### ****Projeto "Docker Data Science Environment"****

**Autor: Eric Pimentel**

**Belém-PA - 2025**

### ****Roteiro Completo: Checklist do Projeto****

#### ****Fase 1: Configuração Inicial****

1. **Criar o Repositório GitHub** :
   * Criar o repositório no GitHub (já feito!).
   * Definir a estrutura de pastas conforme sugerido anteriormente.
   * Adicionar o arquivo README.md com a documentação inicial.
2. **Configurar Ambiente Local** :
   * Instalar Docker e Docker Compose na sua máquina.
   * Testar o Docker com um comando simples (docker run hello-world).
3. **Definir o Arquivo** docker-compose.yml:
   * Criar o arquivo docker-compose.yml na raiz do projeto.
   * Definir os serviços básicos (Jupyter, PostgreSQL, MetaBase, Grafana, Prometheus) e suas conexões via rede Docker.

#### ****Fase 2: Configuração dos Serviços****

1. **Configurar o Jupyter Notebook** :
   * Criar a pasta jupyter/ e adicionar o Dockerfile.
   * Criar o arquivo requirements.txt com as bibliotecas Python necessárias.
   * Testar o contêiner Jupyter isoladamente (docker-compose up jupyter).
2. **Configurar o PostgreSQL** :
   * Criar a pasta postgres/ e adicionar o Dockerfile.
   * Criar o arquivo init.sql com scripts SQL iniciais.
   * Configurar o volume Docker para persistir dados (./volumes/postgres).
   * Testar o contêiner PostgreSQL isoladamente (docker-compose up postgres).
3. **Configurar o MetaBase** :
   * Criar a pasta metabase/ e adicionar o Dockerfile.
   * Criar o arquivo config.env com variáveis de ambiente para conectar ao PostgreSQL.
   * Testar o contêiner MetaBase isoladamente (docker-compose up metabase).
4. **Configurar o Prometheus** :
   * Criar a pasta prometheus/ e adicionar o Dockerfile.
   * Criar o arquivo prometheus.yml com a configuração de métricas.
   * Configurar o volume Docker para persistir dados (./volumes/prometheus).
   * Testar o contêiner Prometheus isoladamente (docker-compose up prometheus).
5. **Configurar o Grafana** :
   * Criar a pasta grafana/ e adicionar o Dockerfile.
   * Criar o arquivo grafana.ini com configurações personalizadas.
   * Configurar o volume Docker para persistir dados (./volumes/grafana).
   * Testar o contêiner Grafana isoladamente (docker-compose up grafana).

#### ****Fase 3: Integração dos Serviços****

1. **Conectar os Serviços via Rede Docker** :
   * Garantir que todos os serviços estejam conectados à mesma rede Docker (data-science-network).
   * Testar a comunicação entre os serviços:
     + Jupyter ↔ PostgreSQL.
     + MetaBase ↔ PostgreSQL.
     + Grafana ↔ Prometheus.
2. **Centralizar os Volumes Docker** :
   * Garantir que todos os volumes estejam centralizados no diretório ./volumes/.
   * Testar a persistência de dados para cada serviço.

#### ****Fase 4: Testes e Validação****

1. **Testar o Ambiente Completo** :
   * Executar todos os serviços simultaneamente (docker-compose up -d).
   * Acessar as interfaces:
     + Jupyter Notebook: http://localhost:8888.
     + MetaBase: http://localhost:3000.
     + Grafana: http://localhost:3001.
     + Prometheus: http://localhost:9090.
2. **Validar Funcionalidades** :
   * Testar a análise de dados no Jupyter Notebook.
   * Criar dashboards no MetaBase.
   * Monitorar métricas no Grafana.
3. **Simular um Caso de Uso** :
   * Criar um script Python no Jupyter Notebook para coletar dados e armazená-los no PostgreSQL.
   * Criar um dashboard no MetaBase para visualizar os dados.
   * Monitorar o desempenho do sistema no Grafana.

#### ****Fase 5: Documentação e Publicação****

1. **Atualizar a Documentação** :
   * Atualizar o README.md com instruções detalhadas sobre como configurar e usar o ambiente.
   * Incluir screenshots das interfaces (Jupyter, MetaBase, Grafana).
   * Adicionar o diagrama do ecossistema.
2. **Publicar no GitHub** :
   * Fazer commit e push de todas as alterações para o repositório GitHub.
   * Verificar se o repositório está organizado e fácil de navegar.
3. **Compartilhar no LinkedIn** :
   * Escrever uma postagem explicando o projeto e compartilhar o link do repositório.
   * Destacar os diferenciais do projeto (modularidade, replicabilidade, boas práticas).

### ****Cronograma Sugerido****

Aqui está uma sugestão de ritmo para executar o projeto:

| ****Fase**** | ****Duração Estimada**** |
| --- | --- |
| Configuração Inicial | 1-2 dias |
| Configuração dos Serviços | 5-7 dias |
| Integração dos Serviços | 2-3 dias |
| Testes e Validação | 2-3 dias |
| Documentação e Publicação | 2-3 dias |

Total estimado: **2-3 semanas** (dependendo do tempo disponível diariamente).