CondoManage

Manual de intruções de como instalar o projeto em máquina local e resolução das questões do case.

Authors

- @IDevKennedyMoreira

🔗 Links

PROJECT REPOSITORY

MY PORTFOLIO

LINKEDIN

Local Deployment

Para deploy deste projeto em máquina local é necessário efetuar algumas ressalvas, este projeto foi desenvolvido em sistema operacional MacOs e os passos de instalação podem mudar de acordo com sistema operacional, aqui basearei apenas a instalação em MacOs pois não possuo muita familiaridade com outros sistemas operacionais e não saberia como descrever o passo a passo nesse momento pois n tenho acesso facilitado a outros sistemas operacionais.

Instalando o Java versão 11.

```
brew install openjdk@11
```

Instalando o Apache Spark.

```
brew install apache-spark
```

Instalando o Python.

```
brew install python3
```

Instalar o virtualenv na máquina local.

```
pip install virtualenv
```

Abrir projeto e criar seu ambiente virtual.

```
virtualenv venv
```

Abrir projeto e iniciar o ambiente virtual.

```
source env/bin/activate
```

Instalar os requirements.

```
pip install -r requirements.txt
```

Definir variável de ambiente do airflow.

```
export AIRFLOW_HOME=~/Desktop/case_super_logica
```

Para executar o projeto em modo local sem airflow via terminal.

Para executar o projeto em modo local usando o airflow

```
airflow standalone
```

Acesse http://localhost:8080
logue com usuário admin e senha presente no arquivo
standalone_admin_password e acione a DAG superlogica_data_pipeline

Para execução da pipeline streaming saia do Airflow e execute o arquivo
dags/stream_read_data_from_landing.py. Em outro terminal
execute

```
cd dags
```

E para criação de novos arquivos na camada de landing execute

```
python3 landing_read_data_from_external.py
```

Questão 1

A arquitetura de datalake escolhida para a criação dessa projeto foi a medalhão
aqui temos:

Camada landing apenas recebendo os arquivos csv para processamento
camada raw apenas transforma os arquivos da camada landing em formato parquet;

camada refined aplica regras de negócio e gera a OBT (One Big Table) modalagem escolhida
por mim para este projeto;

camada trusted é uma cópia da camada refined com dados prontos para análise.

Questão 2

A ingestão de dados desse projeto foi criada a partir de uma simulação da criação de
arquivos csv a classe responsavel por isso é a DataGenerator presente no arquivo
dags/models/datagenerator.py ela é orquestrada a partir de dags/landing_read_data_from_external.py
e por sua vez os dados entram na camada raw através de dags/raw_read_data_from_landing.py
a explicação de como cada arquivo funciona está em seus comentários internos.

Questão 3

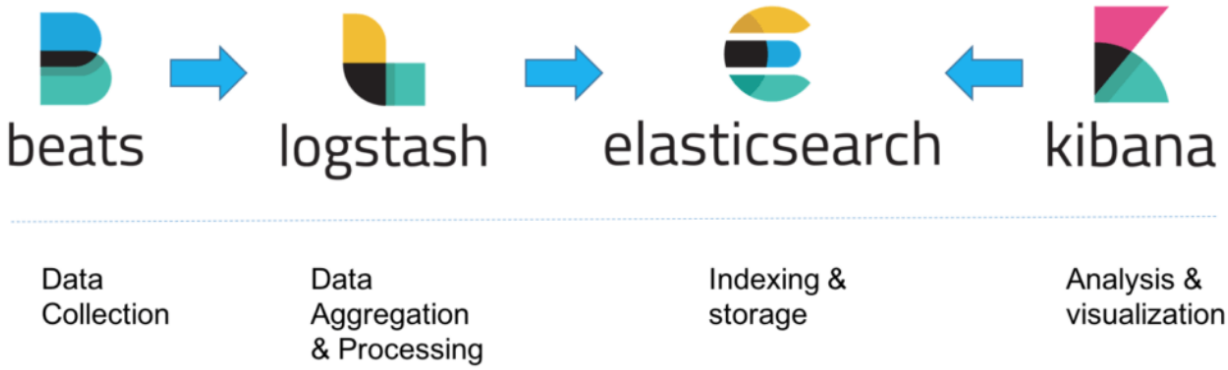
A resolução dessa questão está presente em dags/report_read_data_from_trusted.py

Questão 4

Vide comentários internos do arquivo dags/raw_streaming_read_data_from_landing_townhouse.py
lá é feita a primeira parte da ingestão usando streaming, porém pipeline exposta na DAG e no arquivo
dags/superlogica_data_pipeline_local.py é do tipo batching.

Questão 5

Para definicão de documentação usaria um repositório de documentação como o confluence e também geraria comentário seguindo a PEP8 e com a ferramenta mkdocs conseguiria fazer a documentação automática
a partir dos cometários nos jobs.
Sobre as questões envolvendo logging e monitoramento decidiria por usar a ELK stack tendo assim o Beatsfile
efetuando pooling em arquivos de dados e logs presentes na ferramenta de orquestração como o Airflow por sua vez o Beatsfile aciona o Logstash com a mensageria para que ele definir o melhor índice do ElasticSearch para inserir essa mensagem, por fim teriamos a liberdade de definir boards no Kibana com os logs propostos.
Para enriquecimento dos logs poderiamos criar uma classe de log com um decorator de logging onde este por sua vez persistiria a mensagem em arquivos de logs de cada job e também colocaria o ELK stack para enxergar estes mesmos logs criando assim uma solução de monitoramento robusta.



Para a arquitetura citada acima a presença de uma instância de Kafka entre o Beats e o Logstash pode sanar possíveis gargalos de processamento.

Badges

Licença: