Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

**Instituto de Gestão e Tecnologia da Informação**

**Relatório do Projeto Aplicado**

Lives analytics para negócios

Pedro Henrique Faria Teixeira

Orientador(a):

Daniel Viana

01/06/2022

****

**Pedro Henrique Faria Teixeira**

**INSTITUTO DE GESTÃO E TECNOLOGIA DA INFORMAÇÃO**RELATÓRIO DO PROJETO APLICADO

Lives analystics para negócios

Relatório de Projeto Aplicado desenvolvido para fins de conclusão do curso [MBA Engenharia de Dados].  
  
Orientador (a): Daniel Viana

**Araguari  
01/06/2022**

**Sumário**

[1. CANVAS do Projeto Aplicado 4](#_Toc101281510)

[1.1 Desafio 5](#_Toc101281511)

[1.1.1 Análise de Contexto 5](#_Toc101281512)

[1.1.2 Personas 6](#_Toc101281513)

[1.1.3 Benefícios e Justificativas 7](#_Toc101281514)

[1.1.4 Hipóteses 8](#_Toc101281515)

[1.2 Solução 9](#_Toc101281516)

[1.2.1 Objetivo SMART 9](#_Toc101281517)

[1.2.2 Premissas e Restrições 11](#_Toc101281518)

[1.2.3 Backlog de Produto 13](#_Toc101281519)

[2. Área de Experimentação 14](#_Toc101281520)

[2.1 Sprint 1 16](#_Toc101281521)

[2.1.1 Solução 16](#_Toc101281522)

[● Evidência do planejamento: 16](#_Toc101281523)

[● Evidência da execução de cada requisito: 16](#_Toc101281524)

[● Evidência dos resultados: 16](#_Toc101281525)

[2.1.2 Experiências vivenciadas 16](#_Toc101281526)

[2.2 Sprint 2 17](#_Toc101281527)

[2.2.1 Solução 17](#_Toc101281528)

[● Evidência do planejamento: 17](#_Toc101281529)

[● Evidência da execução de cada requisito: 17](#_Toc101281530)

[● Evidência dos resultados: 17](#_Toc101281531)

[2.2.2 Experiências vivenciadas 17](#_Toc101281532)

[2.3 Sprint 3 18](#_Toc101281533)

[2.3.1 Solução 18](#_Toc101281534)

[● Evidência do planejamento: 18](#_Toc101281535)

[● Evidência da execução de cada requisito: 18](#_Toc101281536)

[● Evidência dos resultados: 18](#_Toc101281537)

[2.3.2 Experiências vivenciadas 18](#_Toc101281538)

[3. Considerações Finais 19](#_Toc101281539)

[3.1 Resultados 19](#_Toc101281540)

[3.2 Contribuições 19](#_Toc101281541)

[3.3 Próximos passos 19](#_Toc101281542)

## 1. CANVAS do Projeto Aplicado

Figura conceitual, que representa todas as etapas do Projeto Aplicado.



## Desafio

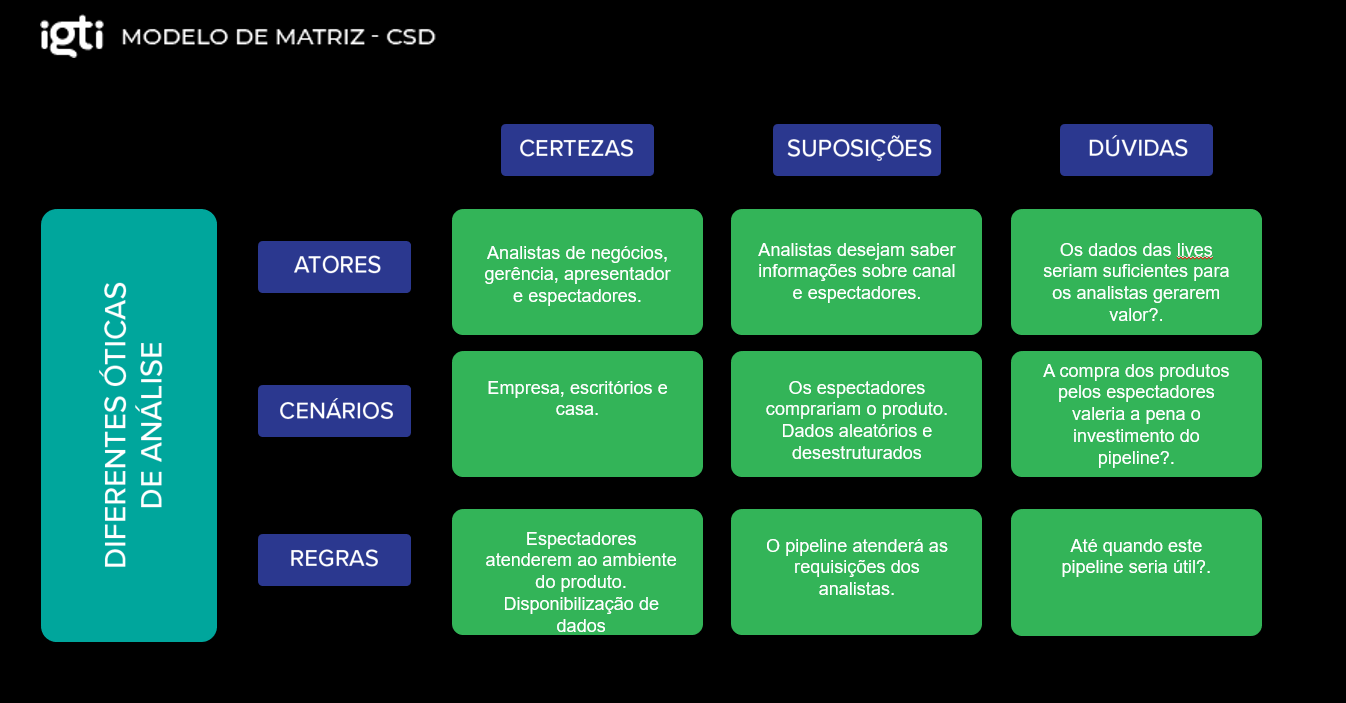
### 1.1.1 Análise de Contexto

Uma empresa possui um produto e quer encontrar uma boa maneira de fazer divulgações a fins de aumentar suas vendas. Dado que hoje estamos vivenciando dias onde podcasts e lives estão sendo realizados diariamente e em praticamente qualquer horário do dia, esta empresa então optou por divulgar seus produtos por meio dessas lives, porém ela não sabe como escolher o melhor canal, apresentador e nicho de espectadores.

Sendo assim, a empresa precisa construir uma aplicação para fazer a captação dos dados dessas lives e disponibilizar em um datalake para que os seus analistas de dados possam conectar suas ferramentas de BI a fim de fazer as análises e relatórios para entregar a gerência e assim estes realizarem o levantamento de negócio e avaliação de viabilidade, para então contratarem o canal ou apresentador para representarem a sua marca.

Dado isso, o desafio para resolver esse problema é construir um pipeline para extrair e concentrar as informações dos espectadores das lives/podcasts em um datalake, disponibilizando esses dados para os analistas fazerem as análises, aplicando regras de disponibilização que deve ser documentada e seguir todos os padrões de LGPD para que assim a gerência possa analisar a viabilidade do anuncio do produto em determinado canal.

Matriz CSD:



Observação POEMS:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PESSOAS | OBJETOS | AMBIENTE | MENSAGEM | SERVIÇOS |
| Quem está presente no contexto em análise? | Que objetos fazem parte do ambiente? | Quais são as características do ambiente? | Que mensagens são comunicadas? | Quais serviços são oferecidos? |
| Apresentador | Computadores, microfones, fones de ouvidos, assentos e mesas | Ambiente descontraído onde se pode conversar sobre assuntos diversos | Mensagem verbal informal ou formal sobre um assunto ou acontecimento | Comunicação com pessoas por meio de uma plataforma online |
| Espectador | Computadores, notebooks, mesas, assentos e smartphones | Ambiente diverso, formal ou informal. Pode ser sua casa, carro ou trabalho. | Mensagem de texto informal sobre um assunto ou acontecimento | Comentários por meio da plataforma online |
| Analista de dados | Notebooks, telas, mesas e assentos | Ambiente formal ou informal podendo ser a sede da empresa ou sua casa | Relatórios com informações sobre as *lives* | Análises e dashboards sobre os comentários dos espectadores |
| Gerência | Notebooks, telas, mesas e assentos | Ambiente formal ou informal podendo ser a sede da empresa ou sua casa | Relatórios sobre viabilidade do negócio | Análise decisória para a contratação de um canal ou apresentador |
| Registros  - Falta de dados  - Não existem relatórios nem análise de negócio | | | Insights  - Hoje as *lives* atraem uma alta taxa de espectadores e isso pode ser utilizado para vendas de produtos  - Criar pipeline de dados para automatizar a captação de dados | |

### 1.1.2 Personas



* + Nome: Laysa Vieira
  + Idade: 28 anos
  + Gênero: Feminino
  + Estado civil: Solteira
  + Educação: Pós-graduada em Gestão de Negócios e superior completo em Ciência da Computação
  + Profissão: Gerente de Projetos
  + Características comportamentais: É uma pessoa séria e formal, não gosta de bebidas alcoólicas e por isso frequenta somente ambientes reservados. Gosta bastante de sair com amigos para jantar. Seu hobby favorito é dormir.
  + Objetivos: Ter indicadores de espectadores que sejam capazes e tenham interesse em comprar produtos pela internet.



### 1.1.3 Benefícios e Justificativas

Os benefícios desse desafio são os seguintes: Criar uma estrutura de dados consistentes que será utilizada pela área de negócio da empresa para acelerar o processo de vendas de seus produtos, isso inclui, agilização do processo de encontrar público alvo, aumento de capital em vendas, maior porcentagem de venda por publicação, mitigação de insatisfação de compra do produto e impacto financeiro negativo. Com os dados disponíveis no *datalake* a equipe de analista de dados poderá focar somente na parte de criação de dashboards e relatórios pois toda a infraestrutura dos dados já estará pronta para ser utilizada.

Outros benefícios que podem ser citados sobre a criação do *pipeline* de dados:

- Centralização dos dados;

- Segurança e backup de dados;

- Automatização de processos;

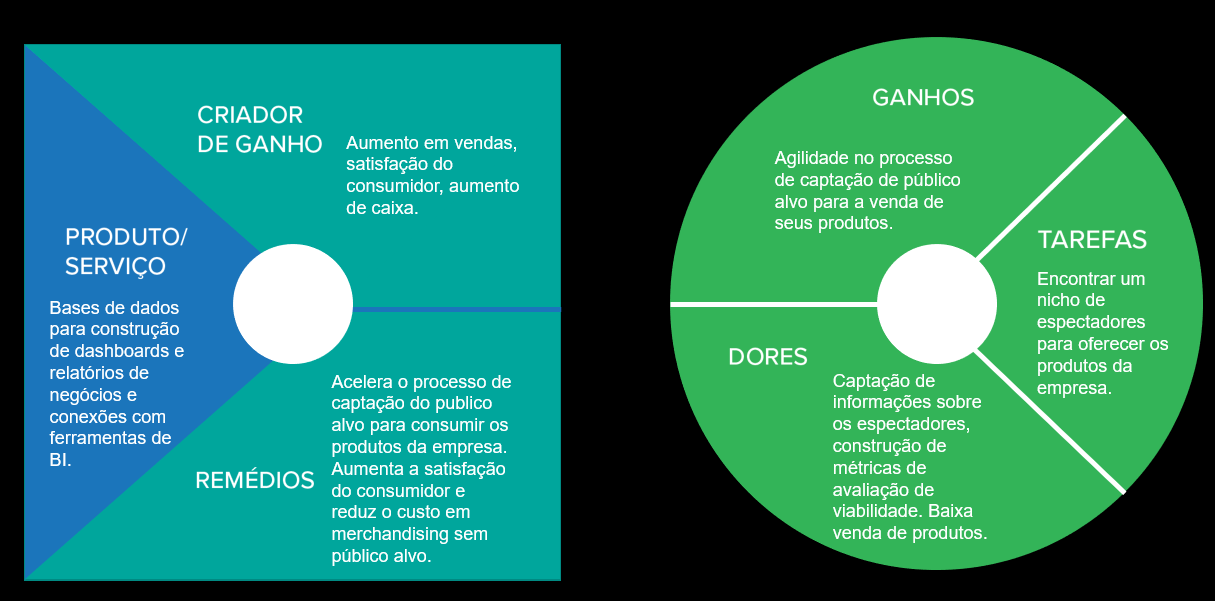
- Compartilhamento de informações

- Aumento de produtividade da equipe;

- Redução de esforços desnecessários.

**BluePrint:**

|  |  |
| --- | --- |
| Itens | Detalhamento |
| Objetivos | Conhecer o perfil do espectador, encontrar indicadores de adesão ao produto, realizar vendas de produtos por meio de canais que façam *lives*. |
| Atividades | Procurar indicações, procura opções de canais, procura pelos cronogramas das *lives*, encontrar um assunto de seu agrado, assistir ao programa analisar dashboards e relatórios, relatar viabilidade de merchandising do produto. |
| Questões | O que eu quero assistir? Quais os assuntos possíveis? Será que vai ser divertido ou entediante? Vai agregar ao meu conhecimento? Será que meus amigos também assistirão? O que eu irei comentar no chat da *live*? |
| Barreiras | Acesso a internet, local apropriado, forma de assistir ou ouvir, dados desatualizados, falta de relatórios, falta de espectadores e canais. |
| Ações do cliente | Quer assistir a um podcast, procura uma plataforma, faz o login, procura um canal que goste, espera o cronograma, começa a assistir. |
| Funcionalidades | Captação de público alvo para a venda de produtos, possibilidade criação de dashboards e relatórios analíticos, concentração de massa de dados para análise, retenção de dados históricos. |
| Interação | Possibilitar análise de negócio, diversificação de dados para análise, conexões com ferramentas de BI, classificação dos dados de espectadores |
| Mensagem | Descubra o melhor canal para divulgar seus produtos, encontre o nicho de pessoas para vender seus produtos, aumente suas vendas com a análise dos nossos dados, veja qual o ambiente seu produto se adapta as vendas. |
| Onde ocorre | Casa, escritório, site, aplicativos, internet. |
| Tarefas aparentes | Dashboards, relatório de análises, propagandas dos produtos. |
| Tarefas escondidas | Bancos de dados locais, esteiras de pipeline, conexões de ferramentas. |
| Processos de suporte | Documentação dos dashboards, documentação das bases de dados, documentação de relatórios, suporte ao consumidor, reembolso da compra. |
| Saída desejável | Aumentar as vendas da empresa, compras de produtos por espectadores de *lives*, aumento do caixa da empresa, satisfação de quem compra os produtos. |



### 1.1.4 Hipóteses

|  |  |  |
| --- | --- | --- |
| Nº | Observação | Hipótese |
| 1 | Dados são gerados aleatoriamente em tempo real/quase real nas plataformas | Identificar a melhor forma de extrai-los e enriquecer os dados |
| 2 | Os dados estão desestruturados | Identificar locais de armazenamento |
| 3 | Dependência de interação de espectadores para geração de dados | Captar canais com maior volume de acesso |
| 4 | Baixo nível de informação dos dados | Criar mecanismo de classificação |
| 5 | Acessibilidade aos dados pelo pessoal de *analytics* | Disponibilizar os dados em uma ferramenta de BI |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ideias | Critérios de comparação | | | | | | Somatório |
| C1 | C2 | C3 | C4 | C5 | C6 |
| I1 | 1 | 4 | 3 | 5 | 2 | 3 | 18 |
| I2 | 2 | 3 | 1 | 2 | 5 | 2 | 15 |
| I3 | 3 | 4 | 5 | 2 | 1 | 1 | 16 |
| I4 | 4 | 2 | 3 | 2 | 3 | 5 | 19 |
| I5 | 2 | 3 | 5 | 2 | 3 | 2 | 17 |

## 1.2 Solução

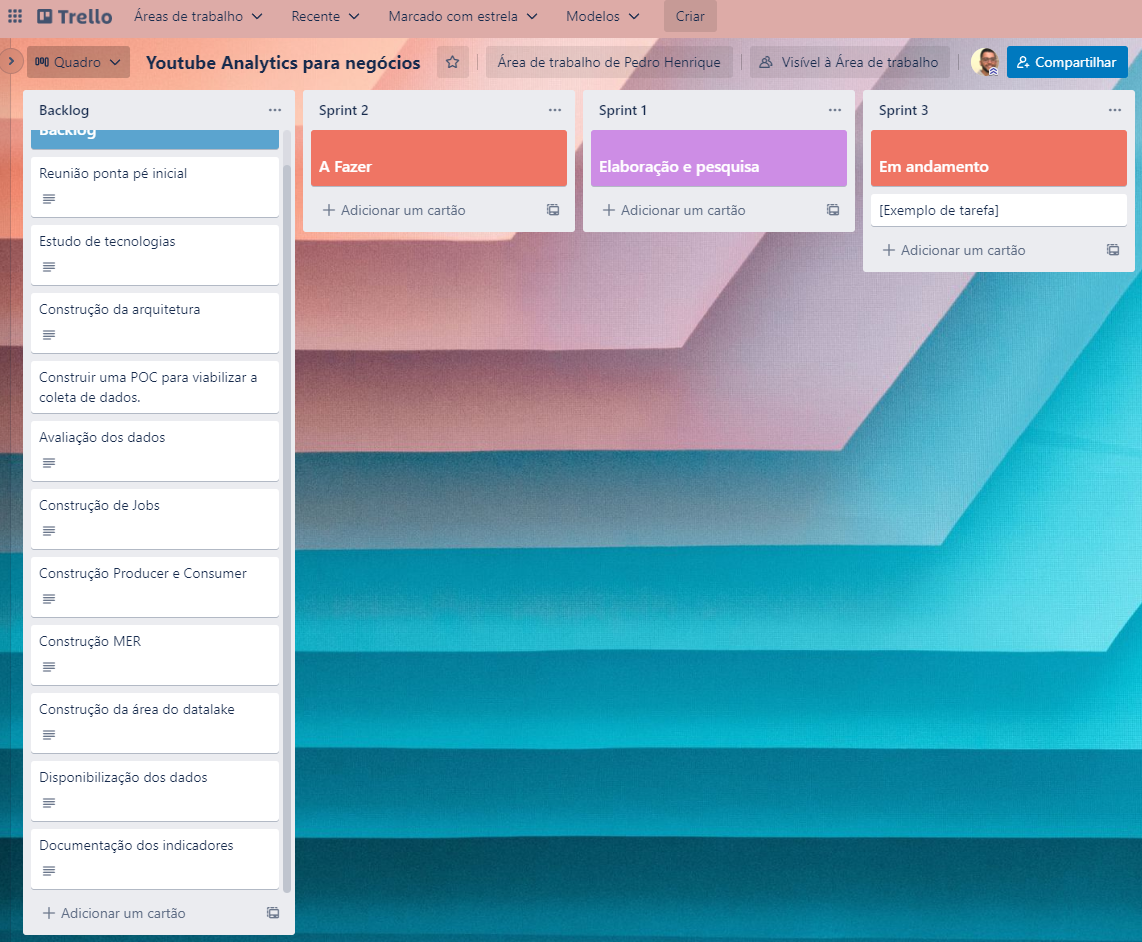
### Objetivo SMART

Construir até o encerramento da disciplina um pipeline de dados que atenda a demanda dos analistas de dados e gerência da empresa onde conseguiam extrair informações para realizar vendas de produtos da empresa para 10% dos espectadores de cada *live*/podcast realizada onde apareça o anúncio. Aumentar em 5% a taxa de venda dos produtos mensalmente. Encontrar canais onde os espectadores pertençam ao nicho do nosso produto ou então espectadores que estão aptos a pagarem pelo produto se divulgado de maneira correta pelo apresentador. Considerando a atração que as *lives* e podcasts estão recebendo do público, se torna um meio mais atual de se realizar merchandising, atingindo mais rápido o público-alvo.

### 1.2.2 Premissas e Restrições

|  |  |  |  |
| --- | --- | --- | --- |
| Risco Identificado | Impacto Potencial | Ações preventivas | Ações corretivas |
| Dados aleatórios | Alto | Criar processos de classificação de dados | Classificar os dados |
| Dados inconsistentes ou duplicados | Alto | Criar processos de validação antes do armazenamento | Identificar e eliminar dados duplicados ou inconsistentes |
| Dados desestruturados | Alto | Criar estruturação de dados para o armazenamento | Estruturar os dados |
| Dados indisponíveis ou ausentes | Alto | Disponibilizar os dados para acesso dos analistas | Regras de disponibilização imediata dos dados mediante solicitação |
| Dados sensíveis de espectadores | Alto | Criar sistema de identificação de dados sensíveis | Deixar os dados sensíveis anônimos e aplicar LGPD |

### 1.2.3 Backlog de Produto



# 2. Área de Experimentação

## 

## 2.1 Sprint 1

### 2.1.1 Solução

#### Evidência do planejamento:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

#### Evidência da execução de cada requisito:

**Reunião ponta pé inicial:**

Foi realizada duas reuniões com a gerência e a equipe que será responsável por fazer a construção do projeto *Lives* *analytics* para negócios, com o intuito de discutir qual o problema a ser solucionado, quais os passos para solucionar os problemas, quais seriam os possíveis impeditivos e quais os são os principais indicadores para o negócio. Nessa reunião foi discutido também as possíveis tecnologias a serem utilizadas e o porquê de serem utilizadas, além do formato de extração e armazenamento dos dados

**Estudo de tecnologias**:

Foi feito um estudo das tecnologias para levantar o custo de uso, vantagens e desvantagens. Esse estudo é importante pois é a partir dele que levantamos a possibilidade de futuramente quando o projeto estiver mais evoluído, trocar as ferramentas por outras mais performáticas. Tendo o estudo inicial é possível realizar a análise de viábilidade de novas tecnologias fazendo a compensação de custo com o faturamento do projeto.

**Construção da arquitetura:**

Essa arquitetura foi projetada pensando em 3 aspectos, disponibilidade dos dados, custo, escalabilidade.

Escalável pois neste primeiro momento será construído em cima de ferramentas locais que são facilmente escaláveis para a nuvem, o que também já impacta na redução de custos, pois são ferramentas ***opensources*** ou então possuem um acesso grátis inicial.

O aspecto de custo se dá por conta de as ferramentas em sua maioria serem de acesso livre e que quando é escalada para a nuvem o custo aumenta, porém o retorno de valor ao negócio é na maioria das vezes maior do que é gasto com as ferramentas.

Já em relação a disponibilidade dos dados, tem se um *datalake* onde é possível armazenar os dados desde quando são “crus” e até quando são mais performáticos e estruturados, tendo possibilidade de versionamento e backup, junto com o banco de dados MySQL que servirá como ferramenta de consulta ao *datalake*.

**Construir uma POC para viabilizar a coleta de dados:**

Foi construído 2 códigos testes para fazer a extração de uma amostra de dados o KafkaProducer e o KafkaConsumer, assim testando parte das tecnologias envolvidas no projeto e viabilizando a análise da estrutura da massa de dados que será extraída das *lives*.  
Um dos códigos é o responsável por fazer a captação dos dados e o outro o armazenamento no *datalake* e podendo levar direto a ferramenta de visualização dos dados o MySQL. Esses dois códigos são exemplos, mas será a partir dos dois que o projeto será evoluído.

**Avaliação dos Dados:**

Foi feita a avaliação dos dados, o que não passa de um estudo da estrutura de como os dados são extraídos, isso para que possamos ter uma noção do valor que aqueles dados geram e fazer o mapeamento dos indicadores relevantes e não relevantes para o negócio. Com esses indicadores os analistas e cientistas de dados conseguiram levantar informações importantes para o negócio, como a participação do autor, tanto monetária quanto proativa. Se o autor é um usuário ativo em *lives* ou não, se o autor tem disposição de comprar um produto ou até mesmo levantar hipóteses de sentimentalidade em cima das mensagens de texto.

#### Evidência dos resultados:

**Reunião ponta pé inicial:**

Nestas reuniões chegamos à conclusão de que o banco de dados a ser utilizado para consulta será o MySQL devido a fácil acessibilidade pelos membros da equipe e por ser um banco de dados relacional onde os analistas e cientistas de dados já possuem certa facilidade em consultar com o SQL. Porém as tecnologias posteriores que levaram os dados até o banco foram deixadas em aberto para uma avaliação e estudo das tecnologias para levantar aspectos de performance, manutenção e custo. Além do banco de dados os dados serão armazenados em um *datalake* com a estrutura de pastas (*bronze*, *silver*, *gold*), para manter dos arquivos segmentada de menos tratado e mais tratado, facilitando também o backup do banco de dados.

|  |  |
| --- | --- |
| **Reunião 18/06/2022** | |
| Participantes | Laysa Vieira, Pedro Teixeira, Marcos Vilela |
| Tema | Ponta pé inicial |
| Assuntos tratados | Primeiros passos, entendimento do problema, ferramentas, disponibilização dos dados |
| Comentários importantes | Os dados são gerados em real-time, assim temos que criar uma estrutura que suporte essa coleta e armazene de forma adequada. Temos que criar uma estrutura que suporte análise em *batchs* para manter dados históricos e backups. |

|  |  |
| --- | --- |
| **Reunião 20/06/2022** | |
| Participantes | Laysa Vieira, Pedro Teixeira, Marcos Vilela |
| Tema | Discussão de indicadores |
| Assuntos tratados | Estrutura do *schema*, relevância dos indicadores, quais os mais importantes |
| Comentários importantes | Os dados são coletados das *lives* no formato *json*, o que já é bem estruturado para dados crus, é um ponto positivo para a evolução do pipeline  Levantamento das principais *features* e hipóteses que podem ser coletadas a partir dos indicadores presentes no *schema* |

**Estudo de tecnologias**:

Para esse projeto é necessário criar uma estrutura que seja capaz de suportar a extração de dados em tempo real, e hoje no mercado tem várias tecnologias, porém a que está sendo mais utilizada no momento é o ***Apache Kafka*** e essa foi a escolhida para uso nesse projeto, dada a sua baixa complexidade de uso e facilidade de integração com outras de linguagens como Java, Go, Python, Scala, etc.

**Prós do uso do Apache Kafka:**

- Integra-se com aplicativos existentes

- Oferece um valor de baixa latência de até 10 milissegundos

- Reduz a necessidade de múltiplas integrações

- Serve como um substituto perfeito para corretores de mensagens tradicionais.

**Contras do uso do Apache Kafka:**

- Falta paradigma vital de mensagens, como filas ponto a ponto

- Ficar aquém em termos de análise

- Tende a se comportar desajeitadamente se o número de filas em um Cluster Kafka aumentar

Como linguagem de programação, foi escolhido o ***Python***, pois é a linguagem mais utilizada na área de dados, tanto na área de engenharia, ciência e analise de dados. É uma linguagem com uma baixa complexidade de uso, possui uma comunidade grande e movimentada e o nível de aprendizado é baixo, pode pecar um pouco em performance porém o ***Spark*** entra no meio para aprimorar o custo benefício olhando pro lado de performance. Como mencionado o Spark, é mais um ponto pela escolha do Python, justamente por possuir integração com o Spark, outra ferramenta muito utilizada para ETL de dados e que será usada neste projeto.

**Prós do uso do Python:**

- Fácil de ler e aprender

- Aumenta a produtividade

- Vasta coleção de biblioteca

- Open-source, grátis e possui vasta comunidade

- Linguagem de programação portatil

- Linguagem interpretada

**Contras do uso do Python:**

- Possui limitação de velocidade

- Não é tão forte em computação mobile

- Pode ocorrer erros de tempo de execução

- Consome muitra memória

- Não é fácil de testar

**Prós do uso do Spark:**

- Rápido

- Fácil de usar

- Possui análise avançada

- Dinâmico por natureza

- Multilingual

- Maior acesso a Big Data

**Contras do uso do Spark:**

- Não possui automatização altomática de processos

- Menor quantidade de algoritmos

- Sistema de gerenciamento de arquivos

- Problema com arquivos pequenos

- Não é adequado para um ambiente multiusuário

Como linguagem de consulta aos dados foi escolhido o SQL por ser a linguagem mais comum, não só na área de dados mas na área de TI no geral.

Prós do uso do SQL:

- Processamento de consultas mais rápido

- Sem habilidades de codificação

- Idioma padronizado

- Portátil

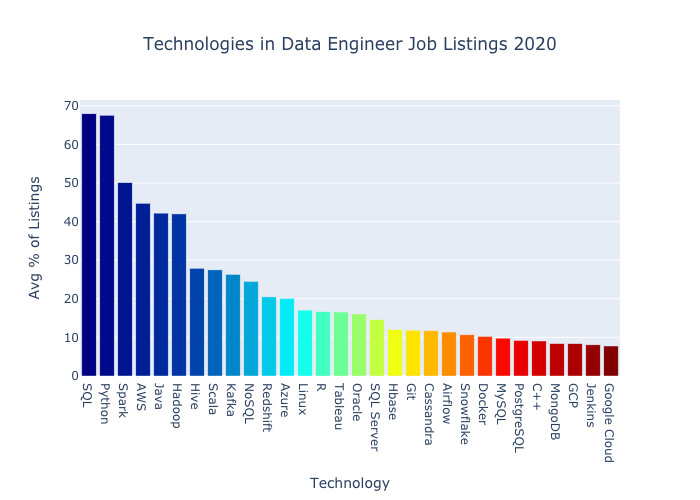
- Linguagem Interativa

- Várias visualizações de dados

Contras do uso do SQL:

- Interface Complexa

- Controle Parcial

Na imagem abaixo temos um comparativo de quais linguagens e ferramentas são mais utilizadas na área de engenharia de dados, o ponto focal desse rojeto e o que levou a escolha das mesmas. 

**Construção da arquitetura:**

**Tabela

Descrição gerada automaticamente com confiança baixa**

**Construir uma POC para viabilizar a coleta de dados:**

Código responsável por fazer a captura dos dados do Youtube/Twitch por meio da biblioteca **pytchat** e disponibilizar em um tópico do Kafka no formato **json**.

Texto

Descrição gerada automaticamente

Código responsável por ler os dados do tópico e assim salvá-lo em um arquivo **json** para ser feito a análise inicial da estrutura do **schema**.

Texto

Descrição gerada automaticamente

**Avaliação dos Dados:**

Exemplo de arquivo **json** que é enviado pelo *KafkaProducer*, nele podemos perceber os indicadores que aparecem quando um autor faz uma ação no *chat* da *live*. Aqui temos diversas informações, sendo elas relevantes e não relevantes para o projeto que será desenvolvido. Julgando o problema a ser solucionado que é encontrar possíveis autores a comprarem um produto, os indicadores que elevam essa possibilidade são os seguintes:

- i***d***: ID do autor que fez a interação com a *live*;

- ***name***: Nome do autor;

- ***message***: Mensagem em texto que o autor enviou;

- ***datetime***: Data da interação que o autor fez na *live*;

- ***isVerified***: Indicador se o autor possui um canal verificado pelo Youtube;

- ***isChatSponsor***: Se o autor é patrocinador do *live* chat;

- ***amountValue:*** Indicador do valor da doação para o canal;

- ***amountString:*** Indicador do valor da doação para o canal;

- ***type***: Esse indicador possui dois valores que podem ser considerados monetários, sendo eles: “***superChat***” que indica que um autor doou dinheiro para fazer alguma propaganda ou interagir na *live* e “***newSponsor***” que indica que um novo autor se tornou um patrocinador.

Texto

Descrição gerada automaticamente

### 2.1.2 Experiências vivenciadas

Ao fazer o levantamento do problema foi preciso realizar pesquisas de *features* que ajudassem na resolução, sendo assim foi pesquisa várias ferramentas visando o melhor custo benefício para dar o andamento ao projeto. Sendo assim tendo uma curva de aprendizado teórica muito grande que ajudará nas construções de futuros projetos que envolva ETL. Todas as tecnologias escolhidas para esse projeto são facilmente escaláveis para um ambiente em nuvem, aumentando ainda mais a escalabilidade, performance, disponibilização dos dados e aplicabilidade de regras de negócios.

Talvez seja necessária aumentar um pouco o escopo inserindo uma ferramenta de consulta a dados em tempo real como o *KSQLDB* para que os analistas possam tirar suas conclusões antes mesmo do dado chegar no *datalake.*

Para a próxima *sprint,* ficará a parte de codificação da estrutura da arquitetura e modelagem do *datalake para armazenar dados históricos*, e construção de um MER padrão para a análise dos dados*.*

## 2.2 Sprint 2

### 2.2.1 Solução

#### Evidência do planejamento:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

#### Evidência da execução de cada requisito:

**Construção da área do *datalake*:**

O *datalake* foi construindo no formato de pastas assim como demonstrado anteriormente no desenho da arquitetura, seguindo o padrão de pastas bronze, silver e gold, onde serão armazenados os dados. Assim como o refinamento desses metais utilizados para a nomenclatura das pastas, serão guardados os dados de acordo com o seu refinamento, bronze o dado bruto vindo direto do *source* no formato *json*, *silver* o dado estruturado e mais performático no formato *parquet* e *gold* as tabelas com o mais alto nível de informações no formato *parquet*.

**Construção MER:**

Foi feita a construção de uma modelo entidade relacionamento entre a mensagem que é captada pelo consumidor Kafka e o autor da mesma, esse modelo foi criado para que possamos abranger o escopo de análises e algoritmos de *marchine learning*. Novos modelos de entidades serão criados de acordo com a evolução do projeto e novas demandas de análises surgirem.

Além da criação da tabela onde serão armazenadas mensagens brutas em parse real-time que passam pelo tópico. A demanda dessa tabela é para que os analistas possam acompanhar o dado previamente e dele extrair *insights* de novas tabelas de negócios que são armazenadas na área *gold*.

**Construção *Producer e Consumer*:**

Foram desenvolvidos dois códigos para fazer a captação e armazenação dos dados, a necessidade para a construção destes scripts se dá ao fato de que para extrair os dados das *lives* está sendo utilizado uma biblioteca de nome *pytchat* que faz a captação do dado em tempo real, e para manter a estrutura de *real-time* ou parse *real-time* é preciso de um Producer, que lê o dado e o disponibiliza em um tópico do Kafka em diferentes formatos, mas para esse projeto está sendo disponibilizado como *json* e é preciso também de um ***Consumer*** que lê essa dado do tópico e faz o armazenamento desse dado no *datalake* ou diretamente no banco de dados. Esses códigos também possuem funcionalidades auxiliares dentro deles para que tudo funciona de forma correta. Essa estrutura de captação do dado utilizando Kafka traz um grande benefício que é a premissa da própria ferramenta de garantir que todos os dados que sejam lidos pelo Producer sejam entregues ao ***Consumer***, então é um grande passo á evolução da vazão de dados, garantindo que todo dado seja entregue e lido do tópico. Outro benefício é que a possibilidade de subir diversos tópicos em paralelo com diversos *producers* e seus respectivos *consumers*, dado que foi utilizado orientação a objetos para construir os mesmos, sendo assim podendo manter um alto volume de dados sendo enviado para o *datalake*.

**Construção Jobs:**

Foi construído ***Jobs*** para fazer a transformação do dado bruto no formato *json* para um formato mais estruturado, o *parquet*. Esse *job* é necessário para deixar o dado mais performático, agilizando os processos de análises e armazenamento em banco de dados, aumentando também a performance em quaisquer requisitos que demande processamento de um grande volume de dados. Para cada live aberta é necessário a construção de um desses *Jobs para realizar a transformação.*

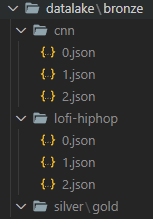
#### Evidência dos resultados:

**Construção da área do *datalake*:**

Assim como no desenho da arquitetura postado anteriormente, foi replicado para o ambiente local.



Aqui em um exemplo de como está sendo armazenado o dado bruto.



**Construção MER:**

Modelo entidade relacionamento entre o autor da mensagem e a respectiva mensagem com indicadores de contribuição.

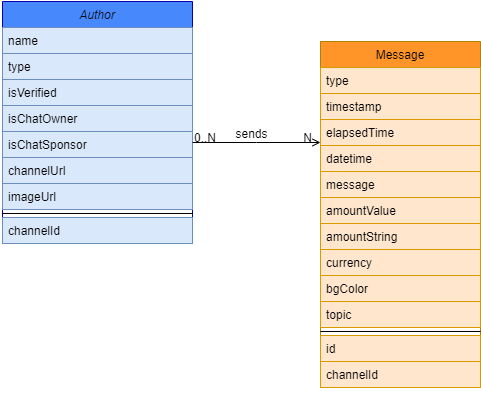
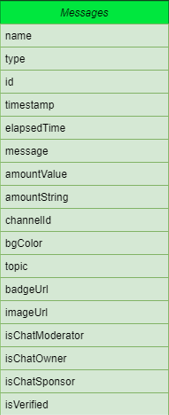


Tabela bruta a onde os analistas poderão acompanhar o dado em *parse real-time* sendo armazenado no MySql.



**Construção *Producer e Consumer*:**

Código do ***Producer***, responsável pela captação do dado das *lives* e disponibilização no tópico:

[Link para o Producer](https://github.com/pedrohft/YoutubeDataAnalytics/blob/main/Producer.py)

Código do ***Consumer***, responsável pela leitura do dado no tópico e armazenação no *datalake*:

[**Link para o Consumer**](https://github.com/pedrohft/YoutubeDataAnalytics/blob/main/Consumer.py)

**Construção Jobs:**

Códigos dos ***Jobs***, responsável pela captação do dado das *lives* na área bronze do *datalake*, transformação e disponibilização na área *silver*:

[Link para os Jobs](https://github.com/pedrohft/YoutubeDataAnalytics/tree/main/notebooks)

### 2.2.2 Experiências vivenciadas

### 

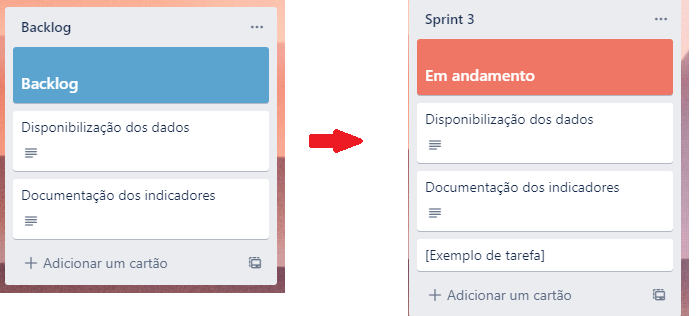
### Durante a evolução dessa etapa consegui evoluir bastante com o código do projeto, uma coisa que me chamou a atenção foi que nem tudo saiu como eu estava imaginando e eu tive que fazer alguns replanejamentos de lógica de código no desenvolvimento dos ***Jobs***, ***Consumer*** e ***Producer***. Isso me agregou como experiência que imprevistos em um projeto podem acontecer, mas que se tudo se mantenha bem planejado é passível de adaptações sem afetar o prazo da entrega. Nesta fase apliquei metodologias de orientações a objetos que aumentaram o escopo evolutivo do projeto, facilitando na replicação de código e também permitindo a criação de diversos tópicos Kafka rodando em paralelo, fazendo com que seja trafegado alta quantidade de dados, essa ideia eu tive durante o desenvolvimento e foi uma das coisas que não estava planejadas a ser desenvolvida inicialmente.

### Acredito que evolui bastante profissionalmente ao decorrer dessa sprint pois de fato enfrentei problemas e os contornei com êxito dentro do prazo da entrega, mantendo o padrão de excelência aos códigos.

## 2.3 Sprint 3

### 2.3.1 Solução

#### Evidência do planejamento:



#### Evidência da execução de cada requisito:

**Disponibilização dos Dados:**

Os dados que estão disponíveis no *datalake* do projeto foram disponibilizados no MySQL, assim como mostrado na arquitetura inicial.   
Os dados foram divididos em três tabelas iniciais para que os analistas pudessem fazer consultas e tirar as suas conclusões sobre os dados e assim fazerem demandas de mais tabelas de negócios.   
Como o MySQL é um dos bancos de dados mais disseminados em projetos de TI, muitos desses analistas provavelmente já tiveram o contato alguma outra vez com a ferramenta, o que facilita e acelera as consultas.

**Documentação dos indicadores:**

Foi feita a documentação dos indicadores das tabelas para que os analistas pudessem consultar e entender como aquele dado funciona. Essa documentação também serve como uma padronização dos indicadores para que caso seja feita alguma modificação futuramente, fique documentado as versões e para que não ocorram desavenças em relação a dado errôneo. É muito comum ocorrer modificações em tabelas e a documentação faz com que tudo siga um fluxo de alteração, sem ela pode ocorrer muitas confusões e acabar gerando dado lixo dentro delas.

#### Evidência dos resultados:

**Disponibilização dos Dados:**

Essas são as três tabelas iniciais. A tabela **Author** possui informações sobre quem mandou a mensagem no chat da *live*, essa tabela tem uma relação de um para muitos com a tabela **Message** que possui as informações das mensagens que cada pessoa digita no chat.

Texto

Descrição gerada automaticamente

Exemplo de massa de dados da tabela **Author**:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Exemplo de massa de dados da tabela **Message**:

Interface gráfica do usuário

Descrição gerada automaticamente

A tabela **Messages** é um amontado de informações cruas, é uma tabela bruta para uma análise prévia e de fácil disponibilização, ela foi criada para atender a demanda mais rápida que não precisa de muita tratativa. Ela possui tanto indicadores dos altores quanto das mensagens.

Exemplo de massa de dados da tabela **Messages**:

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente

Exemplo de análise com as tabelas **author** e **message,** onde representa a quantidade de mensagens por altor:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Documentação dos indicadores:**

A documentação dos indicadores está disponível em um arquivo de texto, no seguinte endereço: [Documentação Indicadores](https://github.com/pedrohft/YoutubeDataAnalytics/blob/main/arquitecture/doc_indicadores.docx)  
Porém não está sendo mostrado, apenas disponível para download.  
Exemplo da documentação:

Tabela

Descrição gerada automaticamente

### 2.3.2 Experiências vivenciadas

### Esta etapa desenvolvida teve o foco na disponibilização e documentação dos dados que serão consumidos pelos analistas. Durante o desenvolvimento foi percebido que essa fase é primordial para a empresa como um todo, pois, o dado é o que agrega no valor da empresa, é de lá que sai todas as análises e novas ideias. Documentar bem os indicadores faz com que amenize a confusão que é gerada ao trabalhar com os dados e aplica a padronização de criação das tabelas. Já presenciei confusões em *datalakes*, como dado sujo, colunas com valores errados, colunas faltantes ou sem valor, devido à falta de uma documentação. Como profissional, aprendi que para tudo que é feito dentro do projeto envolvendo dados ou não, é preciso seguir um padrão de documentação, pois assim facilita a disseminação do conhecimento dentro da equipe e para novos integrantes.

# 3. Considerações Finais

## 3.1 Resultados

O desafio apresentado para o projeto aplicado foi concluído usando ferramentas de alta performance e comuns dentro do ambiente de engenharia de dados, não foi usada nenhuma ferramenta que não tenha uma base sólida. Para fazer a captação dos dados e armazenar no *datalake*, foi construída uma aplicação utilizando o Kafka para que o dado pudesse ser gravado em tempo real dentro do *datalake*, mantendo sua integridade e garantia de entrega de todos os arquivos, após o dado chegar cru no formato *json*, foi feito por meio de *jobs* *spark* o particionamento dos arquivos e a segregação entre os níveis. Essa massa de dados foi disponibilizada no banco de dados MySQL e os responsáveis por fazerem essa carga são também *jobs escritos* em *python*.

Os pontos positivos dessa arquitetura que foi construída é a escalabilidade e possibilidade de aprimoração, além de ser simples e robusta. Os pontos negativos é que no estado em que se encontra a arquitetura está limitada ao ambiente local e a falta de recursos para à escalar para um ambientem em nuvem, junto com a falta de gatilhos para as atividades começarem a rodar por si próprias em um determinado horário.

Para montar o ambiente e fazer com que tudo conversasse foi uma dificuldade, foi até um limitador que fez com que não fosse possível criar um *job spark* que lesse os dados do tópico Kafka em tempo real, foi utilizado containers do Docker para subir o ambiente do banco de dados e do Kafka, já o Spark foi feito uma instalação local, isso fez com o Kafka não conversasse diretamente com o Spark, limitando a arquitetura. Para que tudo isso conversasse teria que ter sido feita a instalação somente local na máquina apontando para o mesmo *localhost,* porém a máquina utilizada para montar a arquitetura não aguentaria a instalação do ambiente e assim foi separado em contêineres.

## 3.2 Contribuições

O pipeline construído para a resolução do desafio foi construído localmente para que não houvesse custos adicionais durante o desenvolvimento, sendo que este também é de fácil escalabilidade para o ambiente de nuvem. As tecnologias utilizadas são hoje as melhores e mais utilizadas no âmbito de engenharia de dados, trazendo uma arquitetura de ponta para o projeto com tecnologias atuais, além de serem fáceis de se manusear e serem disseminadas na comunidade de TI.

Os dados foram guardados, disponibilizados e documentados seguindo os padrões da segurança da informação, podendo ampliar ainda mais essa segurança aplicando restrições dentro do banco de dados. O acesso aos dados ficou simples e fácil de se entender, o que facilita o trabalho dos analistas de dados, acelerando o processo de extração de informação e contribuição para o valor da companhia.

Toda a arquitetura projetada para resolver o desafio proposto está bem documentada e isso facilita a ampliação do projeto, facilitando o entendimento dos novos integrantes da equipe e assim ganhando tempo no decorrer do crescimento do projeto mantendo o padrão proposto.

## 3.3 Próximos passos

O próximo passo é escalar o ambiente do pipeline para a nuvem, onde o suporte será maior para a arquitetura e o escalonamento também, junto com o aumento da performance. Fazer o ambiente rodar como um todo, subir uma instância do serviço do Amazon Managed Streaming for Apache Kafka (MSK) para instalar o servidor do Kafka e subir os tópicos desenvolvidos no ambiente local. Migrar os *jobs sparks* para o AWS Glue para assim ter mais capacidade de processamento e automação. Subir uma instância do Amazon Aurora para poder migrar o banco de dados que está hoje no MySQL, pois, o Aurora possui praticamente as mesmas funcionalidades e migrar o *datalake* para o S3.

# Referências

Marotta, L. (20 de 06 de 2022). *What programming language do you need to become a data engineer?* Fonte: Quora: https://www.quora.com/What-programming-language-do-you-need-to-become-a-data-engineer