



igti

# RELATÓRIO

---

## PROJETO APLICADO

Instituto de Gestão e Tecnologia da Informação  
Relatório do Projeto Aplicado

# Disponibilização de dados analíticos e recomendações para clientes no Varejo E-commerce

Gabriel Lima Novais

Orientador(a):

Murillo Barbosa

14/01/23



GABRIEL LIMA NOVAIS

INSTITUTO DE GESTÃO E TECNOLOGIA DA INFORMAÇÃO

RELATÓRIO DO PROJETO APLICADO

# Disponibilização de dados analíticos e recomendações para clientes no Varejo E-commerce

Relatório de Projeto Aplicado  
desenvolvido para fins de conclusão do  
curso MBA em Engenharia de Dados.

Orientador (a): Murillo Barbosa

Niterói  
14/01/23

## Sumário

1. CANVAS do Projeto Aplicado	4
1.1 Desafio	5
1.1.1 Análise de Contexto	5
1.1.2 Personas	9
1.1.3 Benefícios e Justificativas	12
1.1.4 Hipóteses	14
1.2 Solução	17
1.2.1 Objetivo SMART	17
1.2.2 Premissas e Restrições	18
1.2.3 Backlog de Produto	21
2. Área de Experimentação	24
2.1 Sprint 1	24
2.1.1 Solução	24
• Evidência do planejamento:	24
• Evidência da execução de cada requisito:	27
• Evidência dos resultados:	32
2.1.2 Experiências vivenciadas	34
2.2 Sprint 2	35
2.2.1 Solução	35
• Evidência do planejamento:	35
• Evidência da execução de cada requisito:	38
• Evidência dos resultados:	45
2.2.2 Experiências vivenciadas	47
2.3 Sprint 3	48
2.3.1 Solução	48
• Evidência do planejamento:	48
• Evidência da execução de cada requisito:	50
• Evidência dos resultados:	56
2.3.2 Experiências vivenciadas	57
3. Considerações Finais	58
3.1 Resultados	58
3.2 Contribuições	58
3.3 Próximos passos	58

## 1. CANVAS do Projeto Aplicado

Segue a Figura 01 conceitual canvas do projeto aplicado, que representa todas as etapas do projeto.



Figura 01 - Canvas do Projeto Aplicado

## 1.1 Desafio

### 1.1.1 Análise de Contexto

O varejo, principalmente em países em desenvolvimento como o Brasil, é de suma importância, por ser uma fonte provedora de empregos. No Brasil corresponde cerca de 47,4% do PIB, apresentando um crescimento expressivo e sempre está na mira de empreendedores em busca de oportunidades.

O setor consiste no processo de compra de produtos em quantidades relativamente grande dos produtores atacadistas e outros fornecedores e posteriormente vendidos em quantidades menores ao consumidor final. Além disso, pode ser considerado como atividade responsável por providenciar mercadorias e serviços desejados pelos consumidores.

O varejo é definido, segundo Philip Kotler como método comercial que engloba todas as atividades relativas à venda direta de produtos ou serviços ao consumidor final. No entanto, Parente (2000), conceitua o varejo como todas as atividades que englobam o processo de venda de produtos e serviços para atender a uma necessidade pessoal do consumidor final. Las Casas (2004, p. 17) cita uma definição de varejo de acordo a American Marketing Association, onde considera o varejo como “uma unidade de negócio que compra mercadorias de fabricantes, atacadistas e outros distribuidores e vende diretamente a consumidores finais e eventualmente aos outros consumidores”.

Então qualquer empresa que forneça um produto ou serviço para o consumidor final está praticando varejo. O comércio varejista conta com processos sistematizados, a fim de atender bem às demandas dos clientes. É importante apresentar uma cadeia de suprimentos bem definida para corresponder às necessidades do mercado e demonstrar competitividade. O comércio varejista vem assumindo uma importância crescente no panorama empresarial do Brasil e do mundo.

À medida que as empresas varejistas se expandem, passam a adotar tecnologias avançadas de informação e de gestão e desempenham papel cada

vez mais importante na modernização do sistema de distribuição e da economia brasileira. Para que o setor seja bem-sucedido, é preciso estar alerta e pronto para se adaptar aos desafios de mercado.

A tecnologia da informação desempenha papel primordial nesse processo de evolução, visto que o comércio virtual (e-commerce) é a grande tendência do setor de varejo. O varejo online fornece ao consumidor informações, agilidade na entrega e apresenta descontos atraentes. Não apenas a tecnologia servirá como meio que possibilita a realização da venda, através das plataformas digitais, mas como uma forma essencial de permitir análises de planejamento estratégico, marketing, performance e muitas outras de acentuada importância para tomadas de decisão.

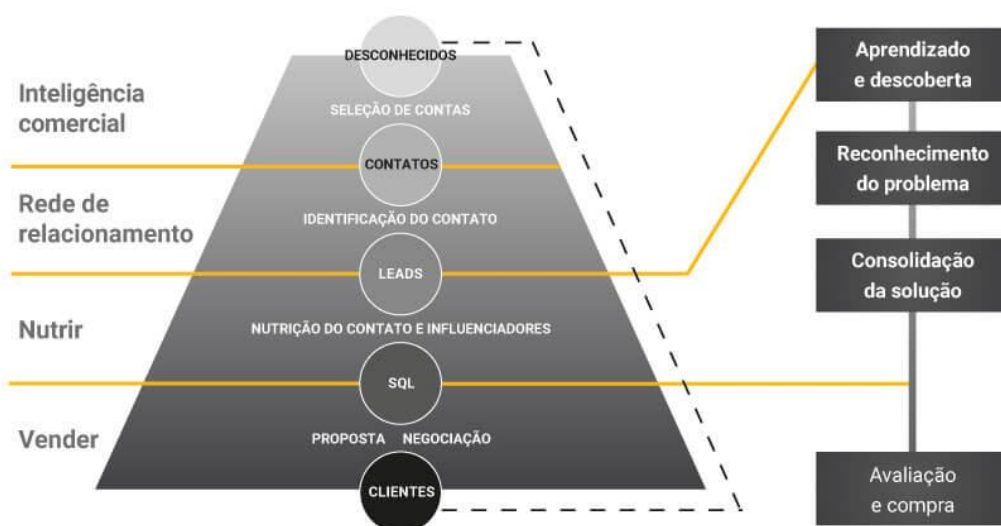


Figura 02 - Account Based Marketing

Fonte: <https://layerup.com.br/account-based-marketing/>

Segundo afirma a Associação Brasileira de Comércio Eletrônico, as vendas online correspondem a 11,6% do setor varejista do país. Contudo, toda esta evolução causa um impacto no comportamento do consumidor.

A tendência do consumidor moderno é ser mais exigente, mais direcionado por questões de tempo, mais necessitado de informação e

altamente individualista. Ações que visam a construção de um ambiente de vendas mais personalizado e com um conforto maior para o consumidor, garantem elevação de rentabilidade ao negócio.



Figura 03 - Modelo de Matriz CSD

Dessa forma, após elucidação do contexto e da exposição do problema, é possível defender cada vez mais o uso intensivo de dados. A necessidade de se construir maneiras mais eficientes de garantir uma experiência do usuário, ao mesmo tempo que se permite um ferramental adequado para acompanhar a performance das vendas, preços e visitas do site e dos desdobramentos de tais monitorias, se baseiam quase que completamente nessa hipótese.

Para estruturar as idéias mencionadas foram realizadas a matriz e a observação contida na Figura 03 acima. Adicionalmente, para realizar a análise do contexto do problema, foi utilizado o canvas abaixo, contido na Figura 04.





Figura 04 - Análise do Contexto do problema

No desenvolvimento de uma solução, um comum exercício que busca identificar o perfil daqueles que serão os usuários é a criação de personas, não necessariamente representando alguém do público alvo que saiba tudo do negócio em questão. Especificamente para este projeto, foram utilizadas as informações detalhadas nas seções anteriores para criar 3 perfis de personas. Esses perfis são de duas “pontas” completamente distintas do negócio em questão.

Enquanto a primeira e a segunda persona que são funcionários da empresa, do local onde a solução será empreendida, sendo os dois analistas de áreas diferentes, a terceira persona representa o consumidor final dos serviços apresentados. De forma resumida, listam-se essas personas:

- Analistas de BI e de Marketing dentro da empresa, na qual a solução será utilizada como um guia para tomada de decisão.
- Clientes, consumidores dos produtos vendidos pelo e-commerce, que desfrutarão de uma experiência personalizada de compra na plataforma.

#### **Persona 1**

**Nome:** Pedro

**Idade:** 30 anos

**Profissão:** Analista de BI

**Características comportamentais:** Pedro é engenheiro de produção e possui uma formação mais analítica. Possui um perfil bem pragmático, gosta de resolver problemas gerando formas de solução que funcionem no dia a dia, não precisando recorrer a complexidades desnecessárias. Ele precisa de dados para acompanhar de maneira mais eficiente as métricas de vendas e visitas da empresa e entender como esses indicadores refletem o valor gerado para companhia. Ele deseja rapidez e facilidade para gerar os reports e visualizações de dados para garantir tomadas de decisões mais concretas sobre preços e atingimento de metas.

### **Persona 2**

**Nome:** Maria

**Idade:** 28 anos

**Profissão:** Analista de Marketing

**Características comportamentais:** Maria é formada em administração e possui pós-graduação em marketing digital. Ela possui um perfil mais criativo, com boas análises sobre tendências de mercado e conhecimentos sobre ações de marketing e segmentação de audiências, o que lhe confere resultados bons nas suas campanhas de marketing. Ela possui uma dificuldade de entender como melhorar os produtos recomendados para seus clientes e como segmentá-los para melhorar suas campanhas de push (envio de mensagens e acionamento de clientes).

### **Persona 3**

**Nome:** Sérgio

**Idade:** 54 anos

**Profissão:** Carpinteiro

**Características comportamentais:** Sérgio possui curso técnico em carpintaria, e no seu negócio, prioriza pela qualidade do acabamento dos seus móveis. Entretanto precisa de ferramentas e outros insumos. Como possui muitas demandas, não pode perder tempo procurando esses recursos e para isso realiza compras online. Além disso, gosta de pescar e fazer trilhas, como forma de lazer, e compra todas as suas iscas e materiais pela internet. Se ele tiver o que precisa e gosta, em um só lugar, com certeza compraria e otimizaria seu tempo tão valioso.

As personas listadas foram construídas após feito o mapa da empatia mostrado na Figura 05 . Note que nesse mapa as dores mostradas são compartilhadas pelas 3 personas e a solução a ser proposta visa sanar essa dor. De modo que, é criada a seguinte relação lógica: Uma vez disponibilizado dados para consulta sobre indicadores e métricas de avaliação do negócio e ao mesmo tempo recomendações de produtos, os usuários terão uma experiência personalizada de consumo, aumentando a rentabilização que poderá ser

acompanhada nos dados da outra tabela, permitindo que os analistas tomem decisões estratégicas para aproveitar não apenas essa nova maneira de exposição de conteúdo ou de seleção de portfólio de produtos a serem vendidos, como aperfeiçoar as campanhas de marketing e melhor interpretar as audiências e seus perfis.

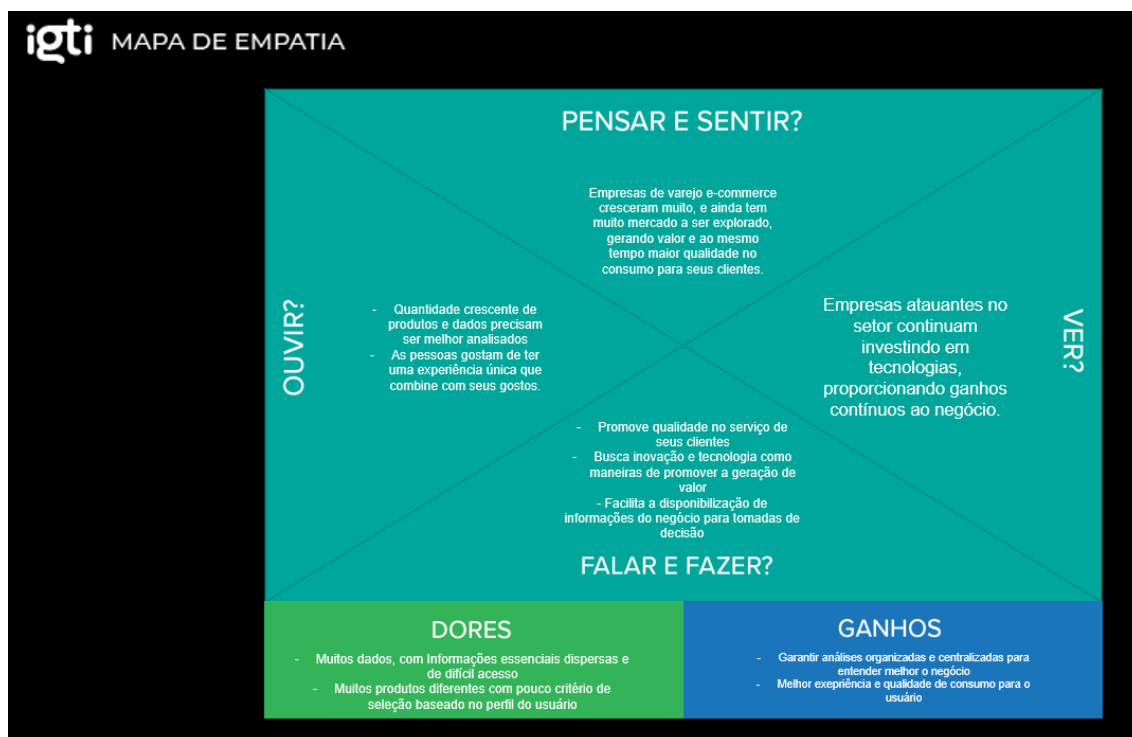


Figura 05 - Mapa de Empatia

### 1.1.3 Benefícios e Justificativas

Conforme apresentado nas seções anteriores, o problema está bem delineado, ou seja, conforme o comércio varejista se expandiu em especial por conta de fatores externos, como a pandemia e evolução tecnológica, o volume de dados, produtos e serviços se expandiram também. Para que seja possível melhorar o acompanhamento e avaliação de métricas e operações com a finalidade de proporcionar tomadas de decisões assertivas, é necessário intensificar o uso de tecnologias relacionadas a big data, processamento e armazenamento de dados em cloud e afins.

O objetivo do projeto é o de proporcionar o tratamento de dados, armazenamento de dados processados e disponibilização de dados para duas finalidades distintas: uma focada nas análises de suporte de tomada de decisão, e outra com a finalidade de melhorar a personalização da experiência do usuário.

Os benefícios auferidos dessa empreitada podem ser observados nas facilidades de obtenção de dados, que estarão disponibilizados em formato de tabela SQL, contendo métricas de avaliação, que inclusive dada a natureza da infraestrutura aplicada, permite expansão de funcionalidades e adição de colunas. Além desses benefícios, existirão dados também disponibilizados em bases que permitem consultas em SQL, que fornecem a reelações de recomendações de produtos e usuários que facilitarão o uso de vitrines, ações de acionamento de clientes, segmentação de audiências e outras finalidades (inclusive mensuradas pelos dados da outra tabela)

O Blue print (Figura 06), divide a experiência em ação, funcionalidade, interação e mensagem. Deve-se entender a ação da persona ao buscar uma solução da sua dor. Para cada ação, a persona verá funcionalidades para as soluções possíveis, como interagir com as possíveis soluções e qual mensagem ela absorve durante aquela ação.



Figura 06 - Blueprint

A ferramenta da proposição de valor (Figura 07) detalha como que a persona enxerga o valor na solução a ser desenvolvida diante das dores que enfrenta. A reflexão de como a persona irá extrair valor ajuda a tornar a proposta mais objetiva. Como resultado, obteve-se que a solução mais adequada está direcionada em fornecer uma fonte de dados que auxilia a tomada de decisão.



Figura 07 - Explicação de Proposição de Valor

### 1.1.4 Hipóteses

As hipóteses seguem expostas na Figura 08, nela as observações feitas pelas personas em uma coluna e na coluna ao lado as hipóteses criadas para desenvolver o produto. Com essa informação foi possível estabelecer especificações mais direcionadas e detalhadas que eventualmente serão consideradas no processo macro de desenvolvimento do projeto.

	Observações	Hipóteses
1	Pandemia do Covid-19 proporcionou uma queda no setor, mas favoreceu as vendas por meio dos canais eletrônicos (e-commerce).	Maiores quantidades de informações navegacionais e transacionais foram e continuam sendo produzidas pelas empresas de varejo. É possível utilizar essa abundância de informações para melhoria do negócio
2	Evolução do comportamento do consumidor que possui maiores exigências, gosta de escolher o conteúdo que consome, é autoral e precisa sanar suas necessidades de forma rápida e precisa. Grande migração para meios eletrônicos e redução de compras por meios físicos.	É possível conquistar o cliente e fidelizá-lo por meio de personalização da experiência, promoções diferenciadas e o entendimento de suas necessidades de maneira mais específica. Para isso precisa-se de acompanhamento e facilidade na monitoria das ações.
3	Avanço de tecnologias relacionadas ao Big Data e IA, proporcionando maior coleta de informações, decisões automatizadas, decisões inteligentes e recomendações eficientes	É possível com ferramentas de armazenamento e processamento de dados de Cloud obter recursos para garantir que as melhores práticas e técnicas estejam ao dispor das soluções que visam melhorar as tomadas de decisão e a experiência do usuário na plataforma de venda.

Figura 08 - Matriz de observações para hipóteses

Verifica-se que as observações da matriz de observações para hipóteses direcionam o projeto na busca por formas de auxiliar tomada de decisões de negócios por meio de dados e as tecnologias associadas ao uso intensivo de dados.

Para elucidar as ideias que originaram a solução do projeto final usou-se um método de priorização de ideias. A matriz de priorização de ideias trata-se de uma ferramenta simples na qual costuma-se inserir as ideias validadas na vertical e os critérios de avaliação (de acordo com as principais necessidades identificadas) na horizontal.

Uma vez construída a matriz, a análise indica a ideia mais provável de sucesso e que poderá trazer maior resultado efetivo para o projeto. Dessa maneira ela serve para fornecer uma melhor visão das ideias e apoiar o processo de tomada de decisão. As pontuações atribuídas na matriz estão de acordo com a tabela na Figura 09 abaixo.

Escala	B - Benefícios	A - Abrangência	S - Satisfação	I - Investimentos	C - Cliente	O - Operacionalidade
5	De vital importância	Total (de 70 a 100%)	Muito grande	Pouquíssimo investimento	Nenhum impacto	Muito fácil
4	Significativo	Muito grande (de 40 a 70%)	Grande	Algum investimento	Impacto pequeno	Fácil
3	Razoável	Razoável (de 20 a 40%)	Média	Médio investimento	Médio impacto	Média facilidade
2	Poucos benefícios	Pequena (de 5 a 20%)	Pequena	Alto investimento	Impacto grande	Difícil
1	Algum benefício	Muito pequena	Quase não é notada	Altíssimo investimento	Impacto muito grande no cliente	Muito difícil

Figura 09 - Matriz da priorização de ideias.

Fonte: Apostila do MBA em Engenharia de Dados IGTI.

Usou-se a classificação conhecida como BASICO para se fazer a priorização de ideias. A sigla nos remete aos seguintes pontos, que devem ser pontuados:

- B (Benefícios): quais são os benefícios para a organização caso a solução seja adotada?
- A (Abrangência): quantas pessoas (clientes internos e externos) serão beneficiadas com essa decisão?
- S (Satisfação): qual é a satisfação das pessoas com a solução a ser adotada?
- I (Investimento): qual será o investimento necessário para a aplicação da solução?
- C (Cliente): Qual o impacto que o cliente sofrerá com a mudança?
- O (Operações): O quão complexa é a implantação da solução?



	Ideia	B	A	S	I	C	O	Total
1	Disponibilizar dados do Olist como simulação dos dados internos de uma empresa de varejo em duas tabelas em SQL separadas por finalidade: uma para análise de métricas e outra com recomendações por usuário, de acordo com os produtos existentes.	5	4	3	3	4	3	22
2	Criar um Aplicativo que esteja integrado com uma Interface amigável para selecionar métricas e ajustar ações de marketing automaticamente	4	4	5	1	4	2	20
3	Integrar dados de redes sociais e do IBGE para que com duas fontes a mais, disponibilizemos os dados em SQL e com visualizações das regiões num mapa sobre os municípios mais rentáveis	3	3	4	2	4	2	18

Figura 10 - Matriz da priorização de ideias.

Antes para desenvolver as ideias da matriz acima na Figura 10, foi realizado um Brainstorm levantando os aspectos que envolvem a solução final, algumas de suas características e as fontes de informação. A Figura 11 mostra um conjunto de ideias separadas por finalidade e contexto. Nessa etapa considera-se que soluções coletivas trazem resultados melhores e mais criativos do que ações individuais e isoladas.

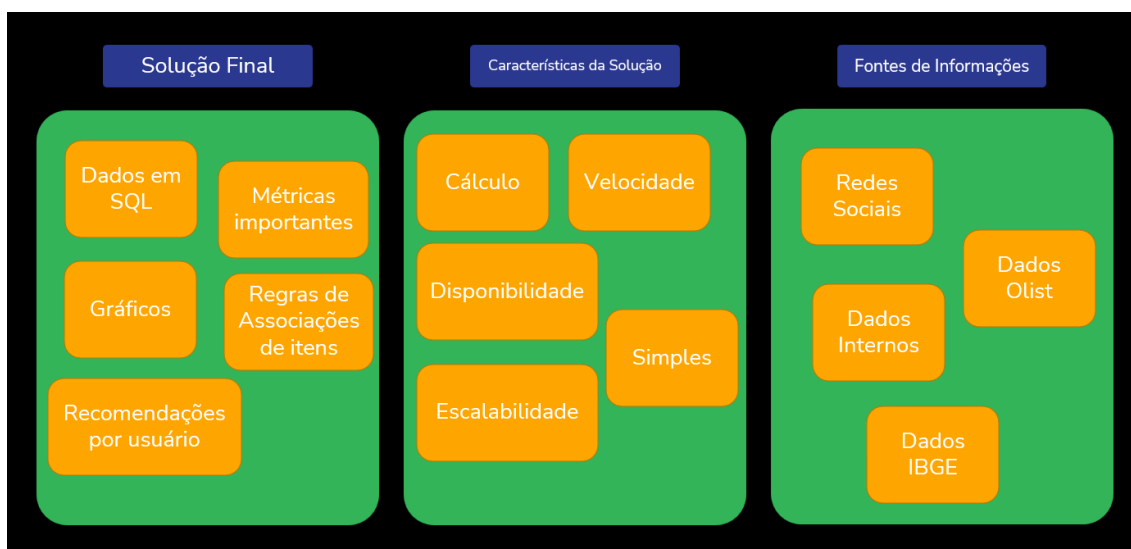


Figura 11 - Brainstorm

## 1.2 Solução

### 1.2.1 Objetivo SMART

*“Objetivou-se fornecer dados tratados e com valor adicionado que possa estar acessível por um DataWarehouse disponível em um serviço de Cloud permitindo extrair análises de duas naturezas distintas.*

*A primeira natureza é um acompanhamento das métricas de desempenho do negócio. E a segunda é disponibilizar para clientes do Varejo E-commerce recomendações personalizadas. O usuário terá acesso a pelo menos uma plataforma para dar entrada a rotinas SQL para extrair informação dos dados. A maior parte da infraestrutura será construída em forma de código e estará em nuvem. O projeto deve ser concluído até Março de 2023.”*

### 1.2.2 Premissas e Restrições

As premissas e restrições do projeto são importantes, uma vez que conhecendo os impactos gerados caso algo ocorra fora do esperado é possível buscar ações efetivas para mitigar os riscos. Dessa forma criou-se uma Matriz de Riscos, conforme apresentado na Figura 11. Os riscos identificados são os seguintes:

- Preços, quantidades vendidas e visitas difíceis de agrupar: significa dados muito separados e dispersos. Os dados que devem idealmente ser utilizados são dados internos, mas para fins de generalização da solução optou-se por utilizar dados do Olist que representam com louvor os dados comumente encontrados nas empresas de varejo e-commerce. O impacto potencial seria a redução de velocidade de consulta.
- Volume muito grande de produtos: significa que caso haja uma variedade muito grande de produtos, diversidade de portfólio elevada, os modelos que calculam as regras de associação entre os dados de compra ou visitas tendem a apresentar resultados de qualidade não tão bons.
- Custo elevado de infraestrutura: refere-se ao custo total necessário para conseguir implementar a solução na cloud da AWS. Esse fator de custo se torna um risco em especial pela necessidade de processamento computacional pelos clusters kubernetes que utilizam máquinas EC2. Uma boa medida preventiva adotada seria a utilização de máquinas spot ao invés de máquina on demand.

	Riscos Identificados	Impacto Potencial	Ações Preventivas	Ações Corretivas
1	Preços, quantidades vendidas e visitas difíceis de agrupar	Reduzir velocidade de consulta e agrupamento dos dados	Estudar e selecionar corretamente as melhores bases e tabelas	Aplicar uma infraestrutura que facilite o agrupamento de dados de diferentes fontes de maneira eficiente
2	Volume muito grande de produtos	Dificultar processamento e reduzir a qualidade das recomendações	Calcular quantidade média de produtos vistos por usuários	Aplicar corte de produtos com base na sua rentabilidade e/ou popularidade
3	Custo elevado de Infraestrutura (máquinas caras, processamento via cluster)	Inviabilizar cálculo de recomendações e/ou agrupamentos de dados	Realizar otimização de custos, preocupando-se com processamento e armazenamento razoável	Manter grande parte do projeto com infraestrutura em formato de código (IaC), facilitando a destruição e construção de recursos

Figura 11 - Matriz de Risco

A arquitetura construída para a solução, destaca-se na Figura 12 abaixo. Nela pode-se verificar que os dados serão distribuídos em 3 buckets no S3: um para ingestão, outro para processamento e outro para consumo.

Dessa forma a estrutura de processamento formada pelo cluster Kubernetes consegue organizar os inputs e outputs e garante a transferência desses dados de maneira rápida e fácil através do AWS Glue Crawler para o serviço de DataWarehouse da AWS, o Athena, local onde os analistas poderão realizar as consultas necessárias e eventuais diligências de ações para rentabilização.

A forma de construção da parte de buckets e serviços do Glue Crawler serão efetivadas por meio de estratégia IaC, conforme ferramental do Terraform. Todos os códigos, comandos, arquivos e instruções necessárias serão disponibilizadas no GitHub, conforme boas práticas de versionamento, no link a seguir: [https://github.com/NovaisGabriel/MBA\\_project](https://github.com/NovaisGabriel/MBA_project)

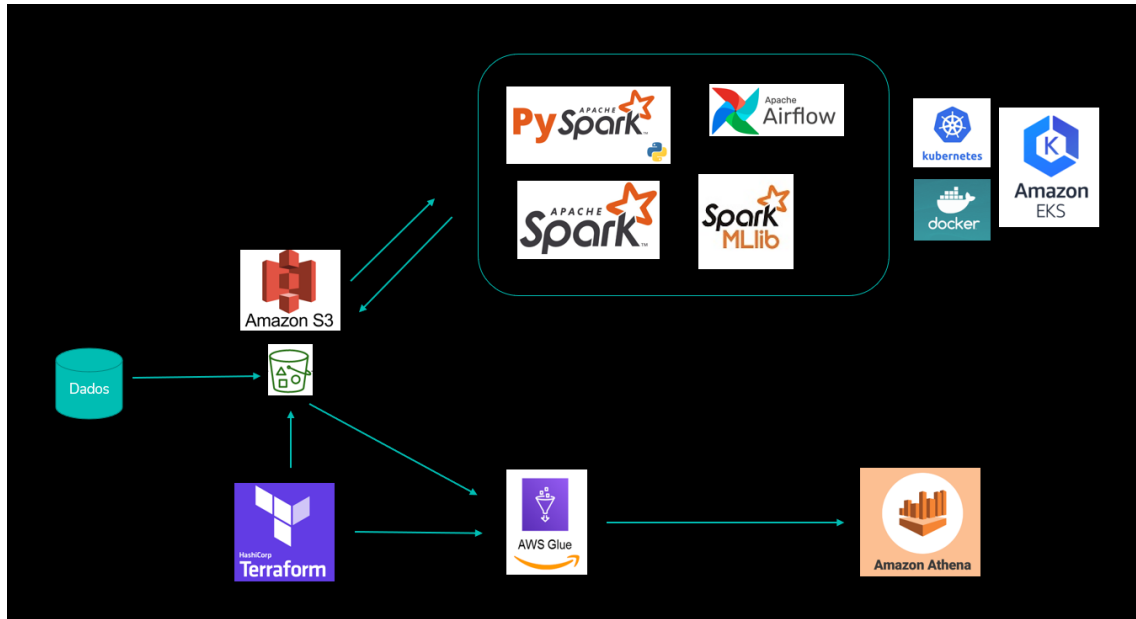


Figura 12 - Arquitetura na AWS

### 1.2.3 Backlog de Produto

O backlog compreende uma lista com todas as tarefas a serem desenvolvidas. As tarefas são mensuradas com prioridades, indivíduos que executarão a tarefa e instruções específicas dos requisitos necessários para que ela seja entendida como concluída. No backlog desse projeto, primeiramente, foram listados os requisitos, e conforme na Figura 13, são descritos como cada requisito será coberto ao longo dos sprints.

Na sprint 3, com sua conclusão, espera-se que a solução acompanhe um MVP, que esteja pronto e possa ser usada pelos usuários definidos. Alguns exemplos e testes serão executados para direcionar e simular a maneira como as personas do projeto extrairão o valor da solução.

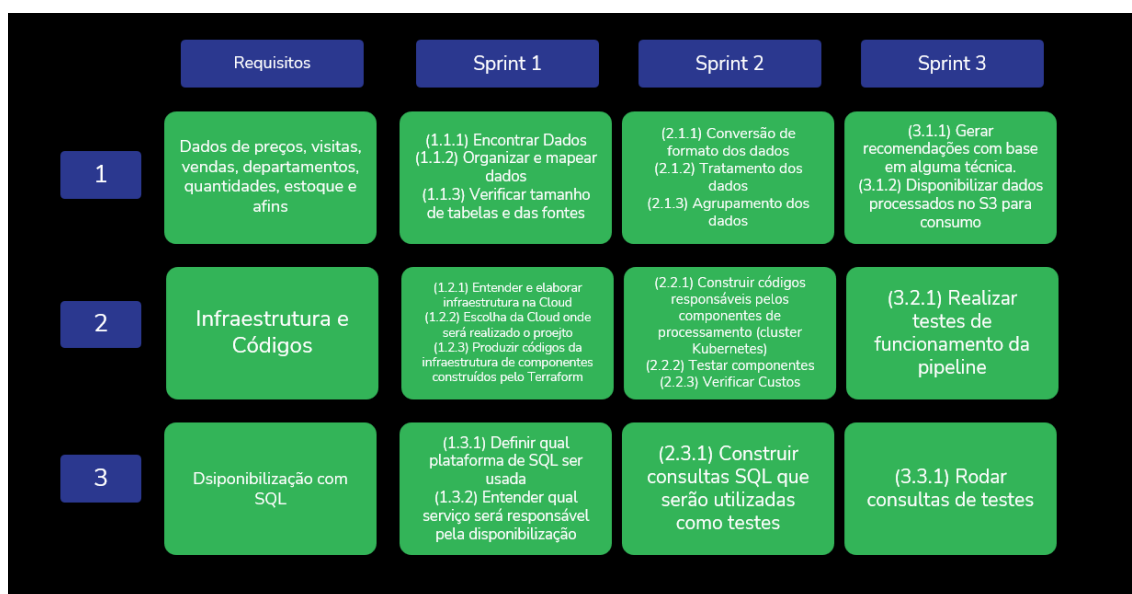


Figura 13 - Requisitos separados por sprints

A materialização do backlog descrito na Figura 13 foi registrado em ferramenta de planejamento Trello <sup>1</sup>. As atividades delineadas na figura acima foram organizadas nos quadros do Trello, descritos como na Figura 14. Conforme forem executadas as tarefas em andamento das Sprints do projeto, as atividades destacadas do backlog serão dadas como concluídas.

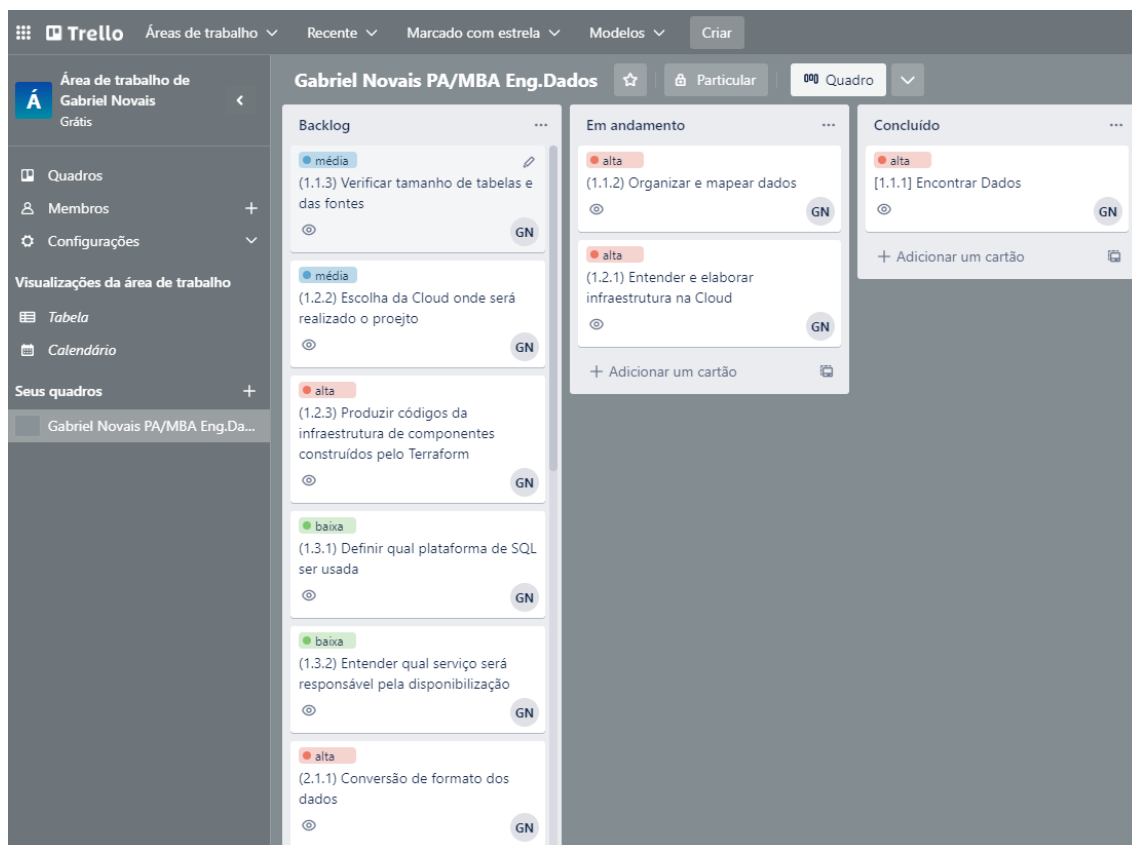


Figura 14 - Sprints, tarefas e tags no Trello

As tarefas estão numeradas conforme código, descrevendo, primeiro a Sprint, depois o requisito e terceiro a etapa. Elas são as que se seguem:

- 1.1.1.1) Encontrar Dados
- 1.1.1.2) Organizar e mapear dados
- 1.1.1.3) Verificar tamanho de tabelas e das fontes

<sup>1</sup> <https://trello.com/b/U8By4GoR/gabriel-novais-pa-mba-engdados>

- 1.2.1) Entender e elaborar infraestrutura na Cloud
- 1.2.2) Escolha da Cloud onde será realizado o projeto
- 1.2.3) Produzir códigos da infraestrutura de componentes construídos pelo Terraform
- 1.3.1) Definir qual plataforma de SQL ser usada
- 1.3.2) Entender qual serviço será responsável pela disponibilização
- 2.1.1) Conversão de formato dos dados
- 2.1.2) Tratamento dos dados
- 2.1.3) Agrupamento dos dados
- 2.2.1) Construir códigos responsáveis pelos componentes de processamento (cluster Kubernetes)
- 2.2.2) Testar componentes
- 2.2.3) Verificar Custos
- 2.3.1) Construir consultas SQL que serão utilizadas como testes
- 3.1.1) Gerar recomendações com base em alguma técnica.
- 3.1.2) Disponibilizar dados processados no S3 para consumo
- 3.2.1) Realizar testes de funcionamento da pipeline
- 3.3.1) Rodar consultas de testes



## 2. Área de Experimentação

A seção a seguir possui o objetivo de apresentar as evidências do planejamento dos requisitos selecionados do Backlog de Produto, além de mostrar a maneira como eles foram desenvolvidos e registrar os resultados alcançados. Dessa maneira é importante expor a execução e a validação dos experimentos relacionados ao desenvolvimento da solução, testando o caminho certo ou se algo precisa ser pivotado.

### 2.1 Sprint 1

A primeira Sprint do projeto possui como principal objetivo delinear as arquiteturas e infraestrutura necessárias, além de procurar entender e elencar os objetos necessários para as atividades descritas na Sprint 2. Os dados utilizados devem também ser procurados e encontrados para a sua eventual utilização.

#### 2.1.1 Solução

Após a elucidação dos objetivos descritos para a Sprint 1, será realizado um conjunto de evidências e descrições mais específicas das tarefas realizadas ou em progresso dessa Sprint.

- Evidência do planejamento:

##### 1.1.1) Encontrar Dados:

Procurar dados que possuam informações sobre e-commerce que possam simular um ambiente típico enfrentado pelas empresas varejistas.

##### 1.1.2) Organizar e mapear dados:

Entender através de um olhar mais aprofundado sobre os dados, as suas relações e dependências e mapear essas ligações.

### *1.1.3) Verificar tamanho de tabelas e das fontes:*

No caso verificar aspectos mais quantitativos e menos qualitativos sobre a base de dados, entendendo qual seria o tamanho deles e quais tipos de dados existem naquela base.

### *1.2.1) Entender e elaborar infraestrutura na Cloud:*

Desenhar e pensar no tipo de arquitetura que seria ideal para resolver o problema escolhido. Entender quais produtos podem auxiliar no fluxo da solução.

### *1.2.2) Escolha da Cloud onde será realizado o projeto:*

Etapa muito relacionada à tarefa 1.2.1, pois a escolha da Cloud está muito relacionada aos tipos de produtos que serão utilizados e como eles facilitam o desenvolvimento e produção da solução.

### *1.2.3) Produzir códigos da infraestrutura de componentes construídos pelo Terraform:*

Produzir as etapas iniciais da solução em código Terraform para providenciar os serviços básicos relacionados à solução construída.

### *1.3.1) Definir qual plataforma de SQL ser usada:*

Etapa muito ligada aos pontos 1.2.1 e 1.2.2, nos quais definem os serviços e cloud utilizada. Nessa etapa o que se procura é definir qual o local e o serviço para que os dados possam ficar eventualmente disponíveis para consulta SQL.

### *1.3.2) Entender qual serviço será responsável pela disponibilização:*

Etapa dedicada para entender o funcionamento da plataforma escolhida no ponto 1.3.1, no qual o que se procura é entender como ela facilita a consulta e como pode ser utilizada por analistas.

Para mostrar o progresso da Sprint segue a Figura 15 do Trello sobre o planejamento realizado.

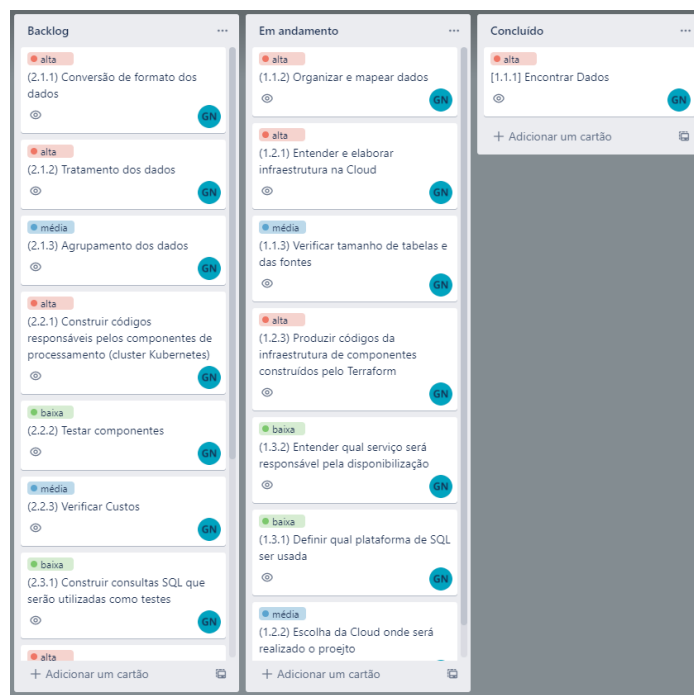


Figura 15 - Progresso da Sprint no Trello.

Após realização dos pontos destacados acima pode-se colocar cada card na coluna da direita, coluna destinada às tarefas concluídas, conforme observado na Figura 16.

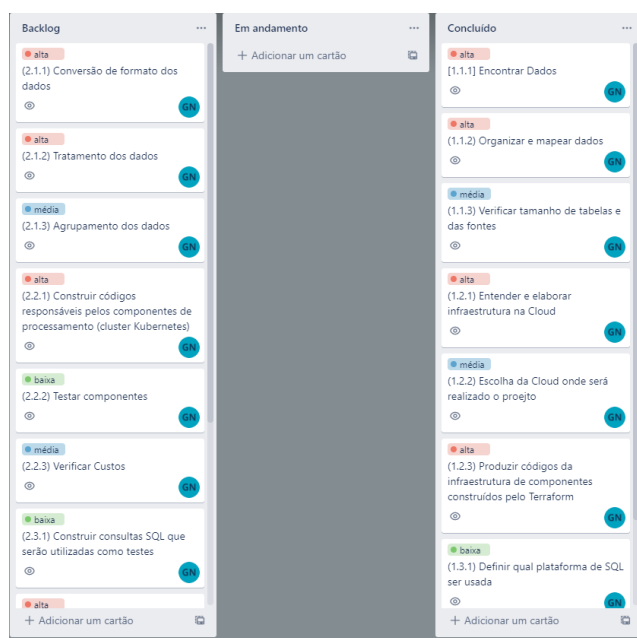


Figura 16 - Tarefas da Sprint 1 concluídas.

- Evidência da execução de cada requisito:

Após a descrição do planejamento e dos itens é possível destacar as evidências dos requisitos realizados.

### 1.1.1) Encontrar Dados

Procurar dados que possuam informações sobre e-commerce que possam simular um ambiente típico enfrentado pelas empresas varejistas. No caso do projeto foi escolhido dados disponíveis publicamente do Olist. Na Figura 17 é possível observar a fonte de download dos dados citados.

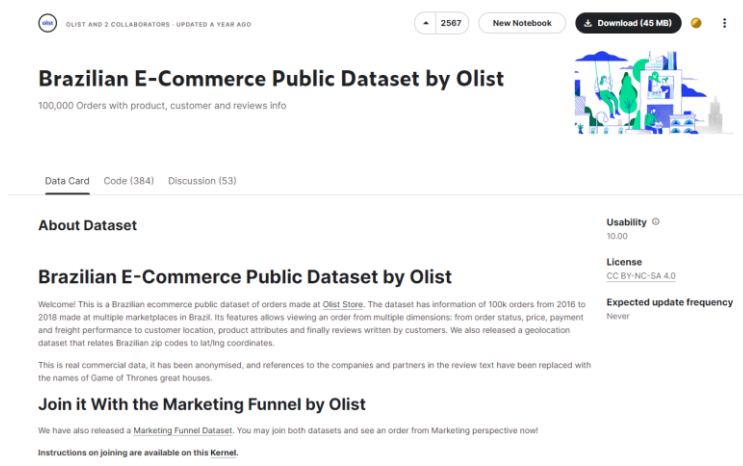


Figura 17 - Página de Download dos dados

Link da fonte de dados:

<https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce>

### 1.1.2) Organizar e mapear os dados

O objetivo nesta etapa é entender através de um olhar mais aprofundado sobre os dados, as suas relações e dependências e mapear essas ligações. Com existem diversas tabelas é possível que apenas algumas sejam utilizadas. Na Figura 18 é possível verificar as relações que existem entre as tabelas.

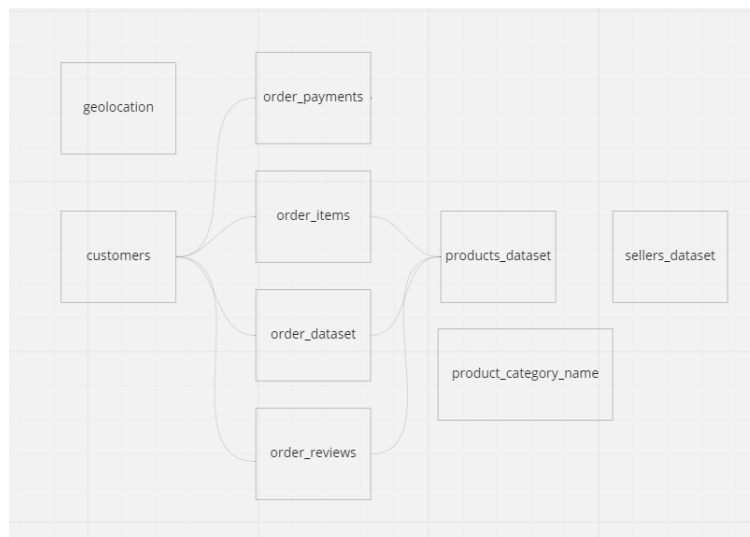


Figura 18 - Relação entre as tabelas

### 1.1.3) Verificar tamanho de tabelas e das fontes

Nesta etapa o objetivo foi verificar aspectos mais quantitativos e menos qualitativos sobre a base de dados, entendendo qual seria o tamanho deles e quais tipos de dados existem naquela base. Para isso construiu-se o código em python para analisar formatos e tamanhos. Um exemplo de análise pode ser verificada na Figura 19.

#### Customers

```
df = pd.read_csv("C:\\Users\\Gabriel\\Desktop\\backup\\Repositorios\\MBA_proje
print(df.head())
df.info()
```

```
<
      customer_id      customer_unique_id \
0  06b8999e2fba1a1fbc88172c00ba8bc7  861eff4711a542e4b93843c6dd7febb0
1  18955e83d337fd6b2def6b18a428ac77  290c77bc529b7ac935b93aa66c333dc3
2  4e7b3e0288586ebd00712fdd0374a03  060e732b5b29e8181a18229c7b0b2b5e
3  b2b6027bc5c5109e529d4dc6358b12c3  259dac757896d24d7702b9acbbff3f3c
4  4f2d8ab171c80ec8364f7c12e35b23ad  345ecd01c38d18a9036ed96c73b8d066

      customer_zip_code_prefix  customer_city customer_state
0                14409          franca                SP
1                9790  sao bernardo do campo                SP
2                1151          sao paulo                SP
3                8775      mogi das cruzeiras                SP
4                13056          campinas                SP
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99441 entries, 0 to 99440
Data columns (total 5 columns):
#   Column                Non-Null Count  Dtype
---  -
0   customer_id            99441 non-null  object
1   customer_unique_id     99441 non-null  object
2   customer_zip_code_prefix 99441 non-null  int64
3   customer_city           99441 non-null  object
4   customer_state          99441 non-null  object
dtypes: int64(1), object(4)
memory usage: 3.8+ MB
```

Figura 19 - Exemplo de tamanhos e qualidade dos dados

### 1.2.2) Escolha da Cloud onde será realizado o projeto

A cloud escolhida pode ser ilustrada na Figura 20, no caso a AWS, com alguns de seus principais produtos e a tela home de entrada na conta dessa cloud. Nesta etapa o importante foi entender quais produtos podem auxiliar no fluxo da solução.

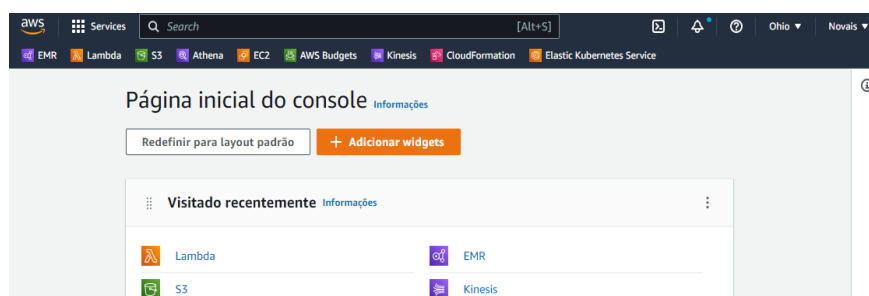


Figura 20 - Home da cloud AWS

### 1.2.1) Entender e elaborar infraestrutura na Cloud

Nesta etapa o importante foi desenhar e pensar no tipo de arquitetura que seria ideal para resolver o problema escolhido. Etapa muito relacionada à tarefa 1.2.2, pois a escolha da Cloud está muito relacionada aos tipos de produtos que serão utilizados e como eles facilitam o desenvolvimento e produção da solução. Na Figura 21 é possível verificar o desenho da arquitetura elaborado.

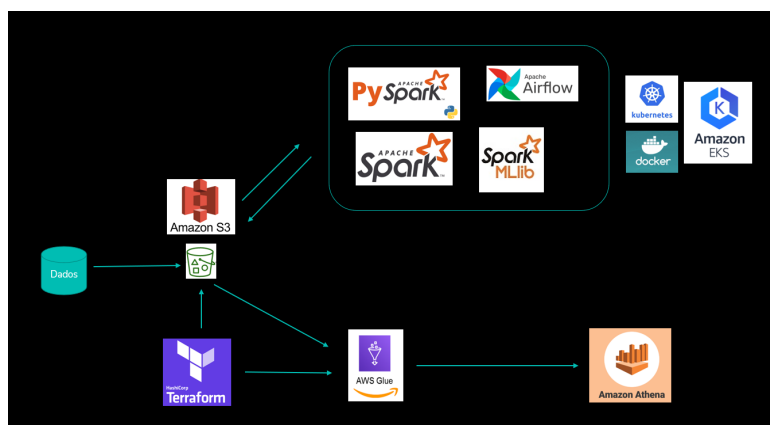


Figura 21 - Arquitetura da solução

### 1.3.1) Definir qual plataforma de SQL ser usada

Nessa etapa o que se procura é definir qual o local e o serviço para que os dados possam ficar eventualmente disponíveis para consulta SQL. A escolha dessa estrutura na cloud ocorreu pois a mesma possui imenso destaque devido suas funcionalidades e possibilidades de elaboração de consultas. Na Figura 22 é possível verificar a home desse serviço na cloud.

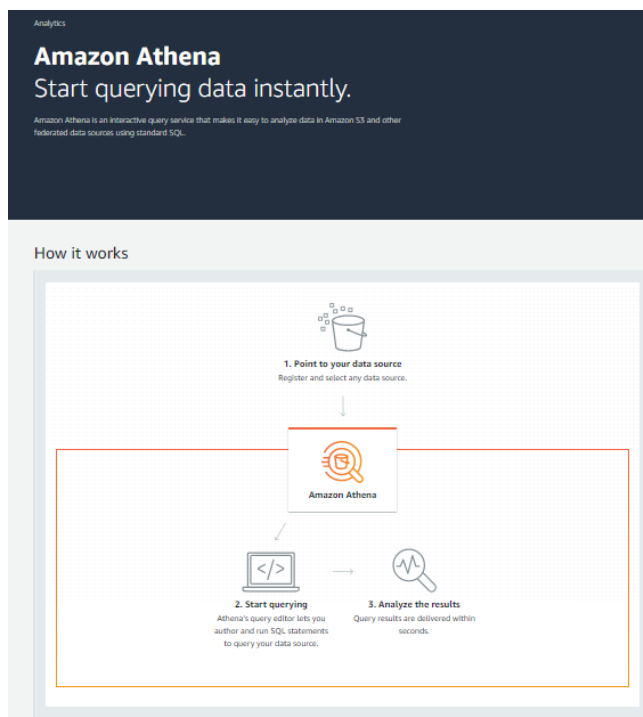


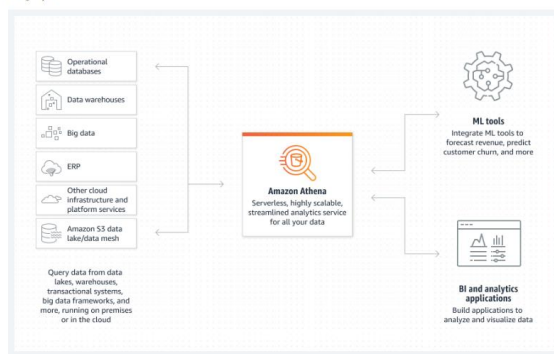
Figura 22 - Home do serviço de consulta (Athena)

### 1.3.2) Entender qual serviço será responsável pela disponibilização

Etapa dedicada para entender o funcionamento da plataforma escolhida no ponto 1.3.1, no qual o que se procura é entender como ela facilita a consulta e como pode ser utilizada por analistas. Na Figura 23 é possível verificar a documentação online disponível para o entendimento do serviço de consulta, denominado pela AWS de Athena.

### Como funciona?

O Amazon Athena é um serviço de análise interativo e sem servidor criado em frameworks de código aberto, com suporte a formatos de tabela e arquivo abertos. O Athena fornece uma maneira simplificada e flexível de analisar petabytes de dados onde eles residem. Analise dados ou crie aplicações a partir de um data lake do Amazon Simple Storage Service (S3) e mais de 25 fontes de dados, incluindo fontes de dados on-premises ou outros sistemas em nuvem usando SQL, ou Python. O Athena é construído com mecanismos Trino e Presto de código aberto e frameworks Apache Spark, sem necessidade de provisionamento ou configuração.



Clique para ampliar

Casos de uso

Execute consultas  
federadas

Prepare dados para  
modelos de ML

Crie mecanismos  
distribuídos de

Análise dados do  
Google Analytics

Figura 23 - Documentação Athena

### 1.2.3) Produzir códigos da infraestrutura de componentes construídos pelo Terraform

Nesta etapa o objetivo foi o de produzir as etapas iniciais da solução em código Terraform para providenciar os serviços básicos relacionados à solução construída. Os códigos necessários para construir os buckets (landing, processing e delivery), além das policies, crawlers e demais serviços necessários podem ser vistos nas imagens abaixo. Todos os códigos estão disponíveis no github. Na Figura 24 é possível verificar um exemplo de código em Terraform utilizado.

```

1 resource "aws_iam_role" "glue_role" {
2   name = "Role_GlueCrawler"
3   path = "/"
4   description = "Provides write permissions to CloudWatch Logs and S3 Full Access"
5   assume_role_policy = file("../permissions/Role_GlueCrawler.json")
6 }
7
8 resource "aws_iam_policy" "glue_policy" {
9   name = "Policy_GlueCrawler"
10  path = "/"
11  description = "Provides write permissions to CloudWatch Logs and S3 Full Access"
12  policy = file("../permissions/Policy_GlueCrawler.json")
13 }
14
15 resource "aws_iam_role_policy_attachment" "glue_attach" {
16   role = aws_iam_role.glue_role.name
17   policy_arn = aws_iam_policy.glue_policy.arn
18 }
19
20 resource "aws_iam_role" "lambda_decompress" {
21   name = "Role_Lambda_decompress_S3"
22   path = "/"
23   description = "Provides write permissions to CloudWatch Logs and S3 Full Access"
24   assume_role_policy = file("../permissions/Role_Lambda_decompress_S3.json")
25 }
26
27 resource "aws_iam_policy" "lambda_decompress" {

```

Figura 24 - Exemplo de código Terraform



- Evidência dos resultados:

Através do que foi realizado nas tarefas acima, é possível verificar que os componentes e serviços desejados foram construídos com os códigos e organizações elaboradas, de forma que os resultados são evidenciados segundo as Figuras 25, 26 e 27. Na Figura 25 pode-se ver as estruturas dos buckets, divididos em zones de dados crus, processados e prontos para o consumo, traduzidos pelas palavras, respectivamente, landing, processing e delivery.

Amazon S3 > Buckets

► Snapshot da conta  
O Storage Lens fornece visibilidade sobre o uso e as tendências de atividades. Saiba mais

**Buckets (6)** Info  
Os buckets são contêineres para dados armazenados no S3. Saiba mais

Q Encontrar buckets por nome

	Nome	Região da AWS
<input type="radio"/>	delivery-zone-715036709715	Leste dos EUA (Ohio) us-east-2
<input type="radio"/>	landing-zone-715036709715	Leste dos EUA (Ohio) us-east-2
<input type="radio"/>	processing-zone-715036709715	Leste dos EUA (Ohio) us-east-2
<input type="radio"/>	temp-functions-rony-715036709715	Leste dos EUA (Ohio) us-east-2
<input type="radio"/>	terraform-logs-gabriel	Leste dos EUA (Ohio) us-east-2
<input type="radio"/>	testekuberneteslogs	Leste dos EUA (Ohio) us-east-2

Figura 25 - Estrutura de buckets

Na Figura 26 é possível verificar a atividade dos crawlers construídos com a finalidade de coletar os dados dos buckets e possibilitar a transferência para o Athena.

**Crawlers**  
A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates

**Crawlers (3)** Info  
View and manage all available crawlers.

Q Filter crawlers

<input type="checkbox"/>	Name	State	Schedule	Last run
<input type="checkbox"/>	dl_delivery_zone_crawler	Ready		-
<input type="checkbox"/>	dl_landing_zone_crawler	Ready		Succeeded
<input type="checkbox"/>	dl_processing_zone_crawler	Ready		-

Figura 26 - Crawlers ativos

E na Figura 27 é possível observar uma query de teste no conjunto de dados da tabela costumers na landing zone, obtidos via o crawler construído anteriormente.

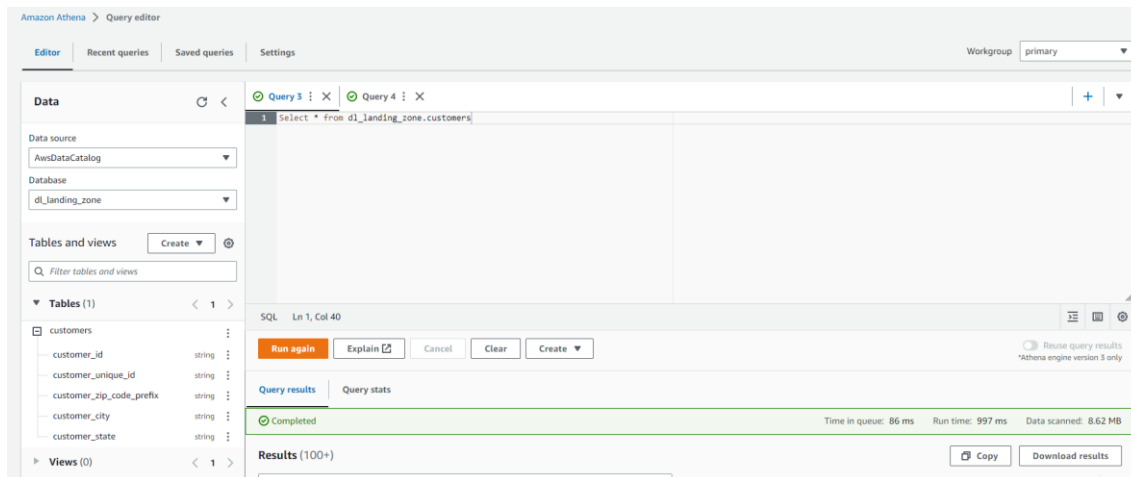


Figura 27 - Consulta de teste na landing zone

## 2.1.2 Experiências vivenciadas

Ao longo da Sprint, algumas tarefas possuíram um nível de dificuldade menor de forma que a conclusão dessas foi relativamente rápida, quando comparadas às tarefas mais difíceis relativas a construção dos códigos de Terraform, responsáveis por disponibilizar os serviços necessários para arquitetura almejada. Existem alguns pontos a serem destacados:

- Quantidade de dados: Os dados coletados em sua maioria refletem a natureza dos dados de uma empresa de varejo e-commerce, mas apenas algumas tabelas devem ser utilizadas nesse projeto. Será interessante na próxima sprint verificar quais dessas tabelas de fato serão utilizadas.
- Qualidade de dados: Talvez valha a pena pensar se na próxima sprint pode ser válido acrescentar uma tarefa de limpeza de dados ou algo que possa garantir a qualidade dos dados em questão. Ou seja, na tarefa futura de “tratamento dos dados” talvez seja interessante dividir em formato e limpeza.

Sobre os códigos construídos em Terraform, vale salientar que muito do que foi construído, se deve em especial pela utilização do pacote de construção de infraestrutura IaC facilitado pelo Rony. A utilização do Github no processo de disparo e construção de serviços via “actions” (yaml) facilitou não só a construção dos serviços como a sua destruição e consequente acompanhamento de custos do projeto. Além disso, a detecção de erros foi muito bem entendida justamente por estar bem detalhada e disponibilizada no actions do github o que aumentou a produtividade.

Outro ponto a ser destacado é o fato de que por algumas vezes o serviço, por apresentar algum erro, teve que ser desligado manualmente, o que levou certo tempo.

## 2.2 Sprint 2

A segunda sprint do projeto possui como principal objetivo executar a maior parte dos códigos relacionados à elaboração dos serviços de processamento e análise dos dados. Para isso, foi necessário criar todos os códigos relacionados à criação do cluster Kurbenetes, com os serviços de orquestração de códigos, análise dos clusters e do processamento distribuído via Spark para realizar a execução dos scripts em pyspark. Além das tarefas criadas para essa sprint foram adicionados alguns ajustes percebidos na asprint anterior conforme será explicado de maneira mais detalhada abaixo.

### 2.2.1 Solução

Após a elucidação dos objetivos descritos para a Sprint 2, será realizado um conjunto de evidências e descrições mais específicas das tarefas realizadas ou em progresso dessa Sprint.

- **Evidência do planejamento:**

#### 2.1.1 requisito necessário) *Selecionar as tabelas para o projeto*

Esta etapa foi uma necessidade observada na sprint anterior e foi acrescentada para que possa ser realizada uma seleção das tabelas que de fato serão utilizadas.

#### 2.1.1) *Conversão de formato dos dados*

Etapa responsável por transformar os formatos dos dados de csv para um formato mais interessante como o parquet.

#### 2.1.2) *Tratamento dos dados*

Etapa responsável por tratar os dados em termos de formato de colunas e limpeza para que possa estar pronto para as etapas posteriores de processamento. Foi adicionado um comentário que inclui o ponto de melhoria descrito na sprint 1.

### *2.1.3) Agrupamento dos dados*

Etapa responsável por agrupar os dados das tabelas selecionadas de maneira a fazer sentido para a recomendação e disponibilidade dos dados analíticos.

### *2.2.1) Construir códigos responsáveis pelos componentes de processamento (cluster Kubernetes)*

Etapa responsável pela construção de grande parte da infraestrutura necessária para coletar, tratar e processar os dados. Serão diversos códigos elaborados nesta etapa.

### *2.2.2) Testar componentes*

Verificar se certos componentes do cluster Kubernetes foram corretamente construídos e se estão funcionando adequadamente para o objetivo final.

### *2.2.3) Verificar Custos*

Avaliar os custos na página de billing da AWS e entender qual foi o maior custo no processo de construção dos recursos.

### *2.3.1) Construir consultas SQL que serão utilizadas como testes*

Elaborar algumas consultas como forma de teste para entender se a arquitetura está entregando o necessário.

Para comprovar o progresso da Sprint 2 é possível verificar a Figura 28, onde é possível observar o Trello e os cards em andamento.

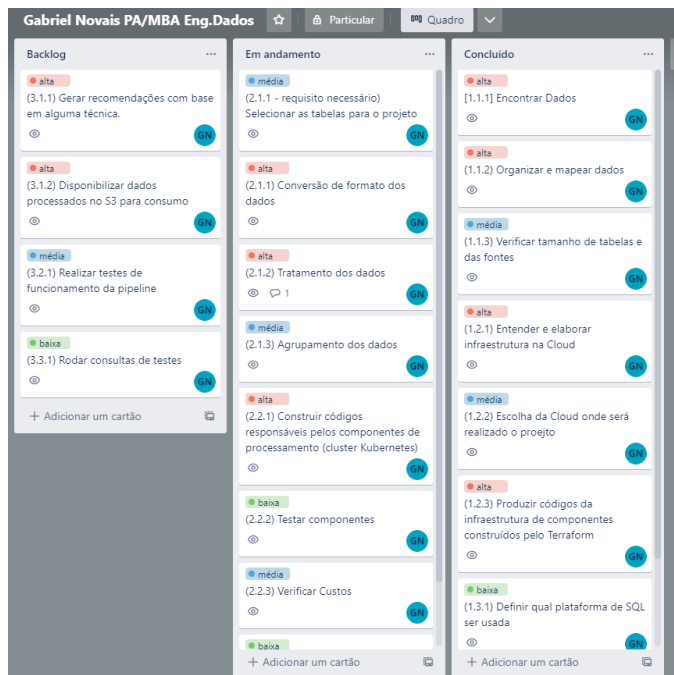


Figura 28 - Sprint 2 em progresso.

Após a realização das tarefas programadas os cards que estavam em andamento foram deslocados para a coluna de concluídas, conforme Figura 29.

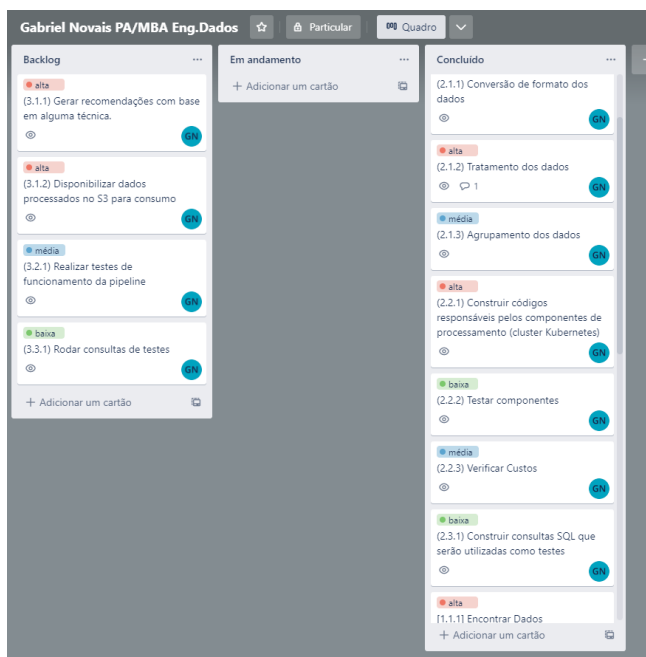


Figura 29 - Sprint 2 com tarefas concluídas

- Evidência da execução de cada requisito:

### 2.1.1 - requisito necessário) Selecionar as tabelas para o projeto

Nesta etapa, que foi um requisito necessário percebido na execução da última sprint, o objetivo foi selecionar previamente as tabelas que de fato serão utilizadas no projeto aplicado. Para a finalidade de servir dados analíticos e recomendações, bastou-se apenas pegar 5 tabelas principais: Customers, order Payments, order Items, order Dataset e order Reviews. A ilustração da seleção pode ser visualizada na Figura 30, construída no Miro.

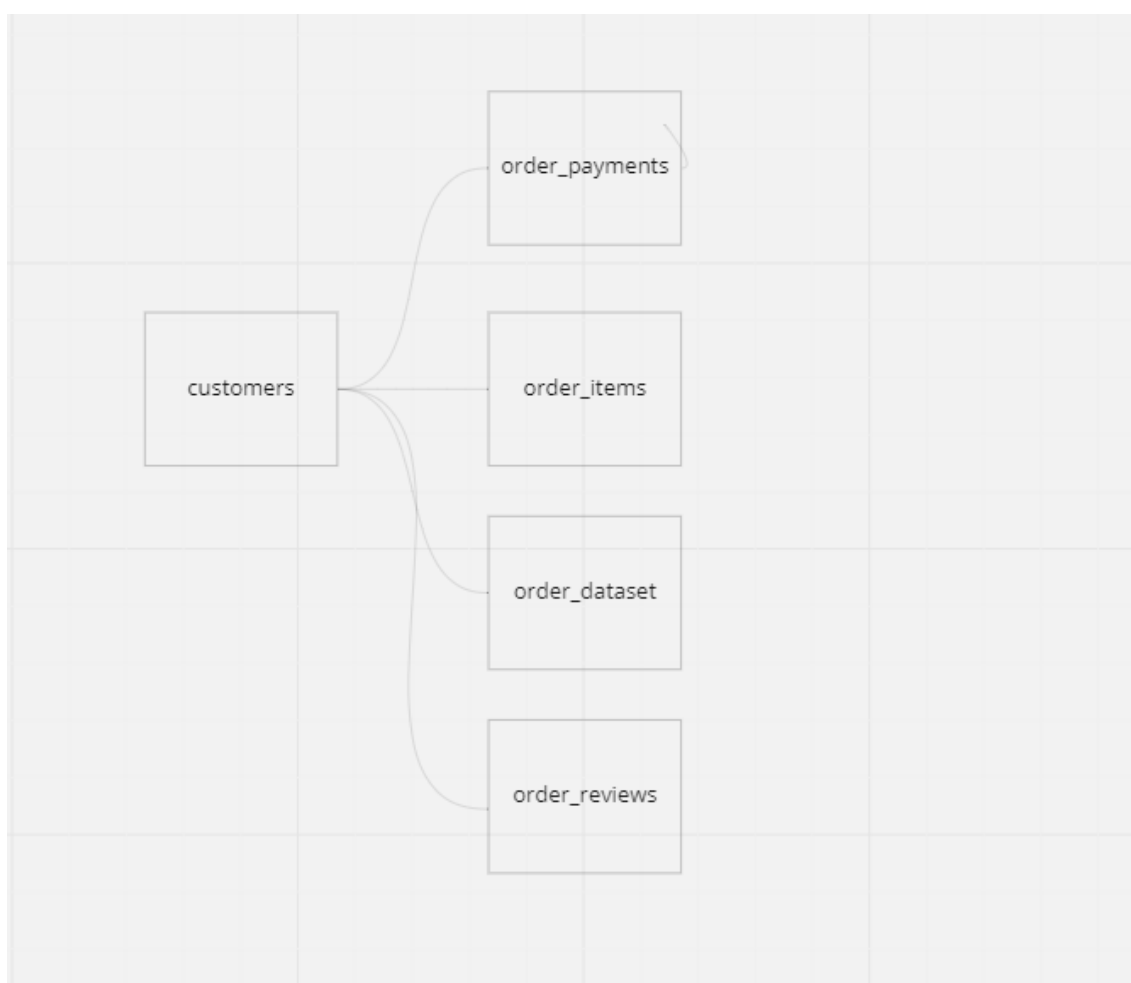
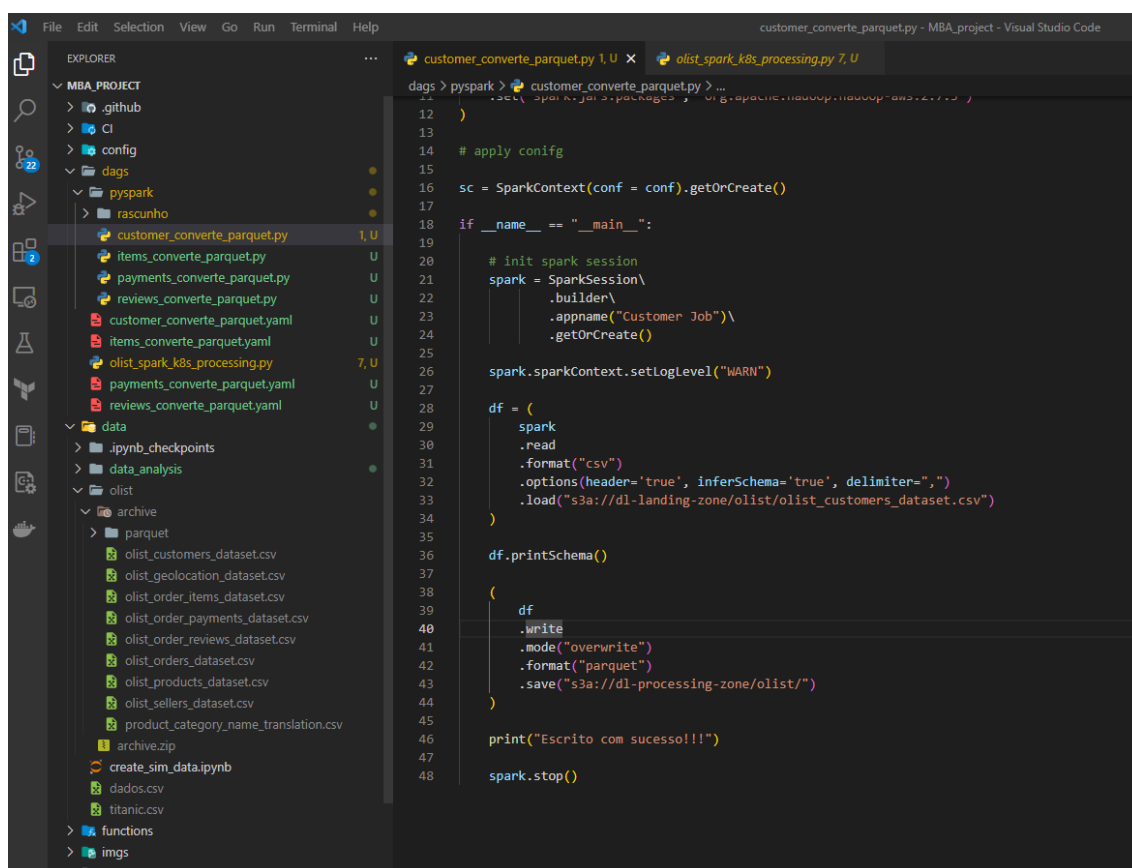


Figura 30 - Tabelas selecionadas

### 2.1.1) Conversão de formato dos dados

Na Figura 31 é possível constatar diversos pontos que demonstram a execução da conversão dos arquivos csv em parquet. O primeiro são os códigos em python da linha 28 até a linha 44, onde verificam-se as instruções para a conversão. Na barra lateral direita é possível também observar os quatro arquivos python e quatro arquivos yml, que serão considerados na orquestração do Airflow.



```

File Edit Selection View Go Run Terminal Help
customer_convert_parquet.py - MBA_project - Visual Studio Code

EXPLORER
MBA_PROJECT
├── .github
├── CI
├── config
├── dags
├── pyspark
├── rascunho
├── customer_convert_parquet.py 1, U
├── items_convert_parquet.py U
├── payments_convert_parquet.py U
├── reviews_convert_parquet.py U
├── customer_convert_parquet.yml U
├── items_convert_parquet.yml U
├── olist_spark_k8s_processing.py 7, U
├── payments_convert_parquet.yml U
├── reviews_convert_parquet.yml U
├── data
├── .ipynb_checkpoints
├── data_analysis
├── olist
├── archive
├── parquet
├── olist_customers_dataset.csv
├── olist_geolocation_dataset.csv
├── olist_order_items_dataset.csv
├── olist_order_payments_dataset.csv
├── olist_order_reviews_dataset.csv
├── olist_orders_dataset.csv
├── olist_products_dataset.csv
├── olist_sellers_dataset.csv
├── product_category_name_translation.csv
├── archive.zip
├── create_sim_data.ipynb
├── dados.csv
├── titanic.csv
├── functions
├── imgs
├── infrastructure

customer_convert_parquet.py 1, U
olist_spark_k8s_processing.py 7, U

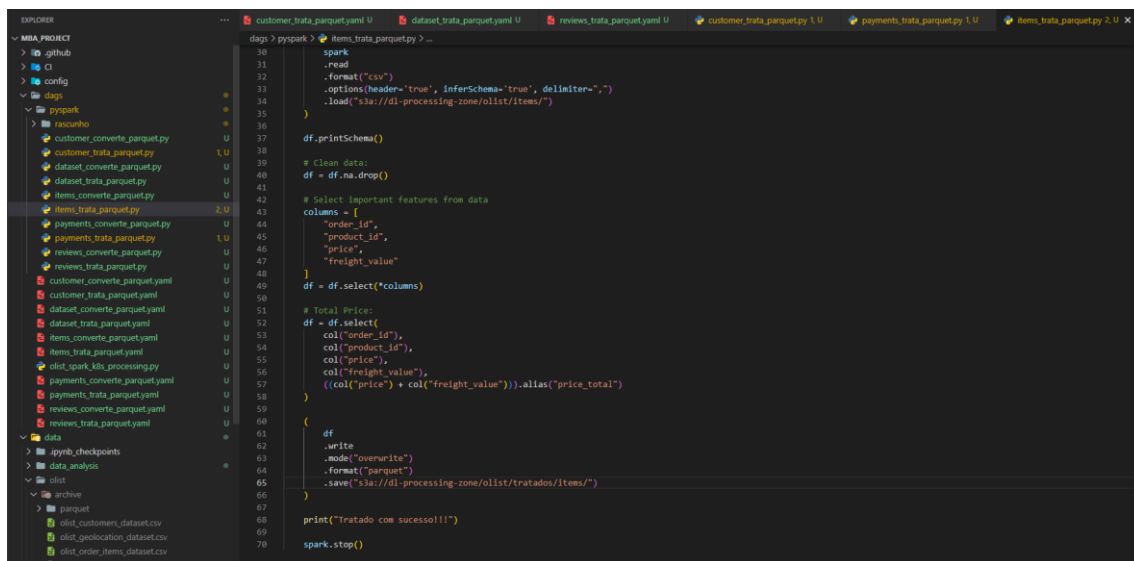
dags > pyspark > customer_convert_parquet.py > ...
11 set( spark.executor.packages += "org.apache.hadoop.hdfs-2.7.1" )
12 )
13
14 # apply config
15
16 sc = SparkContext(conf = conf).getOrCreate()
17
18 if __name__ == "__main__":
19
20     # init spark session
21     spark = SparkSession\
22         .builder\
23         .appName("Customer Job")\
24         .getOrCreate()
25
26     spark.sparkContext.setLogLevel("WARN")
27
28     df = (
29         spark
30         .read
31         .format("csv")
32         .options(header='true', inferSchema='true', delimiter=",")
33         .load("s3a://dl-landing-zone/olist/olist_customers_dataset.csv")
34     )
35
36     df.printSchema()
37
38     (
39         df
40         .write
41         .mode("overwrite")
42         .format("parquet")
43         .save("s3a://dl-processing-zone/olist/")
44     )
45
46     print("Escrito com sucesso!!!")
47
48     spark.stop()
  
```

Figura 31 - Código de transformação do formato dos dados, tabela customers



### 2.1.2) Tratamento dos dados

Na Figura 32 é possível verificar alguns passos posteriores ao que é apresentado na Figura 31, onde tem-se da linha 39 até 57 a limpeza e seleção de features importantes para contruir a tabela final de items. Além disso, também pode-se observar que os demais arquivos de tratamento das tabelas são vistos na barra lateral, no formato python, e abaixo destes, os arquivos yml associados.



```

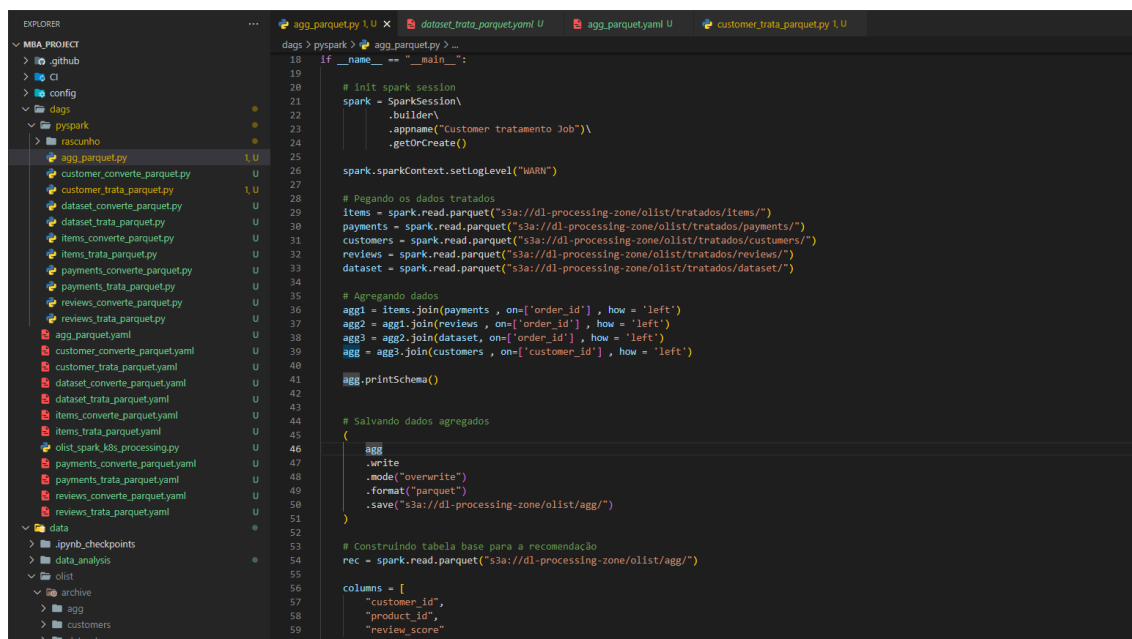
30 spark
31 .read
32 .format("csv")
33 .options(header="true", inferSchema="true", delimiter=",")
34 .load("s3a://di-processing-zone/olist/items/")
35
36
37 df.printSchema()
38
39 # Clean data:
40 df = df.na.drop()
41
42 # Select important features from data
43 columns = [
44     "order_id",
45     "product_id",
46     "price",
47     "freight_value"
48 ]
49 df = df.select(*columns)
50
51 # Total Price:
52 df = df.select(
53     col("order_id"),
54     col("product_id"),
55     col("price"),
56     col("freight_value"),
57     ((col("price") + col("freight_value")).alias("price_total"))
58 )
59
60 (
61     df
62     .write
63     .mode("overwrite")
64     .format("parquet")
65     .save("s3a://di-processing-zone/olist/tratados/items/")
66 )
67
68 print("Tratado com sucesso!!!")
69
70 spark.stop()

```

Figura 32 - Código de tratamento dos dados, tabela de items

### 2.1.3) Agrupamento dos dados

Após ter sido realizado tanto a transformação do formato das tabelas quanto o tratamento delas o próximo passo natural foi a realização da agregação em uma tabela única responsável por fornecer dados tanto para as funcionalidades analíticas da solução, quanto para a recomendação. Na Figura 33 é possível verificar a presença dos códigos em pyspark responsáveis por agregar as tabelas e na barra lateral o arquivo em formato yml associado.



```

EXPLORER
MBA_PROJECT
├── .github
├── .gitignore
├── config
├── dags
├── pyspark
├── resumo
├── agg_parquet.py
├── customer_converte_parquet.py
├── dataset_converte_parquet.py
├── dataset_trata_parquet.py
├── items_converte_parquet.py
├── items_trata_parquet.py
├── payments_converte_parquet.py
├── payments_trata_parquet.py
├── reviews_converte_parquet.py
├── reviews_trata_parquet.py
├── agg_parquet.yaml
├── customer_converte_parquet.yaml
├── dataset_converte_parquet.yaml
├── dataset_trata_parquet.yaml
├── items_converte_parquet.yaml
├── items_trata_parquet.yaml
├── olist_spark_k8s_processing.py
├── payments_converte_parquet.yaml
├── payments_trata_parquet.yaml
├── reviews_converte_parquet.yaml
├── reviews_trata_parquet.yaml
├── data
├── jupyter_checkpoints
├── data_analysis
├── olist
├── archive
├── agg
├── customers
├── dataset

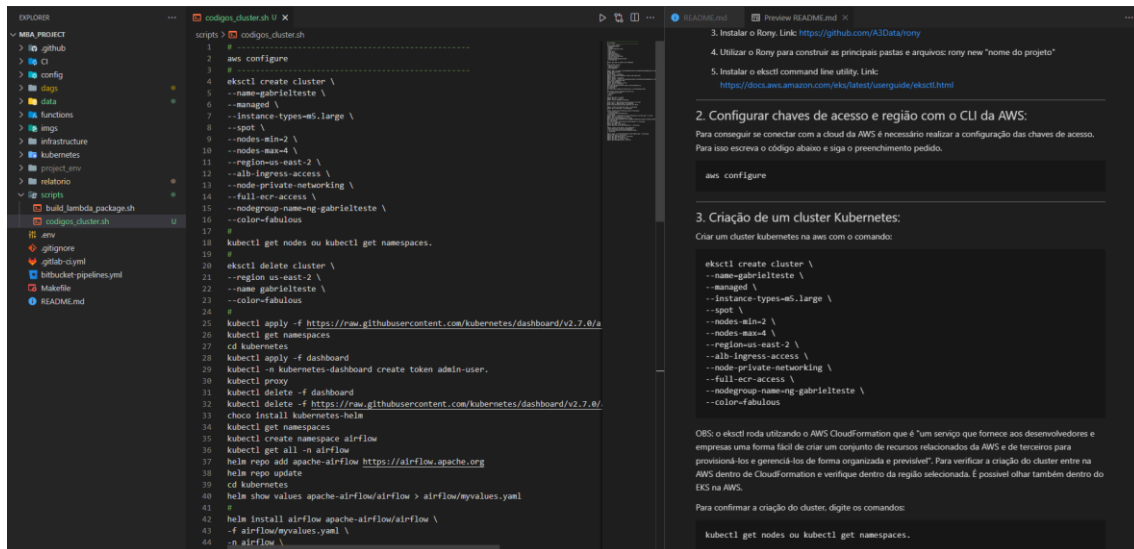
dags > pyspark > agg_parquet.py
18 if __name__ == "__main__":
19
20     # Init spark session
21     spark = SparkSession\
22         .builder\
23         .appName("Customer tratamento Job")\
24         .getOrCreate()
25
26     spark.sparkContext.setLogLevel("WARN")
27
28     # Pegando os dados tratados
29     items = spark.read.parquet("s3a://dl-processing-zone/olist/tratados/items/")
30     payments = spark.read.parquet("s3a://dl-processing-zone/olist/tratados/payments/")
31     customers = spark.read.parquet("s3a://dl-processing-zone/olist/tratados/customers/")
32     reviews = spark.read.parquet("s3a://dl-processing-zone/olist/tratados/reviews/")
33     dataset = spark.read.parquet("s3a://dl-processing-zone/olist/tratados/dataset/")
34
35     # Agregando dados
36     agg1 = items.join(payments, on=["order_id"], how = 'left')
37     agg2 = agg1.join(reviews, on=["order_id"], how = 'left')
38     agg3 = agg2.join(dataset, on=["order_id"], how = 'left')
39     agg = agg3.join(customers, on=["customer_id"], how = 'left')
40
41     agg.printSchema()
42
43     # Salvando dados agregados
44     (
45         agg
46         .write
47         .mode("overwrite")
48         .format("parquet")
49         .save("s3a://dl-processing-zone/olist/agg/")
50     )
51
52
53     # Construindo tabela base para a recomendação
54     rec = spark.read.parquet("s3a://dl-processing-zone/olist/agg/")
55
56     columns = [
57         "customer_id",
58         "product_id",
59         "review_score"
60     ]

```

Figura 33 - Código para agregar as tabelas tratadas

### 2.2.1) Construir códigos responsáveis pelos componentes de processamento (cluster Kubernetes)

Existem duas principais evidências que podem comprovar a construção do cluster Kubernetes. Uma possível são as instruções reunidas no arquivo único em formato sh, cujos códigos se encontram na esquerda da Figura 34. Esses códigos são responsáveis por executar comandos capazes de ordenar a disposição das máquinas EC2 ou EKS na AWS, além de todos serviços necessários. Uma boa parte é realizada de maneira semi automática. No lado direito da mesma figura é possível verificar o Readme do repositório onde estão todos os códigos necessários para aplicação do projeto aplicado (link: [https://github.com/NovaisGabriel/MBA\\_project](https://github.com/NovaisGabriel/MBA_project) ). Neste Readme, estão todas instruções necessárias para reproduzir o projeto, incluindo o cluster Kurbenetes.



```

1 #
2 aws configure
3 #
4 eksctl create cluster \
5   --name=gabrielteste \
6   --managed \
7   --instance-types=m5.large \
8   --spot \
9   --nodes-min=2 \
10  --nodes-max=4 \
11  --region=us-east-2 \
12  --alb-ingress-access \
13  --node-private-networking \
14  --full-ecr-access \
15  --nodegroup-name=ng-gabrielteste \
16  --color=fabulous
17 #
18 kubectl get nodes ou kubectl get namespaces.
19 #
20 eksctl delete cluster \
21   --region=us-east-2 \
22   --name=gabrielteste \
23   --color=fabulous
24 #
25 kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/a
26 kubectl get namespaces
27 cd kubernetes
28 kubectl apply -f dashboard
29 kubectl -n kubernetes-dashboard create token admin-user.
30 kubectl proxy
31 kubectl delete -f dashboard
32 kubectl delete -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/a
33 choco install kubernetes-helm
34 kubectl get namespaces
35 kubectl create namespace airflow
36 kubectl get all -n airflow
37 helm repo add apache-airflow https://airflow.apache.org
38 helm repo update
39 cd kubernetes
40 helm show values apache-airflow/airflow > airflow/myvalues.yaml
41 #
42 helm install airflow apache-airflow/airflow \
43   -f airflow/myvalues.yaml \
44   -n airflow \

```

3. Criar um cluster Kubernetes:

Chamar um cluster kubernetes na aws com o comando:

```

eksctl create cluster \
  --name=gabrielteste \
  --managed \
  --instance-types=m5.large \
  --spot \
  --nodes-min=2 \
  --nodes-max=4 \
  --region=us-east-2 \
  --alb-ingress-access \
  --node-private-networking \
  --full-ecr-access \
  --nodegroup-name=ng-gabrielteste \
  --color=fabulous

```

OBS: o eksctl roda utilizando o AWS CloudFormation que é "um serviço que fornece aos desenvolvedores e empresas uma forma fácil de criar um conjunto de recursos relacionados da AWS e de terceiros para provisionar e gerenciar os de forma organizada e previsível". Para verificar a criação do cluster entre na AWS dentro de CloudFormation e verifique dentro da região selecionada. É possível olhar também dentro do EKS na AWS.

Para confirmar a criação do cluster, digite os comandos:

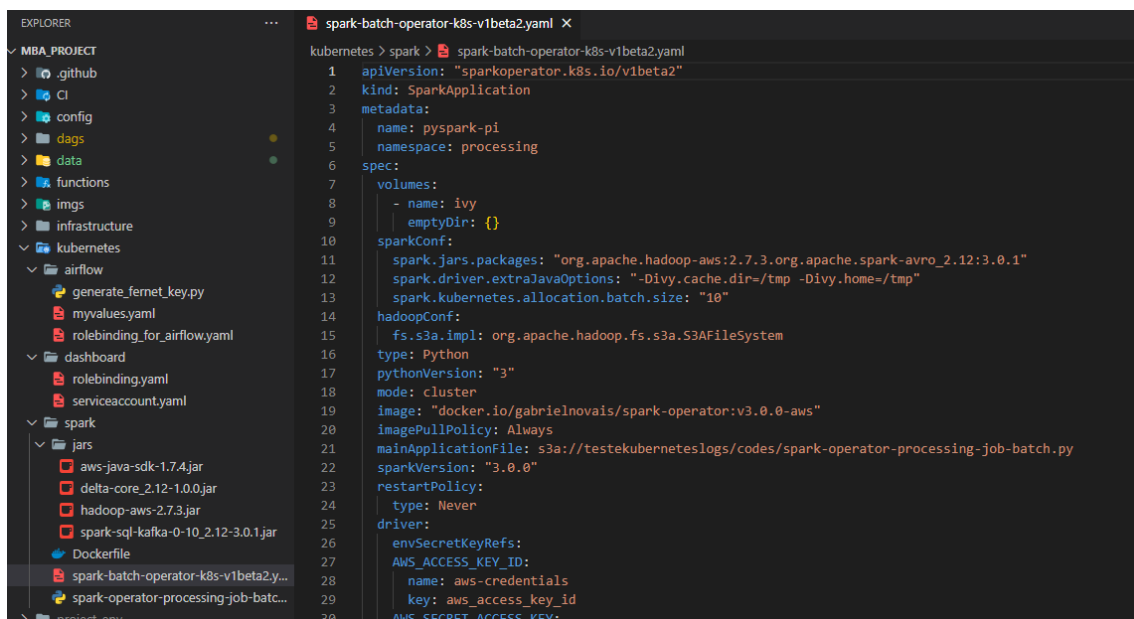
```

kubectl get nodes ou kubectl get namespaces.

```

Figura 34 - Comandos para executar construção do cluster e serviços na AWS

Adicionalmente podemos ver na Figura 35, em especial na barra lateral, outros códigos relacionados aos requisitos necessários para levantar toda a estrutura que é executada conforme os comandos explicitados na Figura 34.



```

1 apiVersion: "sparkoperator.k8s.io/v1beta2"
2 kind: SparkApplication
3 metadata:
4   name: pyspark-pi
5   namespace: processing
6 spec:
7   volumes:
8     - name: ivy
9       emptyDir: {}
10  sparkConf:
11    spark.jars.packages: "org.apache.hadoop-aws:2.7.3.org.apache.spark-avro_2.12:3.0.1"
12    spark.driver.extraJavaOptions: "-Divy.cache.dir=/tmp -Divy.home=/tmp"
13    spark.kubernetes.allocation.batch.size: "10"
14  hadoopConf:
15    fs.s3a.impl: org.apache.hadoop.fs.s3a.S3AFileSystem
16  type: Python
17  pythonVersion: "3"
18  mode: cluster
19  image: "docker.io/gabrielnovais/spark-operator:v3.0.0-aws"
20  imagePullPolicy: Always
21  mainApplicationFile: s3a://testekuberneteslogs/codes/spark-operator-processing-job-batch.py
22  sparkVersion: "3.0.0"
23  restartPolicy:
24    type: Never
25  driver:
26    envSecretKeyRefs:
27      AWS_ACCESS_KEY_ID:
28        name: aws-credentials
29        key: aws_access_key_id
30      AWS_SECRET_ACCESS_KEY:

```

Figura 35 - Arquivos yml necessários para construção do cluster

### 2.2.2) Testar componentes

Existem alguns serviços e componentes que podem ser testados para analisar se a construção do cluster foi realizada com sucesso. Na Figura 36 é possível verificar que existem namespaces e tem-se o IP do cluster na AWS funcionando.

```
PS C:\Users\Gabriel\Desktop\backup\Repositorios\MBA_project> kubectl get pods
No resources found in default namespace.
PS C:\Users\Gabriel\Desktop\backup\Repositorios\MBA_project> kubectl get namespaces
NAME                STATUS   AGE
airflow              Active   54m
default              Active   73m
kube-node-lease      Active   73m
kube-public          Active   73m
kube-system          Active   73m
kubernetes-dashboard Active   62m
PS C:\Users\Gabriel\Desktop\backup\Repositorios\MBA_project> kubectl get all
NAME                TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes  ClusterIP     10.100.0.1   <none>        443/TCP    73m
PS C:\Users\Gabriel\Desktop\backup\Repositorios\MBA_project> []
```

Figura 36 - Teste para verificar se cluster está funcionando

### 2.2.3) Verificar Custos

A análise de custos foi observada de perto, pois os recursos de cloud não costumam ser muito acessíveis. Uma das verificações realizadas foram após o deploy do cluster, de onde na Figura 37 é possível observar que de fato o recurso que consumiu maior parte dos recursos financeiros foi o EC2.

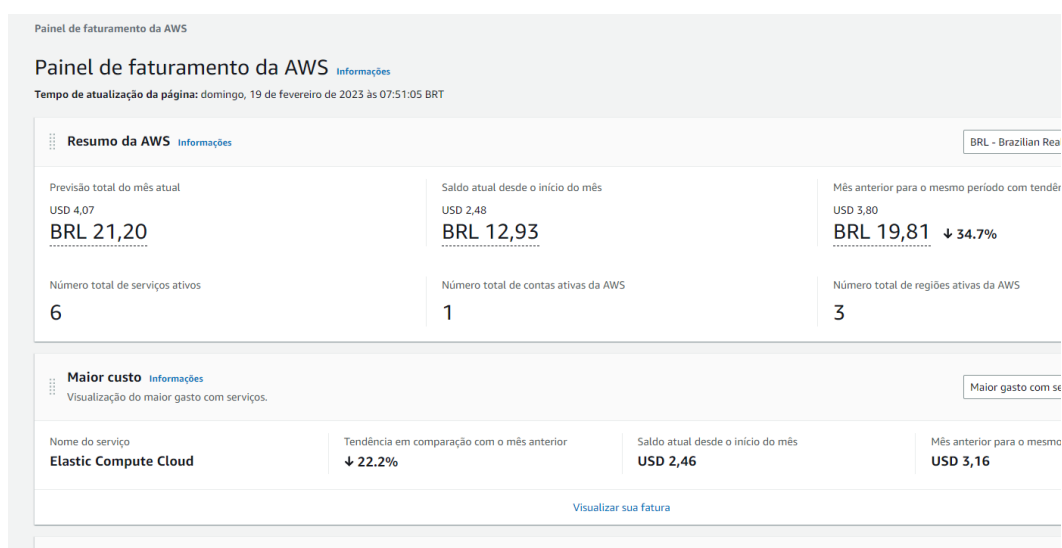


Figura 37 - Custos após deploy do cluster

### 2.3.1) Construir consultas SQL que serão utilizadas como testes

Algumas consultas de teste nos dados analíticos foram construídas para que posteriormente na sprint a seguir possa ser testada. Uma consulta de exemplo pode ser vista na Figura 38, que nos informa a distribuição média dos valores numéricos para os estados dos clientes selecionados de acordo com o nível de pontuação dada na review da compra.

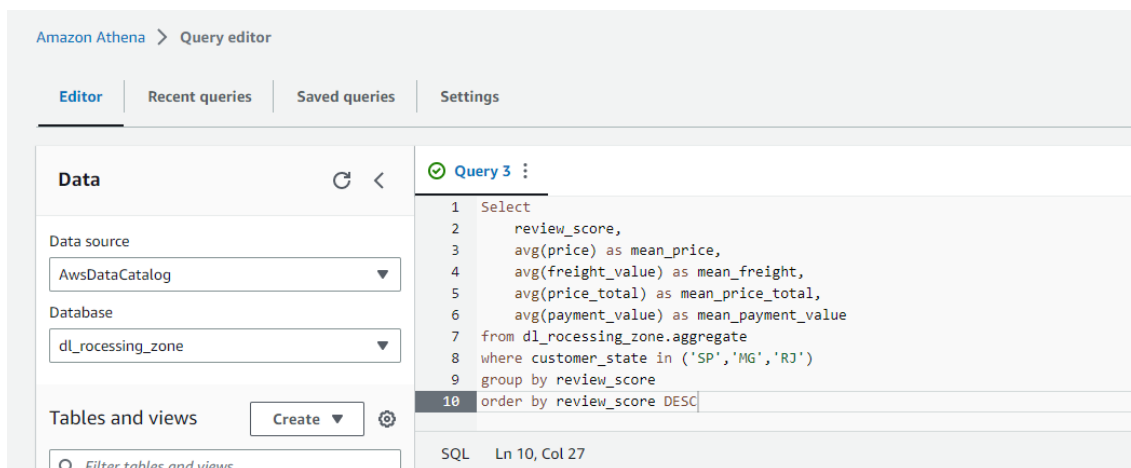


Figura 38 - Consulta de teste, exemplo para dados analíticos

- Evidência dos resultados:

Os principais resultados obtidos nesta sprint estão sem dúvida relacionados à construção dos arquivos pyspark para realizar toda conversão de formato, o tratamento, a limpeza e a agregação dos dados para que possam ser utilizados no cluster, além da construção e levantamento do cluster kubernetes na AWS. Na Figura 39 é possível verificar o serviço do cluster ativo na AWS.

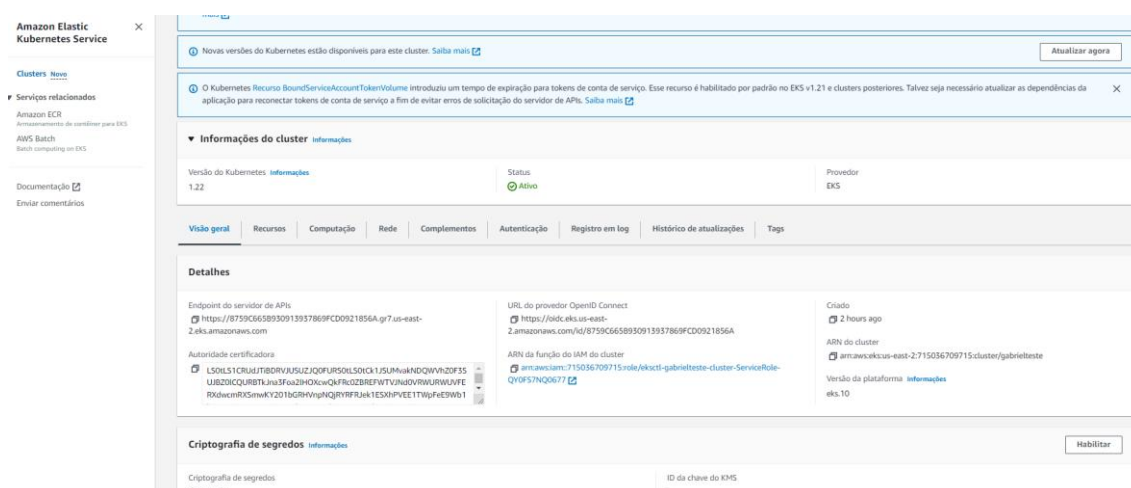


Figura 39 - Cluster kubernetes ativo na AWS

Na figura 40 é possível observar o serviço de CloudFormation ativo na AWS. Para contextualizar, segundo a própria documentação da AWS, o serviço “CloudFormation permite modelar, provisionar e gerenciar recursos da AWS e de terceiros ao tratar a infraestrutura como código”.

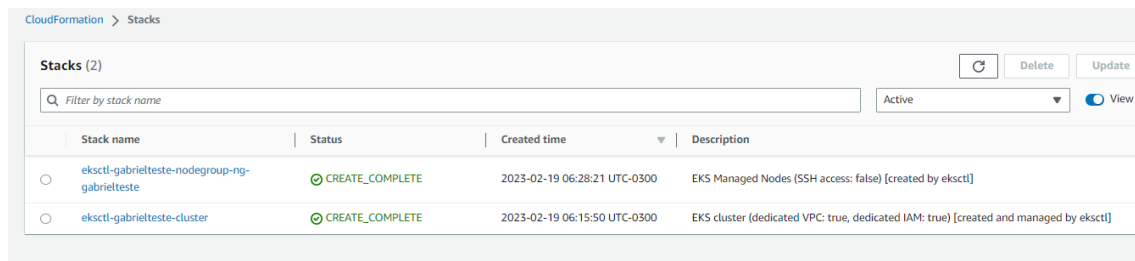


Figura 40 - CloudFormation ativo na AWS

Na figura 41, podemos acessar o CloudFormation mais de perto e verificar as stacks ativas no painel da AWS.

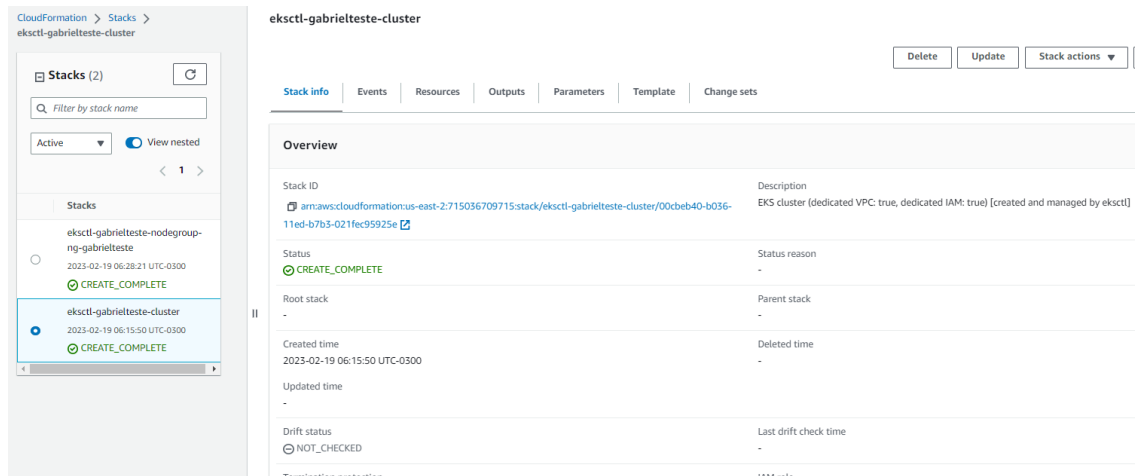


Figura 41 - Stacks do CloudFormation ativas

Na figura 42 é possível verificar os recursos ativos no EC2 devido a criação do cluster pelo serviço de Elastic Kubernetes Services. Isso demonstra com efeito que os serviço estão funcionando e prontos para utilização.

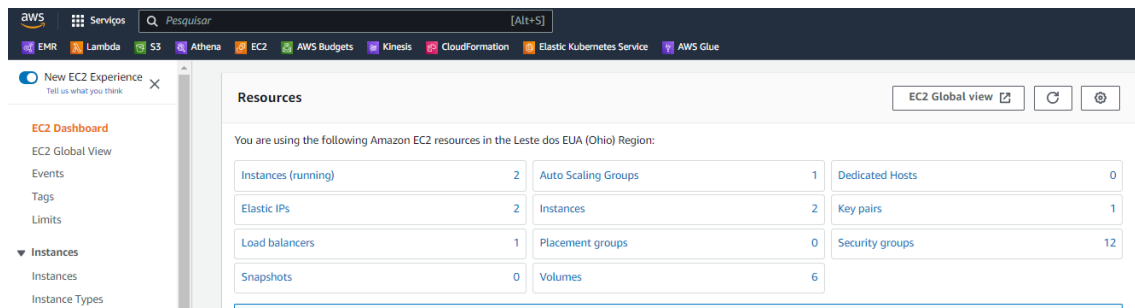


Figura 42 - Recursos ativos no EC2

### 2.2.2 Experiências vivenciadas

Ao longo da Sprint, algumas tarefas possuíram níveis de execução diferentes. As tarefas menos complexas foram relacionadas a construção dos códigos em pyspark, que apesar de serem repletos de detalhes tanto em termos de código quanto de cuidado na manipulação dos dados de teste, pois as tarefas estavam muito bem definidas e a qualidade das informações coletadas estavam suficientemente boas. As atividades mais complexas se voltaram na construção do cluster, que tiveram certos ajustes de configuração e atualização de pacotes de maneira inesperada. Entretanto foi possível executar as tarefas desejadas sem maiores problemas.

Vale ressaltar que, além de ter sido incorporada de maneira fluida as observações realizadas na última sprint, foi possível também incluir uma tarefa que não estava sendo mapeada, que foi a criação de uma seleção específica da tabela agregada para a criação da recomendação. Esta demanda foi concluída e incorporada com sucesso no código pyspark no momento da criação da tabela agregada. Essa construção observada durante a execução da atual sprint, facilitará a próxima sprint.

Outra observação pode ser vista em termos financeiros, pois conforme observado na tarefa 2.2.3, verificação de custos, os custos com máquinas no serviço EC2 foram elevados. Este é um cuidado de extrema relevância para a execução de projetos deste tipo.



## 2.3 Sprint 3

Uma grande parte das tarefas a serem realizadas foram feitas na Sprint 2, de maneira que nessa Sprint 3 os objetivos são mais relacionados a finalizações e complementações dos objetivos dessa última Sprint. Os dois principais objetivos, com nível de relevância alto, referem-se a criação de uma maneira de recomendar ou indicar produtos com base nos dados agregados destinados, e disponibilizar no S3 os dados de forma integral para ser utilizado na pipeline completa desenhada na arquitetura do projeto. Alguns testes simples deverão ser aplicados para fins de sanity check.

### 2.3.1 Solução

Após a elucidação dos objetivos descritos para a Sprint 2, será realizado um conjunto de evidências e descrições mais específicas das tarefas realizadas ou em progresso dessa Sprint.

- **Evidência do planejamento:**

*(3.1.1) Gerar recomendações com base em alguma técnica.*

Etapa responsável por escolher alguma técnica relativamente simples de recomendação de produtos de forma mais personalizada, e em seguida, implementar em pyspark.

*(3.1.2) Disponibilizar dados processados no S3 para consumo*

Etapa responsável por fazer a ingestão total dos dados selecionados para o folder destinado aos dados sem tratamento no bucket S3.

*(3.2.1) Realizar testes de funcionamento da pipeline*

Etapa responsável por verificar se o pipeline está funcionando e está adequado para uso em produção.

### (3.3.1) Rodar consultas de testes

Etapa responsável por testar se os dados fornecidos para consumo estão de fato possibilitando consultas.

Após descrição das etapas, realiza-se a observação das tarefas no trello e coloca-se os cards na área de “em andamento” conforme a sua execução. Na Figura 43, é possível verificar o progresso do projeto na sprint.

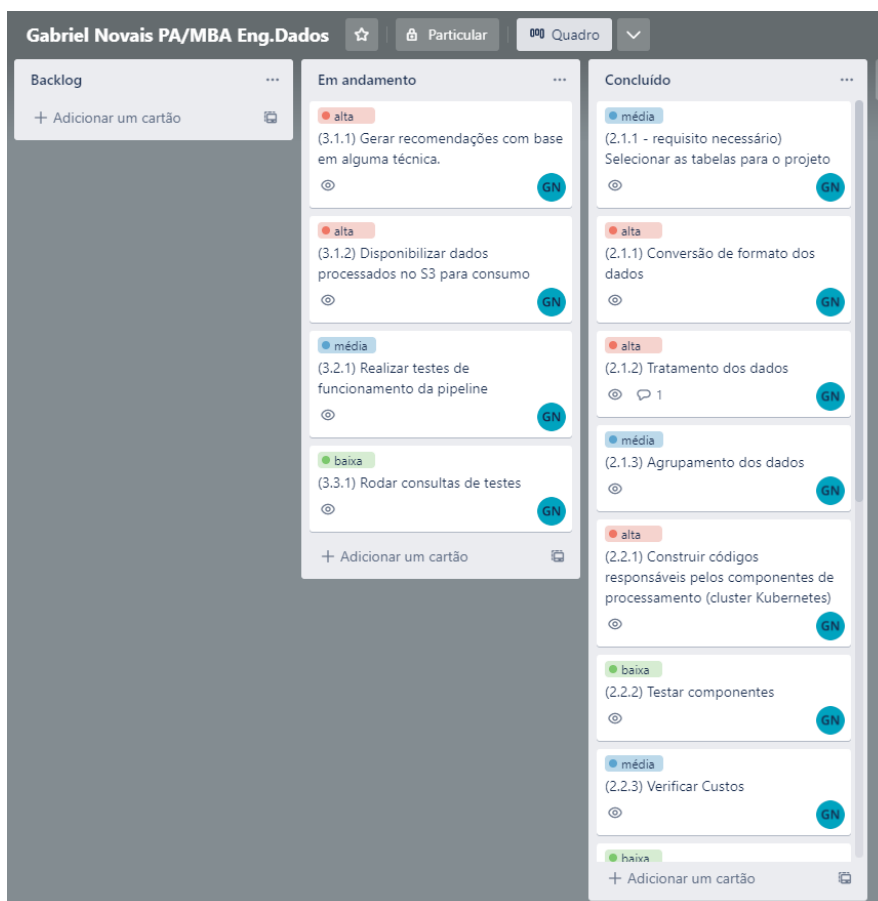


Figura 43 - Progresso da Sprint 3

Além de visualizar na Figura 43 os cards em andamento, verifica-se também que o backlog está vazio, de forma que o projeto segue para sua conclusão. Na Figura 44 é possível observar a conclusão dos cards, e assim a finalização dessa sprint.

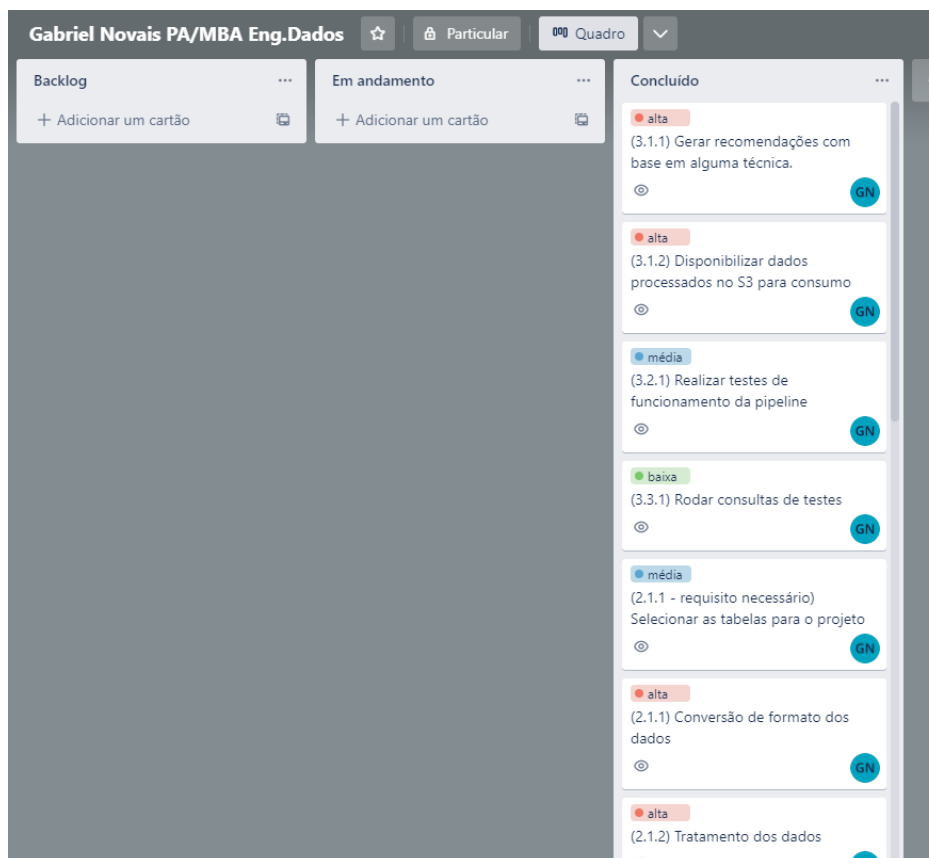


Figura 44 - Conclusão da Sprint 3

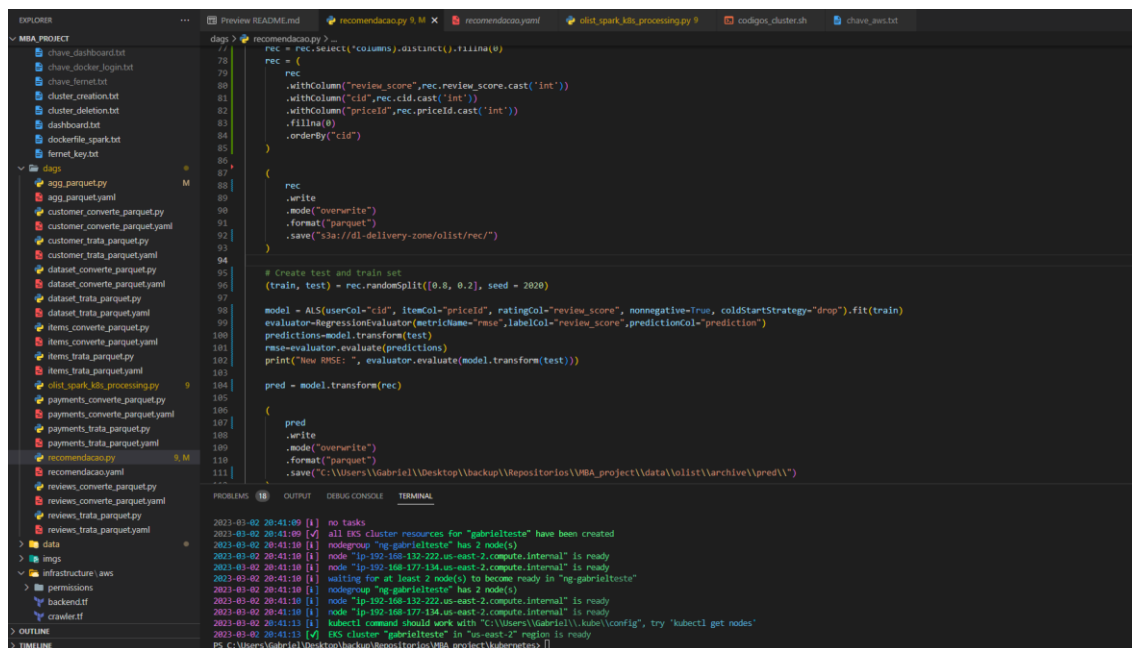
- Evidência da execução de cada requisito:

(3.1.1) Gerar recomendações com base em alguma técnica

Através dos dados agregados que foram salvos na zona de processamento no S3 é possível extrair uma cópia para a mesma zona sob a pasta de recomendação e também para a zona de entrega (delivery) de forma que sirva para os estudos analíticos.

Com os dados de processamento agregados foi utilizado o código apresentado na Figura 45, onde é possível visualizar o tratamento dos dados para enquadrar (no exemplo de técnica) uma tabela com usuários, faixas de preços dos produtos vistos segundo a lógica descrita nos fluxos do código e a pontuação da review daquele consumidor. Uma observação é que para não ser tão custoso os testes em cima da técnica que será descrita, foi realizada uma recomendação em uma amostra dos dados totais. Entretanto, vale ressaltar que o ambiente e a técnica em si são altamente escláveis e não enfrentam muitos problemas para tratar de Big Data.

A técnica escolhida como forma de trabalhar a recomendação foi o *Alternating Least Squares matrix factorization* (ALS), um filtro colaborativo com cálculos baseados em fatorização de matrizes, através de aprendizado de máquina (machine learning). Um link interessante para verificar a documentação da técnica encontra-se em <https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.recommendation.ALS.html>.



```

//
rec = rec.select($"columns", $"distinct", $"fillna(0)")
rec = (
    .withColumn("review_score", rec.review_score.cast("int"))
    .withColumn("cid", rec.cid.cast("int"))
    .withColumn("priceId", rec.priceId.cast("int"))
    .fillna(0)
    .orderBy("cid")
)

rec
    .write
    .mode("overwrite")
    .format("parquet")
    .save("s3a://dl-delivery-zone/olist/rec/")

# Create test and train set
(train, test) = rec.randomSplit([0.8, 0.2], seed = 2020)

model = ALS(userCol="cid", itemCol="priceId", ratingCol="review_score", nonnegative=True, coldStartStrategy="drop").fit(train)
evaluator = RegressionEvaluator(metricName="rmse", labelCol="review_score", predictionCol="prediction")
predictions = model.transform(test)
rmse = evaluator.evaluate(predictions)
print("New RMSE: ", evaluator.evaluate(model.transform(test)))

pred = model.transform(rec)

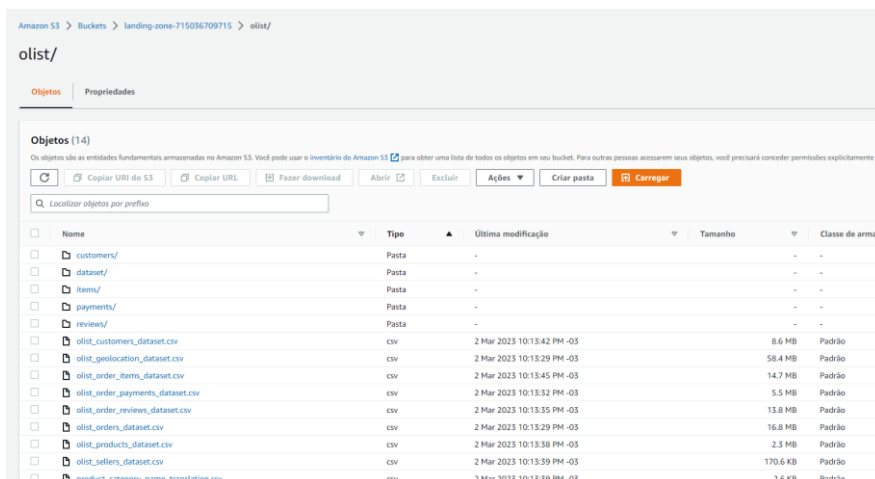
pred
    .write
    .mode("overwrite")
    .format("parquet")
    .save("C:\\Users\\Gabriel\\Desktop\\back\\Repositories\\VBA_project\\data\\olist\\archive\\pred\\")

```

Figura 45 - Código com a técnica de ALS (MLLib do spark) para recomendação

### (3.1.2) Disponibilizar dados processados no S3 para consumo

Através das Figuras 46 e 47 é possível verificar como estão as estruturas das zonas landing e processing, nas quais estão os dados crus e processados após ter sido rodado a pipeline dos dados.



Nome	Tipo	Última modificação	Tamanho	Classe de armazenamento
customers/	Pasta	-	-	-
dataset/	Pasta	-	-	-
items/	Pasta	-	-	-
payments/	Pasta	-	-	-
reviews/	Pasta	-	-	-
olist_customers_dataset.csv	csv	2 Mar 2023 10:13:42 PM -03	8.6 MB	Padrão
olist_geolocation_dataset.csv	csv	2 Mar 2023 10:13:29 PM -03	58.4 MB	Padrão
olist_order_items_dataset.csv	csv	2 Mar 2023 10:13:45 PM -03	14.7 MB	Padrão
olist_order_payments_dataset.csv	csv	2 Mar 2023 10:13:52 PM -03	5.5 MB	Padrão
olist_order_reviews_dataset.csv	csv	2 Mar 2023 10:13:35 PM -03	13.8 MB	Padrão
olist_orders_dataset.csv	csv	2 Mar 2023 10:13:29 PM -03	16.8 MB	Padrão
olist_products_dataset.csv	csv	2 Mar 2023 10:13:38 PM -03	2.3 MB	Padrão
olist_sellers_dataset.csv	csv	2 Mar 2023 10:13:39 PM -03	170.6 KB	Padrão
product_category_name_translation.csv	csv	2 Mar 2023 10:13:39 PM -03	2.6 KB	Padrão

Figura 46 - Dados na zona landing prontos para processamento

Na zona de processamento existem duas pastas muito importantes para o resultado final. Elas são: agg e rec. Através delas é que é realizada a disponibilidade dos dados analíticos na zona de entrega e a construção do mecanismo de recomendação que futuramente disponibilizará os dados de também na zona de entrega.

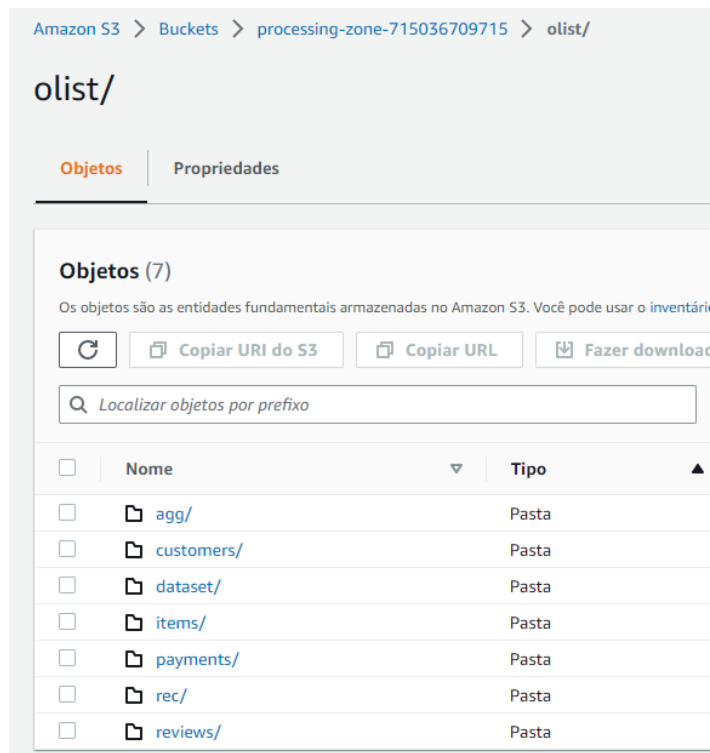


Figura 47 - Dados processados na zona processing

### (3.2.1) Realizar testes de funcionamento da pipeline

A presente tarefa tem como objetivo apresentar algumas observações realizadas nas pipelines de provisionamento de recursos na nuvem e também na pipeline de conversão, tratamento e geração de dados para consumo. Na Figura 48 pode-se verificar o início dos “checks” para realizar merge no github.

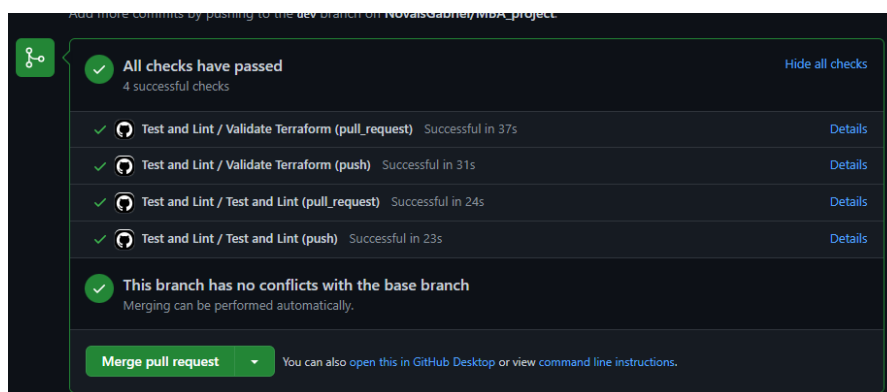


Figura 48 - Checks para realizar o merge no github branch main

Na Figura 49 é possível verificar a tela do github actions, como consequência das ações anteriores, o actions foi então acionado de forma a rodar os códigos do terraform nas funções build, plan e apply.

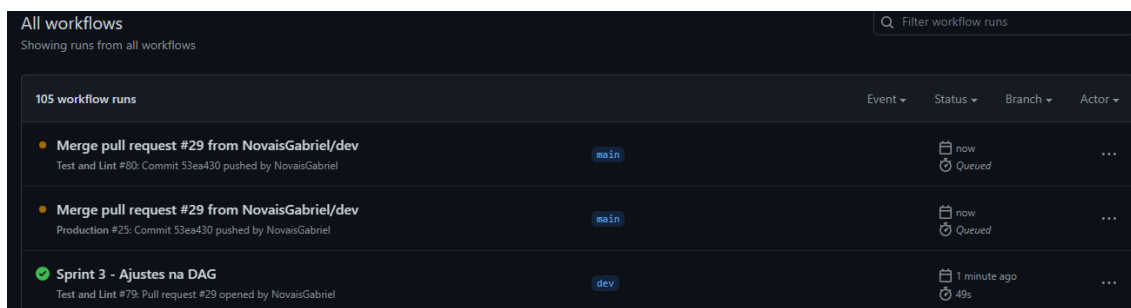


Figura 49 - Actions acionado pelo merge

Na Figura 50 é possível verificar a execução dos códigos terraform para provisionamento e que de fato o actions está rodando e gerando os recursos na nuvem.

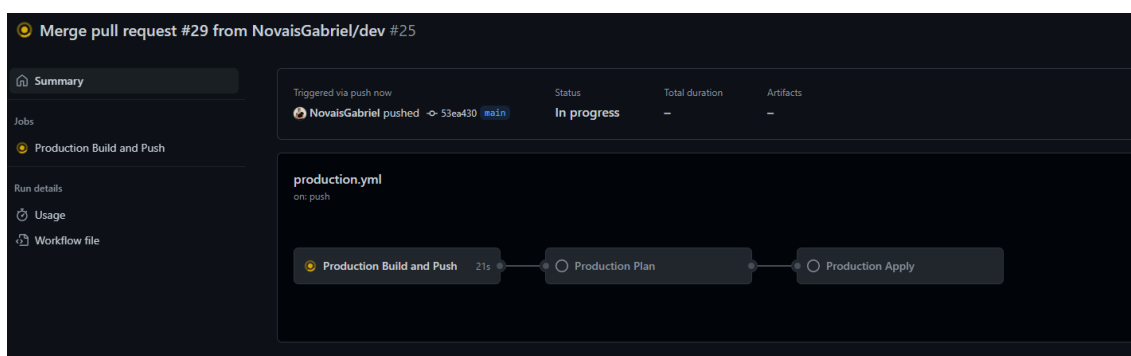


Figura 50 - Terraform rodando no actions do github

Na Figura 51 é possível verificar o quanto de tempo levou o provisionamento dos recursos do Terraform.

Run and billable time	
Learn about OS pricing on GitHub Actions	
Job	Run time
Production Build and Push	37s
Production Plan	39s
Production Apply	40s
	1m 56s

Figura 51 - Tempo para o Terraform prover recursos na AWS

Após o Terraform ser executado o próximo passo é a execução da pipeline descrita na Figura 52, no formato de tree. Esse pipeline é responsável por executar as dags que farão toda execução dos códigos em pyspark para tratamento, agregação e disponibilização dos dados para consumo na área de entrega.

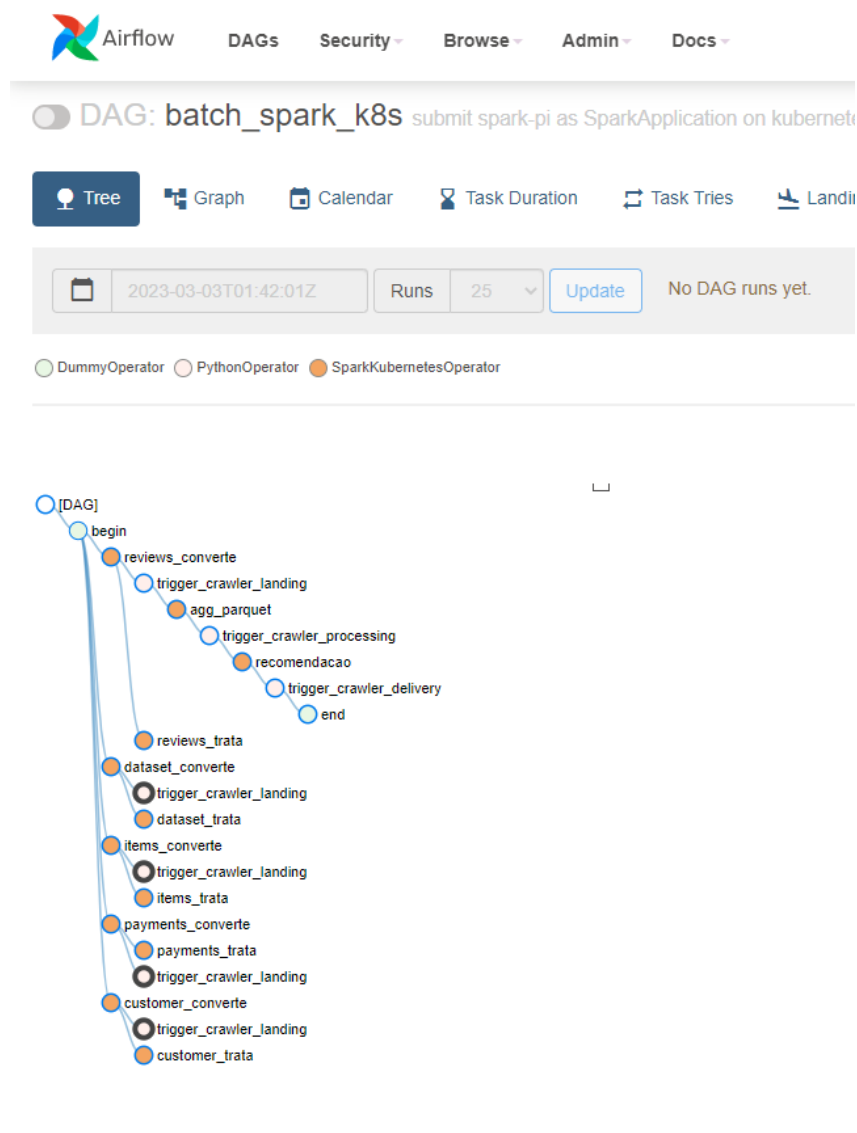


Figura 52 - Airflow pipeline no formato tree

Na Fugura 53 é possível verificar a mesma pipeline sendo executado (running), mas apresentada no formato graph.

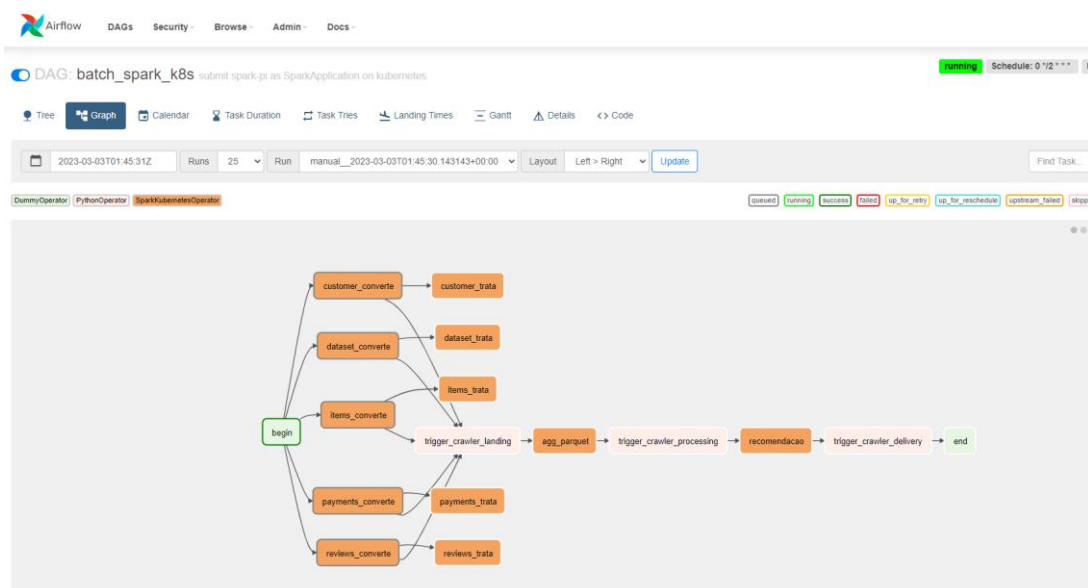


Figura 53 - Airflow pipeline no formato graph

### (3.3.1) Rodar consultas de testes

As consultas de testes servem para verificar o funcionamento do ambiente de consumo dos dados no Athenas, de forma que fique evidente as diversas possibilidades de análises que podem ser aproveitadas dos dados. Na Figura 54 é possível observar o funcionamento correto de uma consulta que visa agregar os reviews scores de forma a obter as médias de preços, fretes e valores de pagamento para os estados de São Paulo, Minas Gerais e Rio de Janeiro, ordenados pela maior pontuação de score na review.

Query 5 :

```

1 Select
2   review_score,
3   avg(price) as mean_price,
4   avg(freight_value) as mean_freight,
5   avg(price_total) as mean_price_total,
6   avg(payment_value) as mean_payment_value
7 from d1_processing_zone.aggregate
8 where customer_state in ('SP','MG','RJ')
9 group by review_score
10 order by review_score DESC

```

SQL Ln 10, Col 27

Run again Explain Cancel Clear Create

Query results Query stats

Completed Time in queue: 108 ms Run time: 976 ms Data scanned: 34.37 MB

Results (6)

Search rows

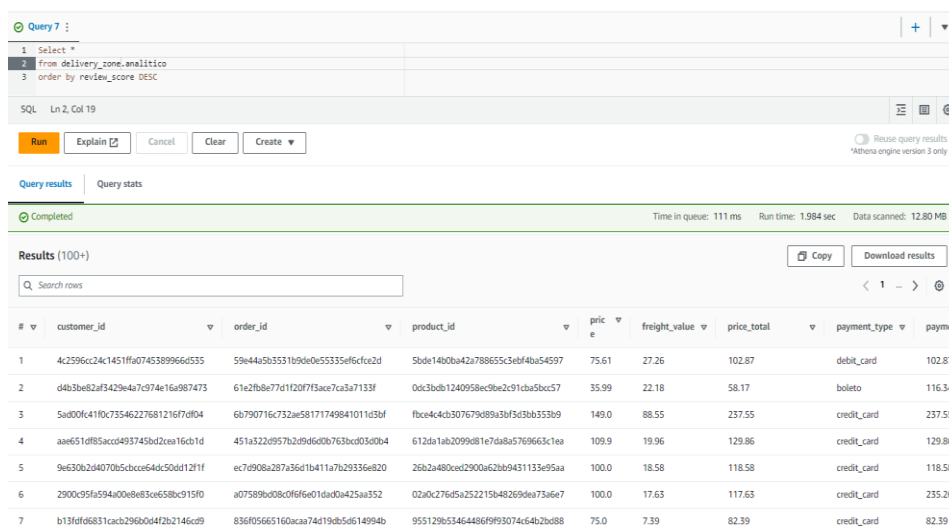
#	review_score	mean_price	mean_freight	mean_price_total	mean_payment_value
1	5.0	118.64004	17.676561	136.3166	160.06873
2	4.0	112.418175	19.156399	131.57457	160.50754
3	3.0	116.14036	18.45235	134.59271	179.86795
4	2.0	102.77837	21.255621	124.034	190.98859
5	1.0	120.78676	18.17302	138.95978	213.53647
6		113.712585	17.167519	130.88011	159.97115

Figura 54 - Consulta de teste realizada na tabela de análises



- Evidência dos resultados:

A Sprint 3 é a última sprint do projeto e por este motivo ela possui uma característica muito forte de conclusão. Dessa maneira os resultados obtidos por esta sprint, terminam por se transformar também no próprio resultado do projeto aplicado. Na Figura 55 é possível verificar que a consulta nos dados analíticos (por mais simples que seja) está funcionando bem, e dos dados disponíveis na tabela pode-se obter diversos insights interessantes para o negócio inclusive o acompanhamento de estatísticas de venda.



Query 7 :

```

1 select *
2 from delivery_zone_analitico
3 order by review_score DESC

```

SQL Ln 2, Col 19

Run Explain Cancel Clear Create

Query results Query stats

Completed Time in queue: 111 ms Run time: 1.984 sec Data scanned: 12.80 MB

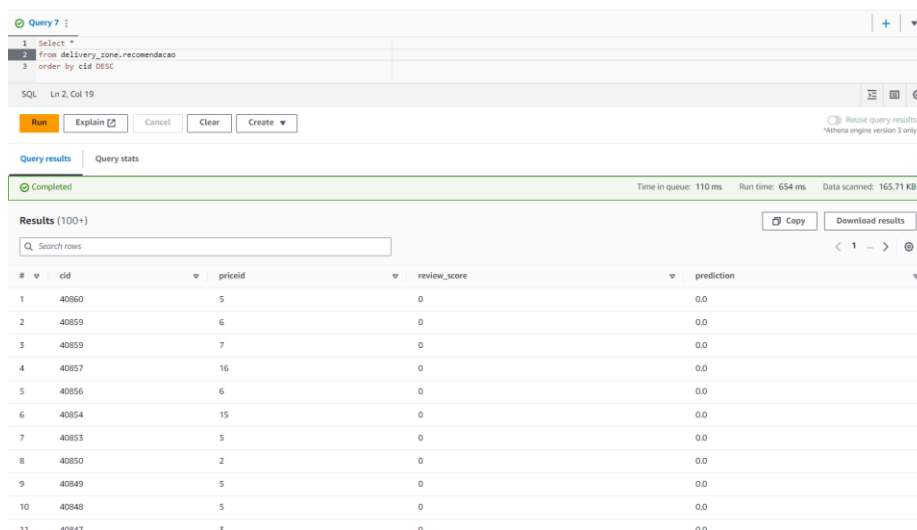
Results (100+)

Search rows

#	customer_id	order_id	product_id	price	freight_value	price_total	payment_type	payment
1	4c2596cc24c1451ffa0745389966d535	59e44a5b3531b9de0e5335ef6cfa2d	5bde14b0ba42a788655c3ebf4ba54597	75.61	27.26	102.87	debit_card	102.87
2	d4b3be82af3429e4a7c974e16a987473	61e2fb8e77d1120f73ace7ca3a7133f	0dc3bdc1240958ec9be2c91da5bdc57	35.99	22.18	58.17	boleto	116.34
3	5ad00f41f0c735462276812167d0d4	6b790716c732ae58171749841011d3bf	fbce4c4cb307679d89a3bf5d3bb353b9	149.0	88.55	237.55	credit_card	237.55
4	aae51df85acc0493745bd2cea16db1d	451a322d957b2d9d6db763bcd03d0b4	612da1ab2099d81e7da8a5769663c1ea	109.9	19.96	129.86	credit_card	129.86
5	9e630b2d4070b5cbccc54dc50dd12f1f	ec7e908a287a36d1b41a7b29336e820	26b2a480ced2900a62bb9431133e95aa	100.0	18.58	118.58	credit_card	118.58
6	2900c95fa594a00e8e83c658bc915f0	a07589bd08c0f6f6e01dad0a425aa352	02a0c276d5a252215b48269dea73a6e7	100.0	17.63	117.63	credit_card	235.26
7	b13f0f66831cacb296b0d472b2146cd9	836f05665160acaa74d19db5d614994b	955129b53464486f993074c64b2bd88	75.0	7.39	82.39	credit_card	82.39

Figura 55 - Consulta à tabela de dados analíticos

A Figura 56 possui objetivo similar ao que foi apresentado na Figura 55, de modo que é possível verificar os dados de recomendação.



Query 7 :

```

1 select *
2 from delivery_zone_recomendacao
3 order by cid DESC

```

SQL Ln 2, Col 19

Run Explain Cancel Clear Create

Query results Query stats

Completed Time in queue: 110 ms Run time: 654 ms Data scanned: 165.71 KB

Results (100+)

Search rows

#	cid	priceid	review_score	prediction
1	40860	5	0	0.0
2	40859	6	0	0.0
3	40859	7	0	0.0
4	40857	16	0	0.0
5	40856	6	0	0.0
6	40854	15	0	0.0
7	40853	5	0	0.0
8	40850	2	0	0.0
9	40849	5	0	0.0
10	40848	5	0	0.0
11	40847	5	0	0.0

Figura 56 - Consulta à tabela de dados de recomendação

Nesses dados tem-se em cada linha um consumidor, uma faixa de preço recomendada e uma pontuação (score) que entrega o quão bom é aquela recomendação para aquele consumidor. Dessa maneira seria, por exemplo, possível identificar as faixas de preços ideais para cada consumidor e fornecer os produtos relativos à essa faixa.

### 2.3.2 Experiências vivenciadas

Esta Sprint é a última do projeto e dessa forma é possível verificar que os resultados dela também representam em boa medida os resultados do projeto. Existiram menos tarefas que as sprints anteriores, mas isso não implicou em uma maior facilidade de execução, uma vez que foram detectadas algumas modificações necessárias para a adaptação do código de recomendação, além de que precisou-se organizar melhor o repositório no github onde os códigos estavam armazenados.

Uma grande dificuldade foi adaptar uma técnica de machine learning pelo pyspark para gerar um sistema de recomendação simples. Os dados não facilitaram o processo de preparação para o input no algoritmo de ALS. Além disso, evitou-se avaliar em demasia a qualidade dos resultados, uma vez que a intenção é a exposição da possibilidade de modelar algo nesse sentido. Para facilitar a modelagem, também foi realizada uma amostragem para reduzir o tempo de debugg do código.

Outra dificuldade que atrapalhou o bom andamento do projeto foi a reconstrução do ambiente (toda vez é realizada a destruição dos recursos para evitar a cobrança de valores na AWS), que foi gerado por certas atualizações nas ferramentas utilizadas, como o kubernetes e o repositório do helm. Várias adaptações tiveram que ser feitas. A observação da pipeline, após os ajustes necessários, foi então produzida.

## 3. Considerações Finais

### 3.1 Resultados

Por meio de um texto detalhado, apresente os principais resultados alcançados pelo seu Projeto Aplicado.

Cite os pontos positivos e negativos, as dificuldades enfrentadas e as experiências vivenciadas durante todo o processo.

### 3.2 Contribuições

Apresente quais foram as contribuições que o seu Projeto Aplicado trouxe para que o Desafio proposto fosse solucionado.

Cite, por exemplo, as inovações, as vantagens sobre os similares, as melhorias alcançadas, entre outros.

### 3.3 Próximos passos

Descreva quais são os próximos passos que poderão contribuir com o aprimoramento da solução apresentada pelo seu Projeto Aplicado.