERHPortainer | localhdfshue - Bem-vindo à...BI ProduçãoBI HomologaçãoBI launch padSIA-WEBSchoologyTradeciety Trading C...Inf

Navegação

Escolha a página

☐ Introdução

☒ Data Summary Report

☐ Visualização de Dados

3. Características dos Dados

Tamanho do Conjunto de Dados

Número de registros: 1160910

Número de colunas: 12

Variáveis Principais e Tipos

FANTASIA:

Nome da unidade de saúde, utilizado como filtro principal. (Tipo: Texto)

PA_MVMR:

Período de movimentação dos dados no formato AAAAMM, utilizado para busca por ano. (Tipo: Numérico)

PA_PROC_ID:

Identificação dos procedimentos realizados na unidade de saúde. (Tipo: Texto ou Numérico)

PROD_APRESENTADA:

Quantidade de produção apresentada por procedimento. (Tipo: Numérico)

PROD_APROVADA:

Quantidade de produção aprovada por procedimento. (Tipo: Numérico)

←

→

↺

localhost:8501

ChatGPTEMSERHPE - EMSERHPaineis - EMSERHPortainer | localhdfshue - Bem-vindo à...BI ProduçãoBI HomologaçãoBI launch padSIA-WEBSchoolologyTradeciety Trading C...

Navegação

Escolha a página

Introdução

Data Summary Report

Visualização de Dados

PROD_APROVADA: Quantidade de produção aprovada por procedimento. (Tipo: Numérico)

Estatísticas Descritivas

Aqui estão as principais estatísticas dos dados numéricos:

	TOTAL_PA_QTDPRO	TOTAL_PA_QTDAPR	TOTAL_PA_VALPRO	TOTAL_PA_VALAPR
count	1,160,910	1,160,910	1,160,910	1,160,910
mean	279.2924	249.0081	2,005.8674	1,867.2859
std	3,579.4726	2,119.708	28,305.3257	18,435.2483
min	0	0	0	0
25%	6	6	7.36	6.75
50%	25	24	110	104.71
75%	116	111	680.8	653.25
max	972,020	770,301	10,604,500.47	2,736,855

Verificação de Valores Faltantes

	0
PA_CBOCOD	14,210
FANTASIA	582

4. Desenvolvimento de uma Aplicação Demo com Streamlit:

- Crie uma aplicação demo utilizando Streamlit, que incluirá:
 - Um título para o projeto.
 - Uma descrição do problema de negócio e dos objetivos do projeto.
 - Links úteis para as iniciativas e fontes de inspiração do projeto.
 - Uma tabela exibindo amostras dos dados que serão utilizados ao longo do projeto.

Página de introdução com título e links uteis

Navegação

Escolha a página

Introdução

Data Summary Report

Visualização de Dados



Bem-vindo ao Projeto de Visualização de Dados do SUS

Este projeto utiliza dados oriundos do FTP do SUS e tem como objetivo auxiliar os gestores públicos na elaboração de políticas públicas de saúde, bem como ajudar os usuários a identificar onde buscar atendimento. Através dessa plataforma, é possível visualizar e analisar dados de saúde de forma acessível e interativa.

Links Úteis

- [FTP do SUS \(Dados\)](#): Local para busca dos dados.
- [FTP do SUS \(Manual\)](#): Local com informações técnicas dos arquivos utilizados.
- [DATASUS](#): Local para pesquisa dos dados sem necessidade de utilização do FTP.

Para apresentação dos dados foi necessário realizar mudanças no banco de dados e solicitar que o usuário escolha quais dados devem ser visualizados, o Streamlit não suportou acessar e apresentar todos os dados

Navegação

Escolha a página

Introdução

Data Summary Report

Visualização de Dados

Filtros de Dados

Unidade de Saúde

HOSPITAL DA ILHA

Ano

2024

Carregar Dados Filtrados

Dados filtrados carregados com sucesso!

Número de registros mostrados: 1218

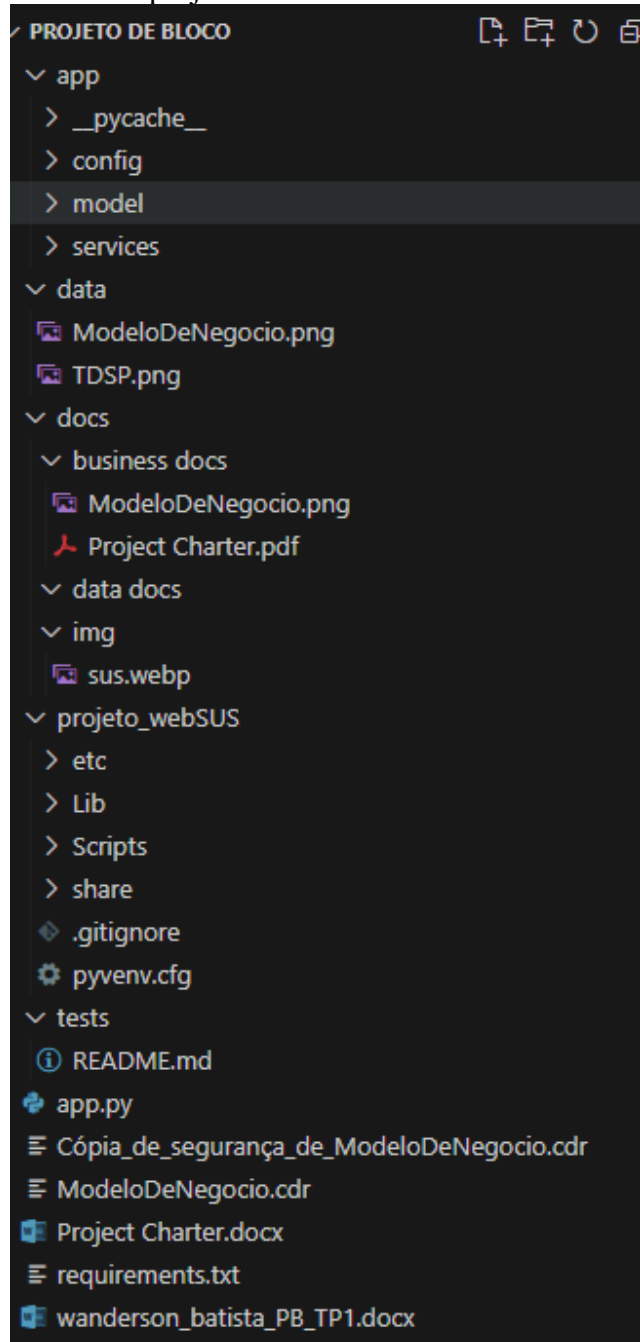
	TOTAL_PA_QTDPRO	TOTAL_PA_QTDAPR	TOTAL_PA_VALPRO	TOTAL_PA_VALAPR	PA_CODUNI
0	353	353	8,556.7200	8,556.7200	2882655
1	553	553	13,404.7200	13,404.7200	2882655
2	516	516	12,507.8400	12,507.8400	2882655
3	553	553	1,863.6100	1,863.6100	2882655
4	516	516	5,160.0000	5,160.0000	2882655
5	194	194	84,669.3600	84,669.3600	2882655
6	265	265	204,474.0000	204,474.0000	2882655
7	10	10	20.1000	20.1000	2882655
8	300	300	555.0000	555.0000	2882655
9	8	8	20.4400	20.4400	2882655

Crie o ambiente de desenvolvimento (pyenv, virtualenv, conda...) e crie um arquivo com os requisitos necessários para rodar sua aplicação. Esse arquivo deve ficar na raiz do seu projeto.

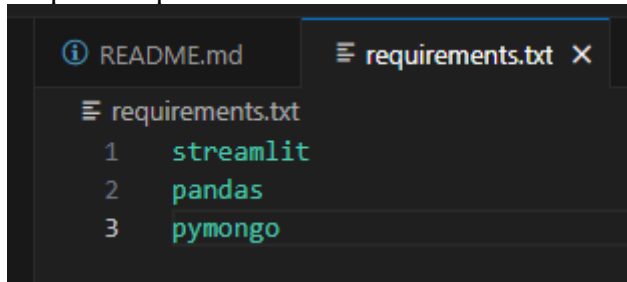
Utilização do ambiente projeto_webSUS

```
O carregamento de perfis pessoais e do sistema levou 1444ms.  
PS C:\Users\wande\OneDrive\INFNET\7. 6º Semestre\Projeto de Bloco> .\projeto_webSUS\Scripts\Activate  
(projeto_webSUS) PS C:\Users\wande\OneDrive\INFNET\7. 6º Semestre\Projeto de Bloco> code .  
(projeto_webSUS) PS C:\Users\wande\OneDrive\INFNET\7. 6º Semestre\Projeto de Bloco> |
```

Árvore do projeto



Arquivo requirements.txt



```
requirements.txt
1 streamlit
2 pandas
3 pymongo
```

Script do código

```
import streamlit as st
from pymongo import MongoClient
import pandas as pd

# Função para conectar ao MongoDB e buscar os dados de análise
@st.cache_data(show_spinner=False)
def fetch_all_data():
    client = MongoClient("mongodb://localhost:27017/")
    db_analise = client["ANALISE"]
    collection_sia_analise = db_analise["SIA_ANALISE"]

    # Obter todos os documentos da collection SIA_ANALISE
    data = list(collection_sia_analise.find({}, {"_id": 0}))

    client.close()

    return data

# Função para converter os dados em um DataFrame do pandas
def convert_to_dataframe(data):
    if data:
        return pd.DataFrame(data)
    else:
        return pd.DataFrame() # Retorna DataFrame vazio se não houver dados

# Página Data Summary Report
def data_summary_report_page():
    # Introdução
    st.title("Data Summary Report")
    st.write("""
    Este relatório resume os dados utilizados no projeto de visualização de dados de saúde
    baseado no FTP do SUS. O projeto visa
    auxiliar gestores públicos na elaboração de políticas de saúde e ajudar os usuários a
    identificar onde buscar atendimento.
    Os dados são oriundos do FTP do SUS, sendo visualizados e analisados por meio de
    uma aplicação desenvolvida em Streamlit.
    """)

    # Descrição dos Dados
```

```

st.header("2. Descrição dos Dados")
st.write("""
- **Fonte dos Dados**: FTP do SUS –
[ftp://ftp.datasus.gov.br/dissemin/publicos/SIASUS/200801_/Dados/](ftp://ftp.datasus.g
ov.br/dissemin/publicos/SIASUS/200801_/Dados/)
- **Coleta de Dados**: A coleta dos dados foi feita a partir dos arquivos
disponibilizados no FTP do SUS e incluídos em um banco de dados MongoDB,
contendo informações de procedimentos e serviços de saúde prestados.
- **Objetivo dos Dados**: O conjunto de dados contém informações sobre a produção
de serviços de saúde, tais como:
- Unidade de Saúde (campo `FANTASIA`)
- Período de Movimentação (campo `PA_MVMR`, no formato `AAAAMM`)
- Procedimentos realizados (campo `PA_PROC_ID`)
- Quantidade de produção apresentada e aprovada (campos
`PROD_APRESENTADA` e `PROD_APROVADA`)
""")

# Carregar os dados do MongoDB
data = fetch_all_data()
df = convert_to_dataframe(data)

if not df.empty:
    # Características dos Dados
    st.header("3. Características dos Dados")

    # Tamanho do conjunto de dados
    st.subheader("Tamanho do Conjunto de Dados")
    st.write(f"Número de registros: {len(df)}")
    st.write(f"Número de colunas: {len(df.columns)}")

    # Variáveis principais e tipos
    st.subheader("Variáveis Principais e Tipos")
    st.write("""
- **FANTASIA**: Nome da unidade de saúde, utilizado como filtro principal. (Tipo:
Texto)
- **PA_MVMR**: Período de movimentação dos dados no formato `AAAAMM`,
utilizado para busca por ano. (Tipo: Numérico)
- **PA_PROC_ID**: Identificação dos procedimentos realizados na unidade de
saúde. (Tipo: Texto ou Numérico)
- **PROD_APRESENTADA**: Quantidade de produção apresentada por
procedimento. (Tipo: Numérico)
- **PROD_APROVADA**: Quantidade de produção aprovada por procedimento.
(Tipo: Numérico)
""")

    # Estatísticas descritivas
    st.header("Estatísticas Descritivas")
    st.write("Aqui estão as principais estatísticas dos dados numéricos:")
    st.write(df.describe())

```

```

# Visualizar os dados de produção apresentada e aprovada
if "PROD_APRESENTADA" in df.columns and "PROD_APROVADA" in
df.columns:
    st.header("Produção Apresentada e Aprovada")
    st.write("Distribuição da produção apresentada e aprovada:")
    st.line_chart(df[["PROD_APRESENTADA", "PROD_APROVADA"]])

# Verificar valores nulos
st.header("Verificação de Valores Faltantes")
missing_values = df.isnull().sum()
st.write(missing_values[missing_values > 0])

else:
    st.warning("Nenhum dado disponível no banco de dados para gerar o resumo.")

# Página de visualização de dados
def data_visualization_page():
    # Função para carregar opções de filtro (Unidade de Saúde e Ano)
    @st.cache_data(show_spinner=False)
    def fetch_filter_options():
        client = MongoClient("mongodb://localhost:27017/")
        db_analise = client["ANALISE"]
        collection_sia_analise = db_analise["SIA_ANALISE"]

        # Obter valores únicos para o campo FANTASIA (Unidade de Saúde)
        fantasia_options = collection_sia_analise.distinct("FANTASIA")

        client.close()

        return fantasia_options

    # Função para conectar ao MongoDB e buscar dados filtrados
    def fetch_filtered_data_from_mongo(fantasia=None, ano=None):
        client = MongoClient("mongodb://localhost:27017/")
        db_analise = client["ANALISE"]
        collection_sia_analise = db_analise["SIA_ANALISE"]

        # Criar filtro dinâmico
        query_filter = {}
        if fantasia:
            query_filter["FANTASIA"] = fantasia
        if ano:
            query_filter["PA_MVMR"] = {"$regex": f"^{ano}" }

        # Obter os documentos da collection SIA_ANALISE aplicando o filtro
        data = list(collection_sia_analise.find(query_filter, {"_id": 0}))

        client.close()

        return data

```

```

# Obter opções de filtro da base de dados
fantasia_options = fetch_filter_options()

# Sidebar para filtros
st.sidebar.header("Filtros de Dados")

# Permitir a busca pelo nome FANTASIA (Unidade de Saúde) com autocomplete
selected_fantasia = st.sidebar.selectbox("Unidade de Saúde",
options=fantasia_options, index=0)

# Campo para selecionar apenas o ano
selected_ano = st.sidebar.text_input("Ano", value="2024")

# Botão para carregar os dados
if st.button('Carregar Dados Filtrados'):
    # Carregar os dados filtrados do MongoDB
    data = fetch_filtered_data_from_mongo(
        fantasia=selected_fantasia,
        ano=selected_ano
    )

    df = convert_to_dataframe(data)

    # Verificar se os dados foram carregados corretamente
    if not df.empty:
        st.success("Dados filtrados carregados com sucesso!")
        st.write(f"Número de registros mostrados: {len(df)}")

        # Exibir os dados em formato de tabela
        st.table(df)

        # Exibir um resumo estatístico dos dados
        st.write("Resumo Estatístico:")
        st.write(df.describe())
    else:
        st.warning("Nenhum dado encontrado com os filtros aplicados.")

# Página de introdução
def introduction_page():
    # Carregar a imagem do SUS
    st.image("docs/img/sus.webp", use_column_width=True)

    # Texto introdutório
    st.title("Bem-vindo ao Projeto de Visualização de Dados do SUS")
    st.write("""
    Este projeto utiliza dados oriundos do FTP do SUS e tem como objetivo auxiliar os
    gestores públicos na elaboração
    de políticas públicas de saúde, bem como ajudar os usuários a identificar onde buscar
    atendimento.
    """)

```

Através dessa plataforma, é possível visualizar e analisar dados de saúde de forma acessível e interativa.

```
""")

# Seção de links úteis
st.header("Links Úteis")
st.markdown("""
- [FTP do SUS
(Dados)](ftp://ftp.datasus.gov.br/dissemin/publicos/SIASUS/200801_/Dados/): Local
para busca dos dados.
- [FTP do SUS
(Manual)](ftp://ftp.datasus.gov.br/dissemin/publicos/SIASUS/200801_/Doc/): Local
com informações técnicas dos arquivos utilizados.
- [DATASUS](https://datasus.saude.gov.br/transferencia-de-arquivos/): Local para
pesquisa dos dados sem necessidade de utilização do FTP.
""")

# Função principal para o controle de páginas
def main():
    st.sidebar.title("Navegação")
    pages = {
        "Introdução": introduction_page,
        "Data Summary Report": data_summary_report_page,
        "Visualização de Dados": data_visualization_page
    }
    choice = st.sidebar.radio("Escolha a página", list(pages.keys()))
    pages[choice]()

# Executar a aplicação
if __name__ == '__main__':
    main()
```

ANEXO PROJECT CHARTER

Project Charter: Dashboard de Visualização de Dados do DataSUS com Streamlit

1. Título do Projeto

Desenvolvimento de Dashboard de Visualização de Dados do DataSUS utilizando Streamlit

2. Objetivo do Projeto

Desenvolver um dashboard interativo com Streamlit que simplifique a visualização e análise dos dados do DataSUS para gestores públicos de saúde. O dashboard será projetado para ser intuitivo, facilitando a compreensão e a tomada de decisões baseadas em dados, substituindo a complexidade das ferramentas atuais como Tabwin e Tabnet.

3. Justificativa do Projeto

Os gestores públicos de saúde enfrentam desafios significativos ao usar ferramentas complexas como Tabwin e Tabnet para analisar os dados do DataSUS. Este projeto visa desenvolver um dashboard com Streamlit, que simplifica o processo de análise, permitindo aos gestores tomarem decisões mais rápidas e informadas, contribuindo para uma gestão mais eficiente da saúde pública.

4. Escopo do Projeto

Inclusões:

- Coleta, tratamento e análise dos dados do DataSUS (incluindo armazenamento em base de dados própria).
- Desenvolvimento de um dashboard interativo utilizando Streamlit.
- Criação de visualizações personalizadas e relatórios automáticos.
- Interface amigável para gestores de saúde, com funcionalidades de filtragem e exploração de dados.
- Treinamento para uso do dashboard e documentação técnica.

Exclusões:

- Desenvolvimento de infraestrutura física.
- Expansão para dados fora do escopo do DataSUS.

5. Entregáveis Principais

- Pipeline de coleta e tratamento de dados do DataSUS.
- Versão beta do dashboard em Streamlit para feedback dos usuários.

- Versão final do dashboard pronta para uso.
- Documentação detalhada do código e manuais de uso.
- Sessões de treinamento para usuários finais.

6. Cronograma de Alto Nível

- **Fase de Planejamento:** 1 semana
- **Coleta e Tratamento de Dados:** 2 semanas
- **Desenvolvimento do Dashboard:** 3 semanas
- **Testes e Feedback com Usuários:** 1 semana
- **Treinamento e Implementação:** 1 semana
- **Lançamento Oficial:** 1 semana
- **Suporte e Manutenção:** Contínuo após o lançamento

8. Stakeholders Principais

- **Patrocinador do Projeto:** Ministério da Saúde / Secretarias Estaduais e Municipais de Saúde
- **Gerente do Projeto:** Wanderson Rafael Mendonça Batista
- **Equipe de Ciência de Dados:** Cientistas de dados, engenheiros de dados e desenvolvedores especializados em Python e Streamlit.
- **Usuários Finais:** Gestores Públicos de Saúde, Epidemiologistas, Profissionais de Saúde Pública.
- **Parceiros:** Universidades e Centros de Pesquisa, Fornecedores de Infraestrutura Tecnológica.

9. Riscos Principais

- **Risco de Qualidade dos Dados:** Problemas com a qualidade e consistência dos dados do DataSUS.
- **Risco de Integração:** Desafios técnicos na integração dos dados do DataSUS no dashboard.
- **Risco de Adoção pelo Usuário:** Baixa adesão dos gestores ao novo dashboard.
- **Risco de Escopo:** Mudanças no escopo do projeto durante o desenvolvimento, impactando prazos e custos.

10. Arquitetura do Aplicativo

A arquitetura do aplicativo será projetada para maximizar a eficiência e a usabilidade, garantindo que o dashboard possa lidar com grandes volumes de dados e fornecer visualizações em tempo real ou quase em tempo real. A seguir estão os principais componentes da arquitetura:

- **Frontend (Interface de Usuário):**
 - Desenvolvido em Streamlit para criar uma interface web interativa e responsiva.
 - Integração com bibliotecas de visualização de dados, como Plotly, Matplotlib e Altair, para gráficos e visualizações dinâmicas.
 - Funcionalidades de filtragem, pesquisa e exportação de relatórios em formatos como PDF e Excel.
- **Backend (Lógica de Negócio):**
 - Python como linguagem principal para processamento de dados e lógica de negócios.
 - Integração com APIs do DataSUS para extração e atualização contínua dos dados.
 - Processamento de dados utilizando Pandas e Numpy para manipulação de grandes datasets.
- **Banco de Dados:**
 - Utilização de bancos de dados relacionais (ex. PostgreSQL) ou NoSQL (ex. MongoDB) para armazenar e gerenciar dados históricos e processados.
 - Mecanismos de caching (ex. Redis) para melhorar a velocidade de acesso a dados frequentemente utilizados.

11. Critérios de Sucesso

- **Entrega do dashboard no prazo e dentro do orçamento.**
- **Alta taxa de adoção e satisfação dos usuários.**
- **Melhoria na eficiência e precisão das análises de dados pelos gestores públicos.**
- **Documentação completa e treinamentos eficazes para os usuários finais.**