

\_\_\_\_\_

PÓS-GRADUAÇÃO

**XP Educação**

**Relatório do Projeto Aplicado**

ARQUITETURA DE REFERÊNCIA

Pedro Henrique Padilha Portella da Cruz

Orientador(a):

Professor Reinaldo Galvão

18 de novembro de 2024

****

Pedro Henrique Padilha Portella da Cruz

**XP EDUCAÇÃO**RELATÓRIO DO PROJETO APLICADO

ARQUITETURA DE REFERÊNCIA

Relatório de Projeto Aplicado desenvolvido para fins de conclusão do curso de Arquitetura de Software e Soluções.  
  
Orientador (a): Professor Reinaldo Galvão

**São Paulo - SP  
31 de outubro de 2024**

**Sumário**

[1. CANVAS do Projeto Aplicado](#_heading=h.xuo90f2eremm) [4](#_heading=h.xuo90f2eremm)

[Desafio](#_heading=h.1fob9te) [5](#_heading=h.1fob9te)

[1.1.1 Análise de Contexto](#_heading=h.3znysh7) [5](#_heading=h.3znysh7)

[1.1.2 Personas](#_heading=h.2et92p0) [6](#_heading=h.2et92p0)

[1.1.3 Benefícios e Justificativas](#_heading=h.3dy6vkm) [7](#_heading=h.3dy6vkm)

[1.1.4 Hipóteses](#_heading=h.1t3h5sf) [8](#_heading=h.1t3h5sf)

[1.2 Solução](#_heading=h.2s8eyo1) [9](#_heading=h.2s8eyo1)

[1.2.1 Objetivo SMART](#_heading=h.26in1rg) [9](#_heading=h.26in1rg)

[1.2.2 Premissas e Restrições](#_heading=h.lnxbz9) [11](#_heading=h.lnxbz9)

[1.2.3 Backlog de Produto](#_heading=h.35nkun2) [13](#_heading=h.35nkun2)

[2. Área de Experimentação](#_heading=h.44sinio) **14**

[2.1 Sprint 1](#_heading=h.z337ya) [16](#_heading=h.z337ya)

[2.1.1 Solução](#_heading=h.3j2qqm3) [16](#_heading=h.3j2qqm3)

[Evidência do planejamento:](#_heading=h.1y810tw) [16](#_heading=h.1y810tw)

[Evidência da execução de cada requisito:](#_heading=h.4i7ojhp) [16](#_heading=h.4i7ojhp)

[Evidência dos resultados:](#_heading=h.49x2ik5) [16](#_heading=h.49x2ik5)

[2.1.2 Lições Aprendidas](#_heading=h.2p2csry) [16](#_heading=h.2p2csry)

[2.2 Sprint 2](#_heading=h.3o7alnk) [17](#_heading=h.3o7alnk)

[2.2.1 Solução](#_heading=h.ihv636) [17](#_heading=h.ihv636)

[Evidência do planejamento:](#_heading=h.32hioqz) [17](#_heading=h.32hioqz)

[Evidência da execução de cada requisito:](#_heading=h.1hmsyys) [17](#_heading=h.1hmsyys)

[Evidência dos resultados:](#_heading=h.4ucy7e5o0v6k) [17](#_heading=h.4ucy7e5o0v6k)

[2.2.2 Lições Aprendidas](#_heading=h.41mghml) [17](#_heading=h.41mghml)

[2.3 Sprint 3](#_heading=h.2grqrue) [18](#_heading=h.2grqrue)

[2.3.1 Solução](#_heading=h.vx1227) [18](#_heading=h.vx1227)

[Evidência do planejamento:](#_heading=h.3fwokq0) [18](#_heading=h.3fwokq0)

[Evidência da execução de cada requisito:](#_heading=h.1v1yuxt) [18](#_heading=h.1v1yuxt)

[Evidência dos resultados:](#_heading=h.h0200faw66br) [18](#_heading=h.h0200faw66br)

[2.3.2 Lições Aprendidas](#_heading=h.4f1mdlm) [18](#_heading=h.4f1mdlm)

[3. Considerações Finais](#_heading=h.2u6wntf) **19**

[3.1 Resultados](#_heading=h.19c6y18) [19](#_heading=h.19c6y18)

[3.2 Contribuições](#_heading=h.3tbugp1) [19](#_heading=h.3tbugp1)

[3.3 Próximos passos](#_heading=h.28h4qwu) 19

## 

## 

## 

## 1. CANVAS do Projeto Aplicado

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



## Desafio

### 1.1.1 Análise de Contexto

Uma empresa de e-commerce chamada Geek Shopping, que atua no setor de varejo, de forma 100% digital, possuí seu sistema totalmente desenvolvido em C#, utilizando o framework .Net Core, em uma arquitetura MVC (model, view, controller) monolítica, e com uma infraestrutura própria.

A empresa vem aumentando seu Market share no setor e buscar concorrer diretamente com os grandes players do segmento, como Amazon, Mercado Livre e Ali Express. Porém com o aumento do número de clientes e de vendas, o gargalo de processamento no sistema vem aumentando consideravelmente, inclusive causando quedas no sistema em momentos de pico, como vésperas de feriados e black Friday.

A partir desse contexto, nos reunimos com representantes dos times de negócios, vendas, do financeiro, arquitetos de software e de soluções e desenvolvedores da empresa, para entender as dificuldades enfrentadas pelas equipes, suas propostas e dúvidas. Reunimos essas informações e elaboramos uma matriz CSD, para termos diferentes óticas de análise do problema.

**Matriz CSD**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Certezas** | **Suposições** | **Dúvidas** |
| **Atores** | O time de negócios e de vendas querem impulsionar o número de clientes e de vendas.  O time de Atendimento ao Cliente vem recebendo reclamações de problemas de indisponibilidade e desempenho do sistema.  O time de T.I deseja melhorar o sistema para que seja mais performático.  O financeiro não quer liberar recursos para compra de mais hardware. | O time de T.I acredita que a mudança para uma arquitetura em microsserviços e usar computação em nuvem aumentará a escalabilidade e resiliência, especialmente durante picos de tráfego.  O time de negócios e financeiro presumem que a migração para nuvem reduzirá custos a longo prazo. | O custo da nuvem será mais vantajoso a longo prazo comparado à manutenção da infraestrutura própria? |
| **Cenários** | A quantidade de acessos ao site vem aumentando, mas também oscila muito durante o ano.  Pedidos, ordens, pagamentos e outros serviços, precisam ser mais performáticos. |  | Qual provedor de nuvem seria o mais adequado para o sistema? |
| **Regras** | O sistema deverá ser capaz de processar um volume crescente de pedidos sem comprometer o desempenho.  A infraestrutura atual deve ser escalável e resiliente para suportar diferentes cargas de trabalho, especialmente em picos de tráfego.  Será necessário a implementação de uma nova arquitetura. | A arquitetura deve ser modular, e cada serviço deve ser escalável independentemente. |  |

A partir do bate papo com representantes dos times de negócios, vendas, do financeiro, arquitetos de software e de soluções e desenvolvedores da empresa, uma matriz POEMS foi gerada:

**Matriz POEMS**

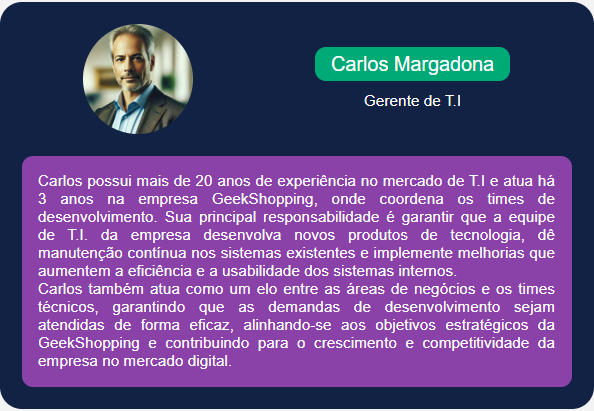
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pessoas** | **Objetos** | **Ambiente** | **Mensagem** | **Serviços** |
| Quem está envolvido no contexto em análise? | Que objetos fazem parte do ambiente? | Quais são as características do ambiente? | Que mensagens são comunicadas? | Quais serviços são oferecidos? |
| **Time de Negócios e de Vendas** | Equipamentos de trabalho, como Notebooks, Monitores e Periféricos | A reunião foi de forma remota (via Microsoft Teams) | Estratégias para melhorar as vendar e como o tempo de resposta do sistema impacta o cliente | Análise de métricas de vendas e estudo de novos produtos, funcionalidades e serviços para os clientes. |
| **Time de Suporte e Atendimento ao cliente** | Equipamentos de trabalho, como Notebooks, Monitores e Periféricos | A reunião foi em uma sala de vidro no meio do escritório | Como funciona o processo de atendimento e de resolução de problemas dos clientes | Atendimento e suporte via chat ou telefone para os clientes |
| **Time de T. I** | Equipamentos de trabalho, como Notebooks, Monitores e Periféricos | A reunião foi de forma remota (via Microsoft Teams) | Foram discutidas as soluções técnicas viáveis para a construção de uma nova arquitetura | Desenvolvimento de novas funcionalidades e manutenção do sistema. |
| **Time do Financeiro** | Equipamentos de trabalho, como Notebooks, Monitores e Periféricos, além de um telão e um projetor | Uma sala de reunião em formato de U, com um projetor no teto e um telão no início do U | Foi discutida a possibilidade de disponibilização de recursos financeiros para o projeto | Gerenciamento dos recursos financeiros da empresa |

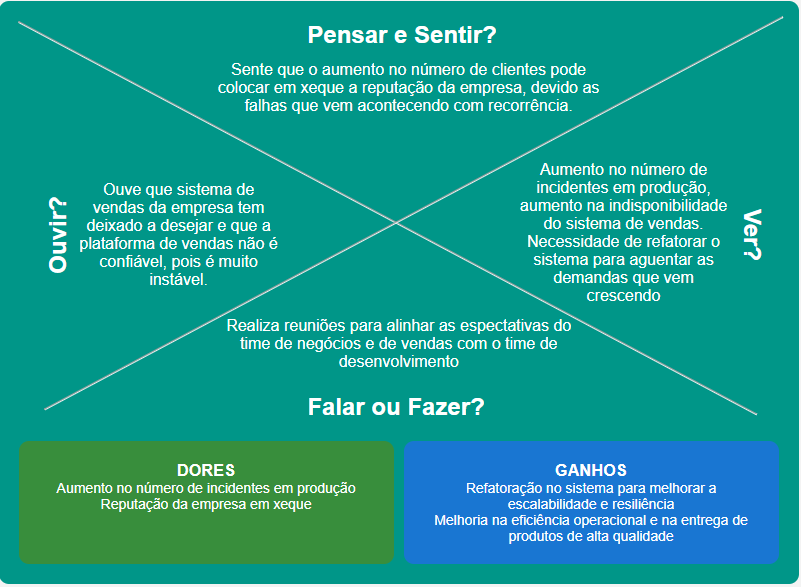
|  |  |
| --- | --- |
| **Registros** | **Insights** |
| As informações iniciais foram obtidas através de bate papos com os representantes dos times de Negócios e Vendas, de T. I, de Suporte e Atendimento ao Cliente, e representantes do time do Financeiro. | * Envolver mais o time de Suporte e Atendimento ao Cliente, pois eles têm uma maior noção das dores dos clientes. * Criar um canal aberto entre o time de negócios e de T.I, pois esses times precisam estar em sincronia para desenvolver um sistema que atenda aos requisitos de Negócios da empresa. |

### 1.1.2 Personas

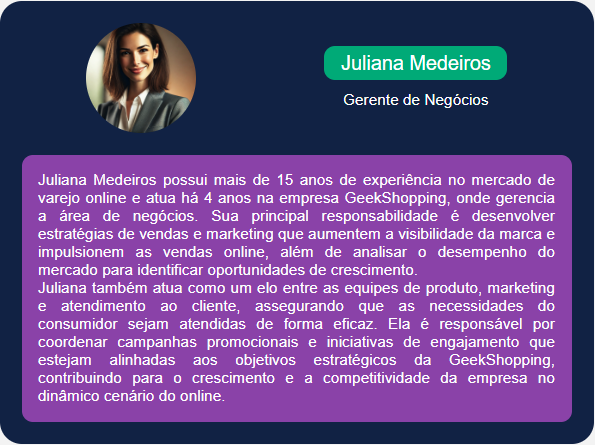
Ao final das entrevistas, foi possível definir o perfil das personas e o mapa de empatia.

**Perfil do Gerente de T.I**



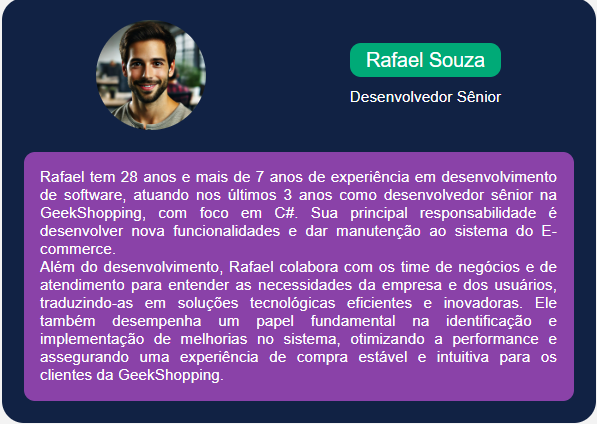


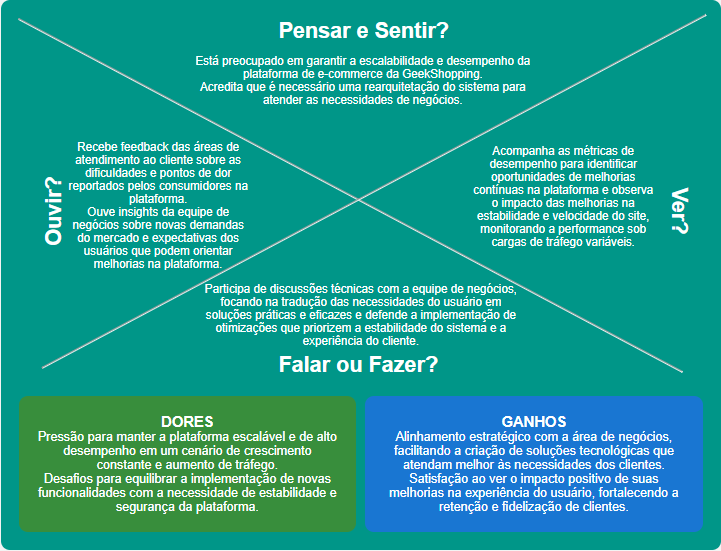
**Perfil da Gerente de Negócios**





**Perfil do Desenvolvedor Sênior**



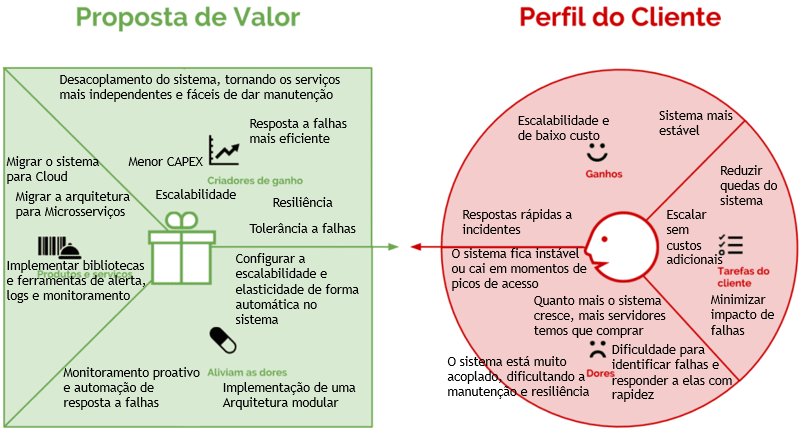


1.1.3 Benefícios e Justificativas

Blueprint do Desafio:

|  |  |  |  |
| --- | --- | --- | --- |
| **Ação** | Melhorar a estabilidade do sistema | Aumentar a capacidade do sistema sem aumentar os custos | Reduzir o tempo de reação e o impacto de falhas |
| **Objetivos** | Reduzir quedas do sistema em picos de acesso | Tornar a capacidade do sistema escalável, sem acréscimo de CAPEX | Reduzir o tempo de reação e o impacto de falhas em produção |
| **Atividades** | Identificar gargalos de processamento | Identificar uma solução que tornar o sistema escalável sem aumento de custos | Configurar monitoramento e alertas em tempo real |
| **Questões** | Como tornar o sistema mais estável? | Como podemos escalar sem custos adicionais? | Como minimizar o impacto de falhas em produção? |
| **Barreiras** | O sistema é um monolito | Limitação de orçamento, da infraestrutura própria e falta de profissionais especializados em Computação em nuvem | Ausência de monitoramento em tempo real e dependência de processos manuais de resposta |
| **Funcionalidades** | Migrar o sistema para uma arquitetura em Microsserviços | Implementação de soluções em nuvem, por exemplo, AWS, hospedando nossos serviços em instâncias EC2, e nosso banco de dados em buckets S3 | Configurar um sistema de monitoramento, utilizando bibliotecas de Logs do C# (como o Serilog), o Amazon CloudWatch coletar métricas e logs na nuvem, configuração de alertas com Amazon SNS e automação de processos com AWS Lambda |
| **Interação** | Reunir os desenvolvedores e arquitetos para discutir a arquitetura | Contratar profissionais de Computação em nuvem | Manter uma equipe de suporte à infraestrutura 24hrs |
| **Mensagem** | Melhorar a performance e a escalabilidade | Escalar o sistema para suportar aumento de clientes e vendas | Responder rapidamente para evitar problemas prolongados |
| **Onde ocorre** | Servidor Local ou na Nuvem | Ambiente em Nuvem | Ambiente de Produção |
| **Tarefas aparentes** | Aumentar a capacidade de processamento do sistema | Planejar e executar a migração para a nuvem | Receber alertas e executar ações corretivas |
| **Tarefas escondidas** | Planejar e executar a migração para a microsserviços | Calcular o CAPEX e OPEX da solução On Premise e na Cloud | Definir processos de escalonamento e resposta rápida a falhas |
| **Processos de suporte** | Treinamento do time técnico para lidar com a arquitetura de microsserviços | Treinamento do time técnico para lidar com o ambiente em nuvem | Suporte contínuo em operações e sistemas de alerta |
| **Saída desejável** | Sistema mais estável, com menos quedas, escalável e | Maior capacidade de processamento, escalabilidade e diminuição de custos | Resolução rápida de falhas e mitigação de impactos |

Após estudar as dores, possíveis soluções e mapear as atividades e processos, foi gerado o CANVAS da proposta de valor:



### 1.1.4 Hipóteses

Matriz de observações para hipóteses, gerada a partir do conhecimento aprofundado do contexto do desafio e da definição das personas:

|  |  |
| --- | --- |
| **Observação** | **Hipótese** |
| O sistema apresenta instabilidade em momentos de pico de acesso. | A migração para uma arquitetura em microsserviços somada a capacidade de escalar facilmente na nuvem, reduzirá a sobrecarga, melhorando a estabilidade. |
| O sistema exige investimentos adicionais em servidores conforme a demanda cresce. | A adoção de uma infraestrutura em nuvem com escalabilidade automática diminuirá o CAPEX e os custos operacionais. |
| A equipe enfrenta dificuldade para localizar e corrigir falhas em tempo hábil. | A implementação de um sistema de monitoramento em tempo real reduzirá o tempo de resposta a incidentes. |
| A infraestrutura monolítica dificulta a manutenção dos serviços. | Desacoplar o sistema em microsserviços tornará a manutenção mais ágil e a escalabilidade mais eficiente. |
| As falhas em produção causam impactos significativos devido à ausência de automação na resposta. | A automação de processos com AWS Lambda e alertas proativos reduzirá os impactos e o tempo de resposta. |
| Falta de conhecimento técnico da equipe sobre tecnologias em nuvem. | Contratar especialistas em computação em nuvem, treinar a equipe e seguir o Cloud Adoption Framework. |
| Falhas de comunicação entre microsserviços podem comprometer a estabilidade do sistema. | A adoção de um sistema de mensageria (ex.: Kafka, RabbitMQ ou Amazon SQS) reduzirá o acoplamento e melhorará a resiliência do sistema. |
| Existe baixa visibilidade das métricas de uso e desempenho do sistema. | Centralizar logs e métricas em uma plataforma de monitoramento unificada, como Amazon CloudWatch trará maior visibilidade e controle. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Itens | B | A | S | I | C | O | Total | Prioridade |
| Migrar a arquitetura monolítica para Microsserviços | 5 | 5 | 5 | 4 | 4 | 3 | 26 | 1 |
| Migrar a infraestrutura local para uma solução em nuvem | 5 | 5 | 5 | 1 | 4 | 1 | 21 | 2 |
| Configurar ferramentas de Alertas, Monitoramento e Logging | 3 | 2 | 3 | 4 | 4 | 4 | 20 | 3 |
| Configurar ferramenta de automação de processos | 3 | 2 | 3 | 4 | 4 | 4 | 20 | 3 |
| Implementar um sistema de Mensageria | 3 | 2 | 3 | 4 | 4 | 4 | 20 | 3 |

Com base nas tarefas identificadas na proposta de valor (melhorar estabilidade, escalabilidade, e capacidade de resposta), a matriz pode ser organizada da seguinte forma:

Uma imagem contendo Texto

Descrição gerada automaticamente

Balizadores para notas da Matriz BASICO

## 1.2 Solução

### 1.2.1 Objetivo SMART

Migrar da arquitetura monolítica para microsserviços e estudar a viabilidade do modelo arquitetural na nuvem AWS (S), até o final de 2024 (T) para facilitar a escalabilidade (R), reduzindo a indisponibilidade do sistema em 60% (M), com a contratação de especialistas em nuvem e o uso do Cloud Adoption Framework como base (A).

### 1.2.2 Premissas e Restrições

**Premissas:**

1. A migração deve ser feita em fases, onde cada fase corresponde à migração de um módulo do sistema para a arquitetura de microsserviços.
2. A migração para microsserviços deve focar em tornar os serviços independentes e desacoplados, de modo que facilite uma possível migração da infraestrutura para a AWS.
3. A migração para microsserviços deve ser conduzida com o mínimo possível de alterações nas regras de negócios, e deve nascer com a implementação de ferramentas de Logging.

**Restrições:**

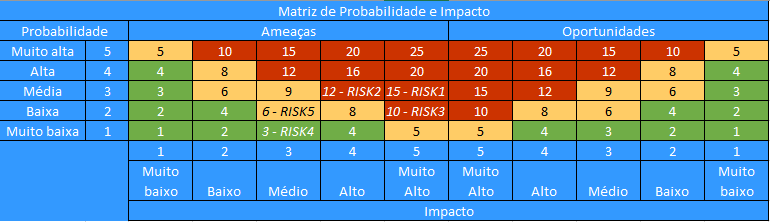
1. A migração para microsserviços não podem impactar o sistema de forma alguma.
2. Todo processo de migração deve passar por testes em um ambiente de homologação, de modo a garantir que não vai quebrar a jornada do cliente quando implementado em ambiente de produção.
3. A migração da solução deve ser finalizada até o final de 2024.
4. O custo operacional na nuvem dos primeiros 3 anos não pode ultrapassar a soma do custo de capital com o custo operacional dos últimos 3 anos.
5. O processo de migração não deve impactar os níveis de segurança ou compliance do sistema e cada fase de migração deve ser aprovada por uma equipe de governança e segurança da informação para garantir aderência a requisitos regulatórios e operacionais.
6. A configuração da infraestrutura deve cumprir com a regras de armazenamento e tratamento de dados sensíveis, onde aplicável.
7. Qualquer alteração no sistema deve manter a compatibilidade com sistemas de terceiros integrados ao sistema atual (como plataformas de pagamentos e de e-mail).

**Riscos do projeto:**

A partir das premissas e restrições, levantamos uma lista de itens que podem representar um risco para o andamento do projeto:

* Risco 001: Atraso na migração da arquitetura ou da solução em nuvem.
* Risco 002: O custo na nuvem ser maior que o custo On Premise.
* Risco 003: A migração causar impacto na segurança da informação, quebrar regras de compliance ou não cumprir com requisitos regulatórios.
* Risco 004: As ferramentas de terceiros não serem compatíveis ao novo sistema migrado para a nuvem.
* Risco 005: Falta de profissionais especialistas em computação em nuvem.





Matriz de Risco

### 1.2.3 Backlog de Produto

O Backlog de Produto desse projeto inclui todo o processo de mapear a aplicação monolítica atual, identificando seus serviços, integrações com ferramentas de terceiros, casos de uso, modelagem do banco de dados, etc. A partir disso, será desenhada a nova arquitetura da aplicação, separada em microsserviços, modelagem do novo banco de dados, e em seguida será executada essa migração, desenvolvendo os microsserviços. Por último, teremos o processo de desenhar a nova infraestrutura em Cloud.

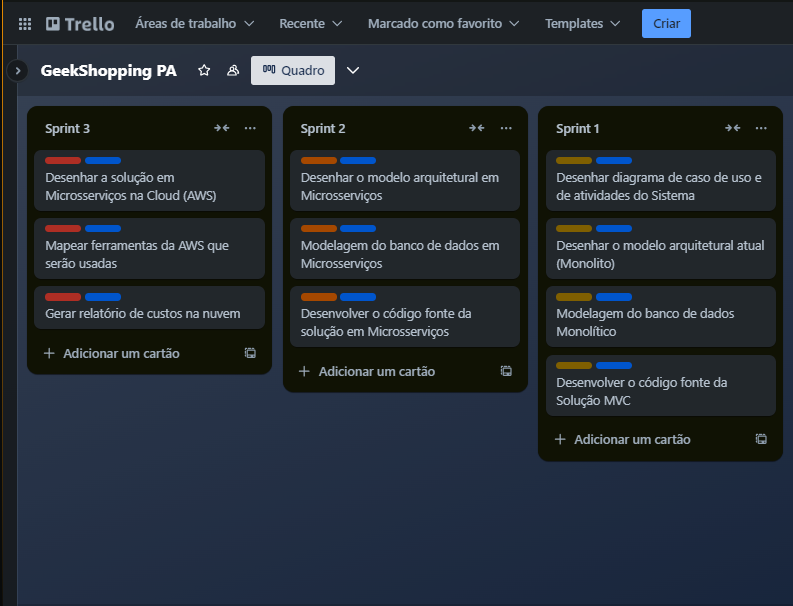


# 2. Área de Experimentação

## 2.1 Sprint 1

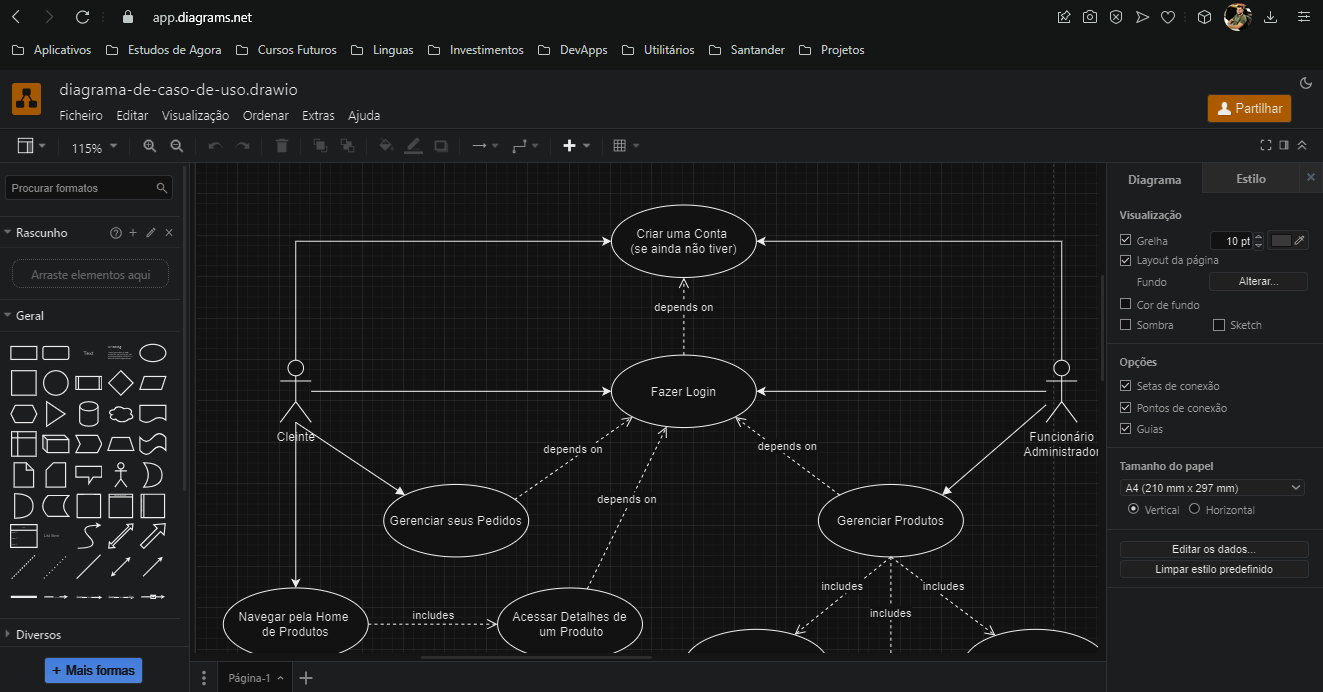
### 2.1.1 Solução

#### Evidência do planejamento:

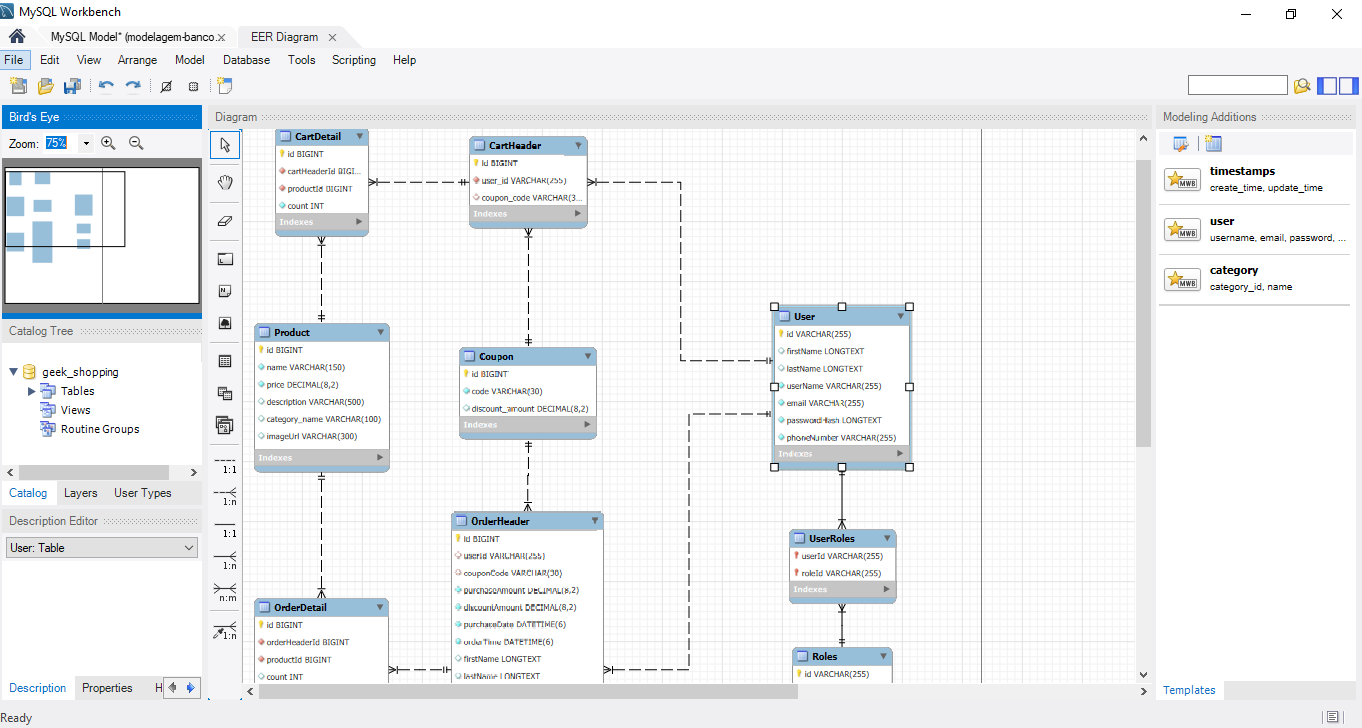


#### Evidência da execução de cada requisito:

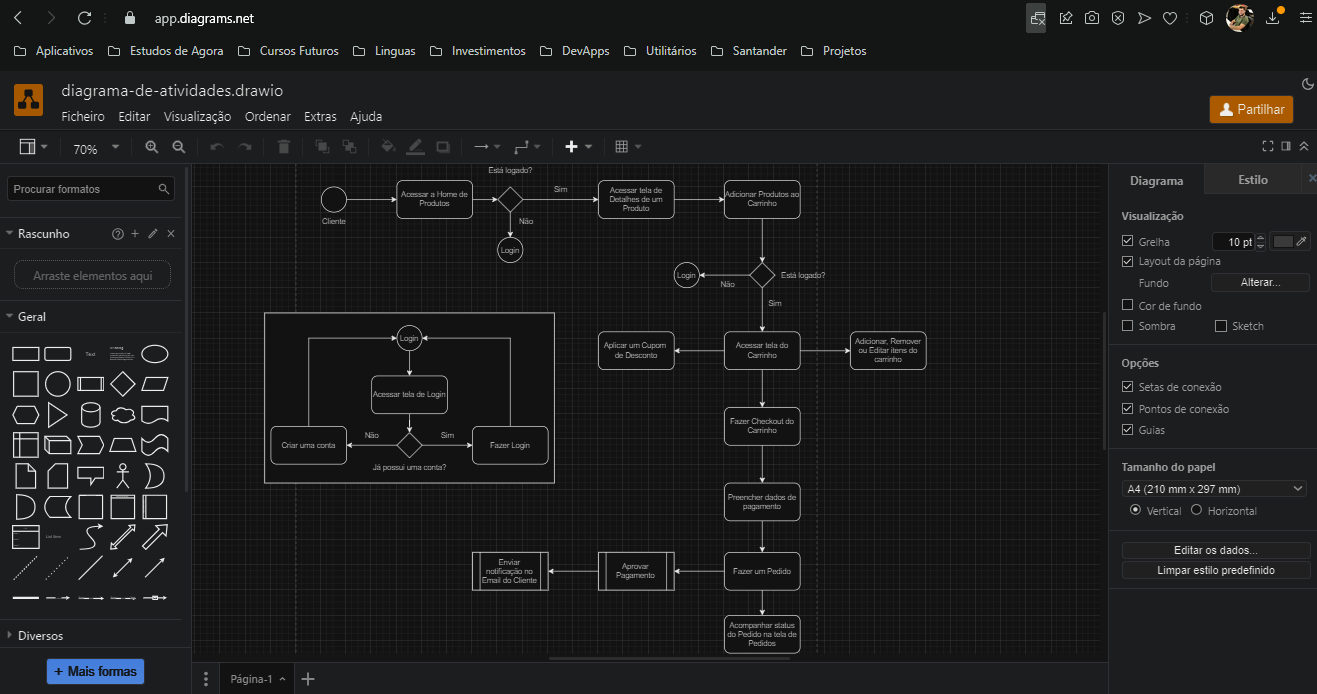
Como evidência da execução de cada requisito, estão disponíveis todos os arquivos gerados, via Draw.io e MySQL Workbench, no Github <https://github.com/PedroPadilhaPortella/GeekShopping_PA/tree/main/sprint-1>



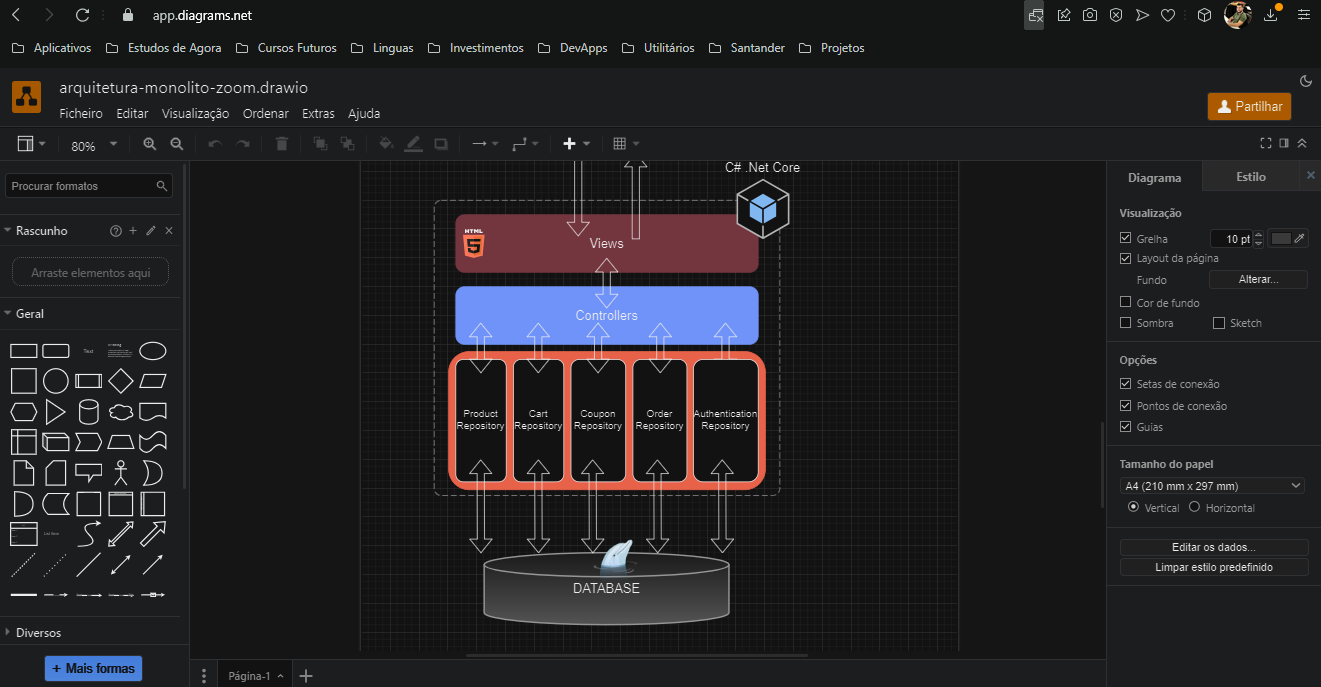
Evidencia de execução do diagrama de caso de uso, disponível em: <https://github.com/PedroPadilhaPortella/GeekShopping_PA/blob/main/sprint-1/diagrama-de-caso-de-uso.drawio>



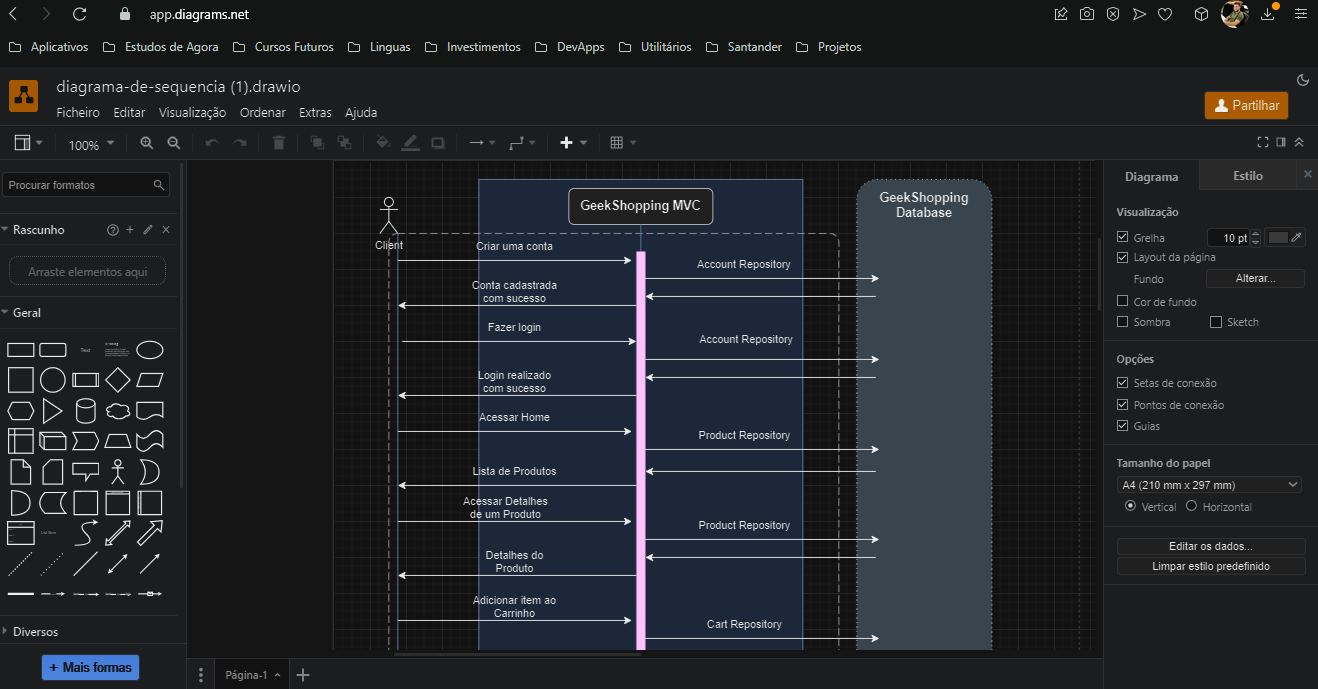
Evidencia de execução da modelagem do banco de dados, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/modelagem-banco-de-dados.mwb



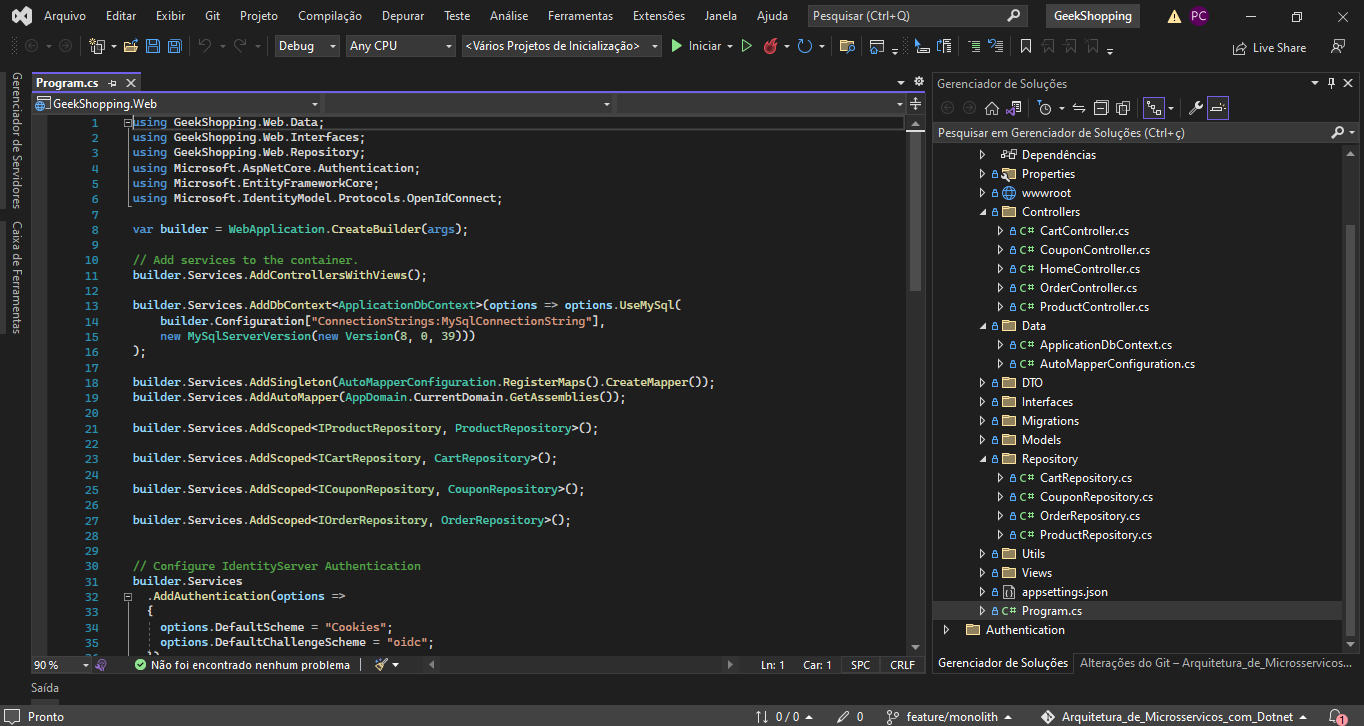
Evidencia de execução do diagrama de atividades, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/diagrama-de-atividades.drawio



Evidencia de execução do diagrama arquitetural , disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/arquitetura-monolito-zoom.drawio



Evidencia de execução do diagrama de sequência, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/diagrama-de-sequencia.drawio



Evidencia de desenvolvimento da POC, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping/tree/feature/monolith

#### Evidência dos resultados:

Para melhor visualização das atividades, casos de uso e a arquitetura do projeto, foram desenhados alguns diagramas, e, junto com eles, será disponibilizado o link para os diagramas salvos no Github, acompanhados do arquivo Draw.io gerado:

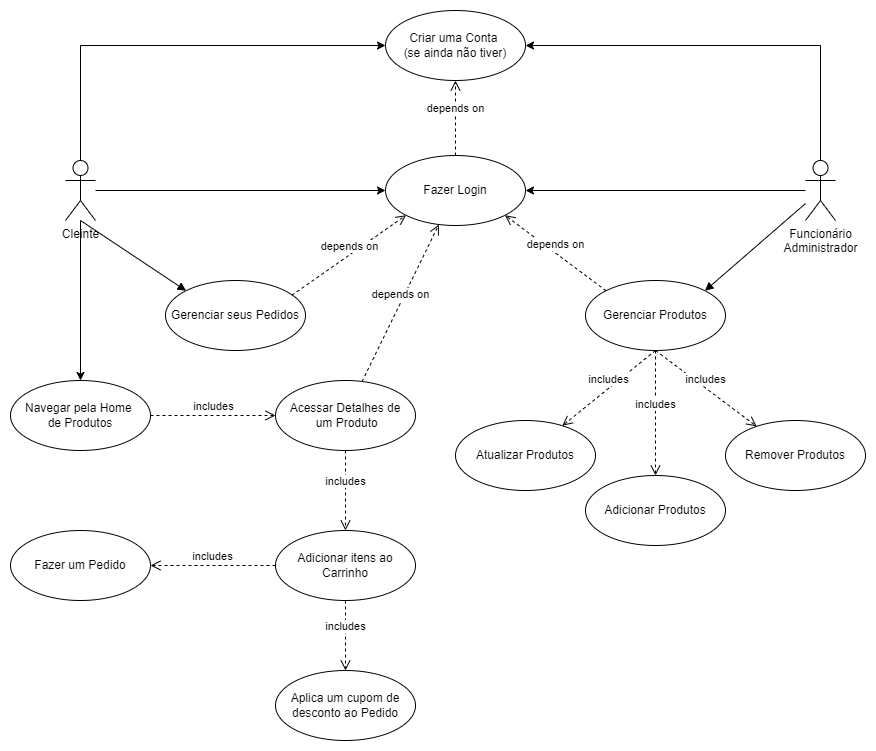


Diagrama de caso de uso do Geek Shopping E-commerce, demonstrando quais atividades podem ser realizadas pelo Cliente e pelo Administrador do Sistema, incluindo quais atividades são dependentes de outras, disponível em <https://github.com/PedroPadilhaPortella/GeekShopping_PA/blob/main/sprint-1/diagrama-de-caso-de-uso.png>

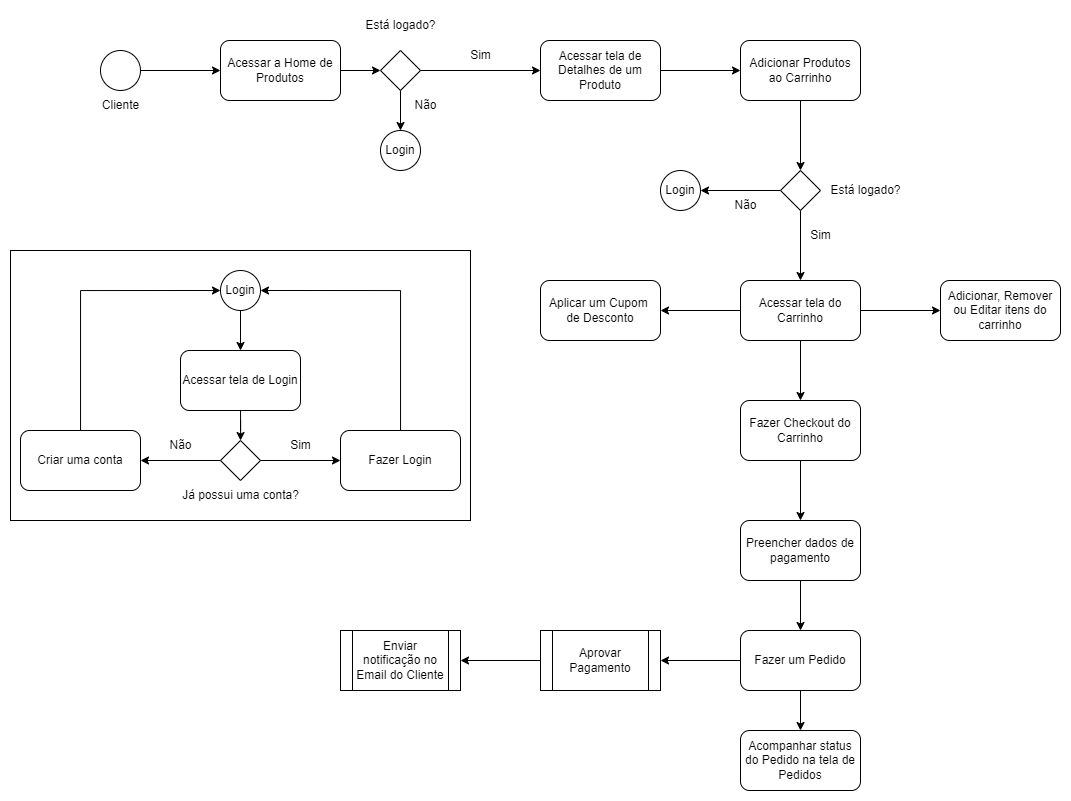


Diagrama de atividades do Geek Shopping E-commerce, apresentando todo o fluxo, desde fazer o login em sua conta, até adicionar produtos ao carrinho, fazer um pedido e acompanhar esse pedido, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/diagrama-de-atividades.png

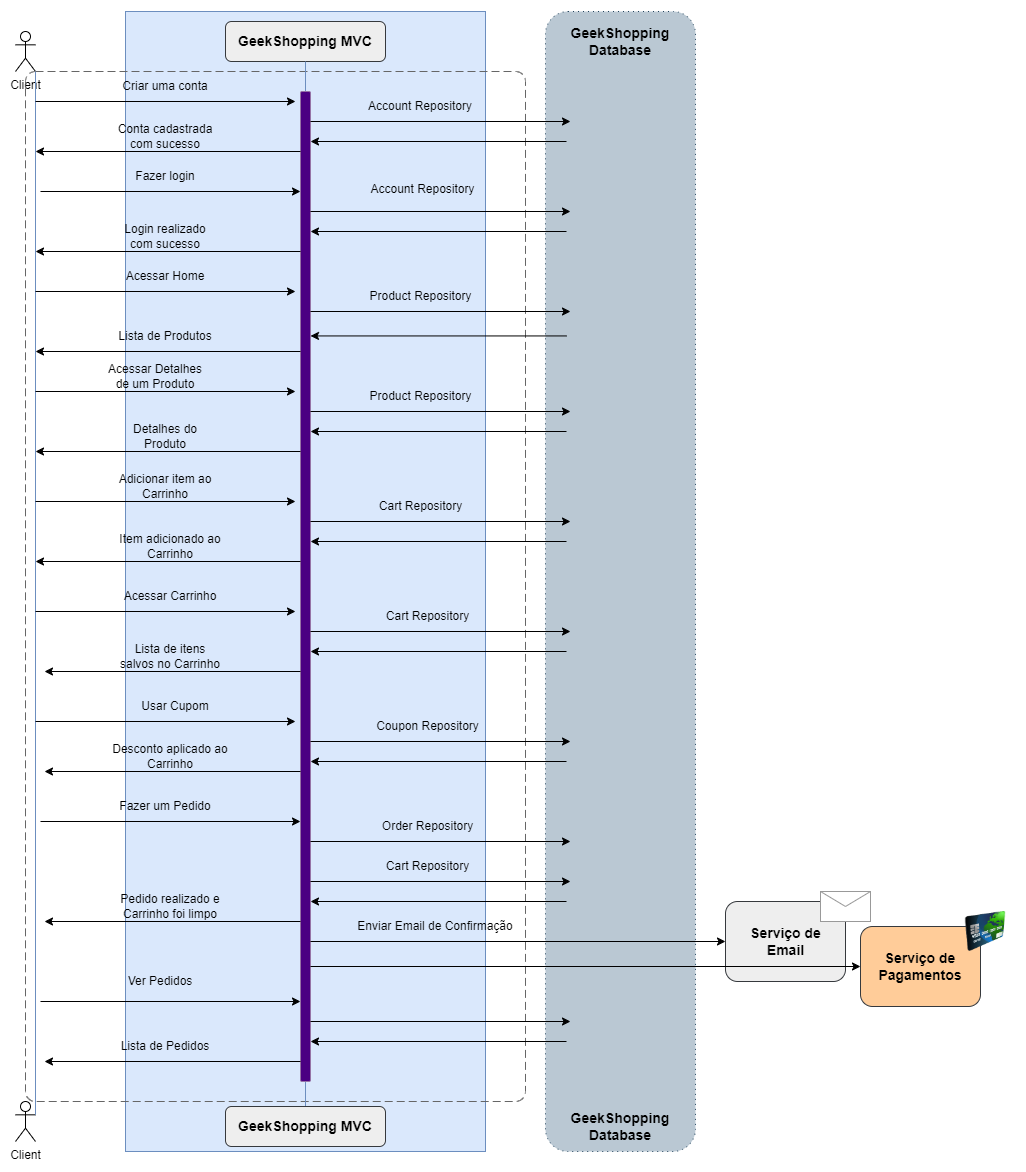
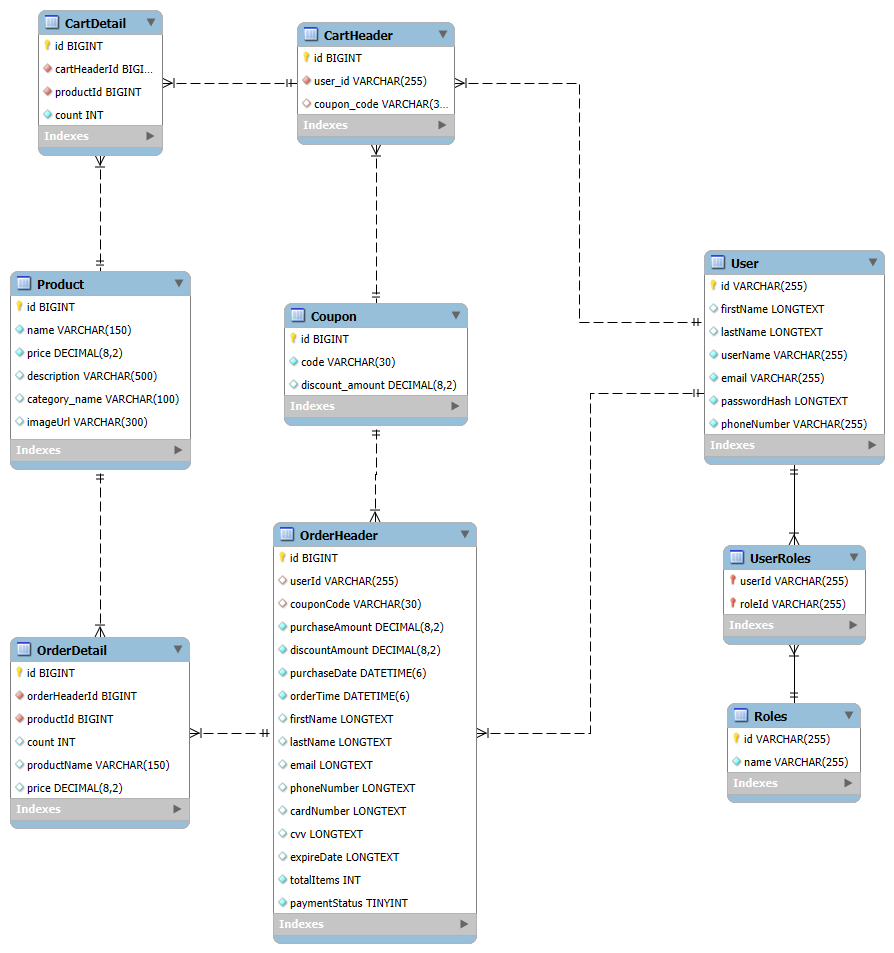
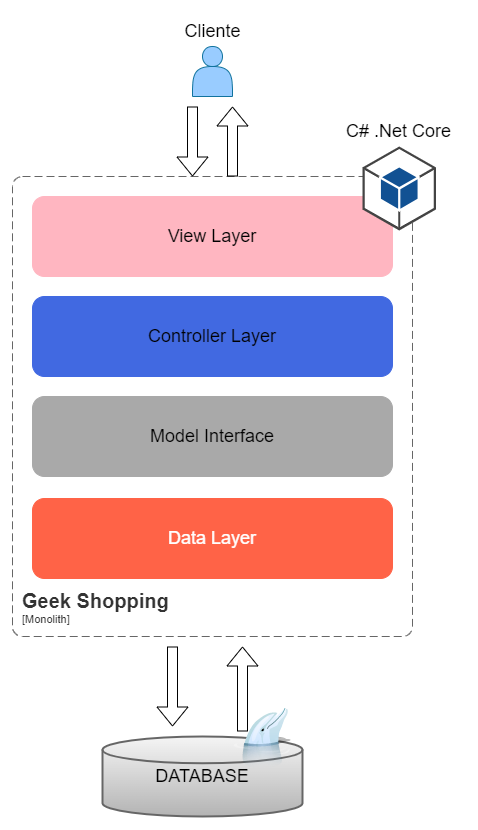


Diagrama de sequência do Geek Shopping E-commerce, apresentando toda a sequência da aplicação, desde fazer o login em sua conta, até adicionar produtos ao carrinho, fazer um pedido e acompanhar esse pedido, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/diagrama-de-sequencia.png



Modelagem do banco de dados do Geek Shopping E-commerce, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/modelagem-banco-de-dados.png

O sistema está construído em um modelo MVC (Model-View-Controller), utilizando a tecnologia .Net Core, esse sistema está integrado a um banco de dados MySQL. Importante ressaltar que o banco de dados do sistema é gerado dinamicamente, utilizando o Entity Framework.

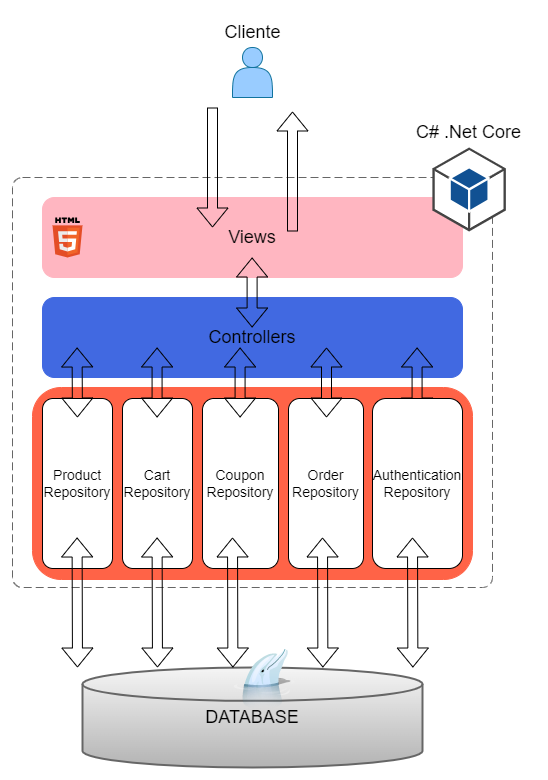


Visão macro da Arquitetura MVC do sistema, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-1/arquitetura-monolito.png

Atualmente o sistema está separado em alguns serviços, que posteriormente serão migrados, cada um, para um microsserviço:

* Serviço de Produtos
* Serviço de Carrinho
* Serviço de Cupom
* Serviço de Pedidos
* Serviço de Autenticação

Os serviços são usados para trazer os dados do banco de dados e entregá-los ao Controller, que renderiza eles na View para o cliente.



Visão micro da Arquitetura MVC do sistema, apresentando os serviços integrados, que são o serviço de Produtos, Carrinho, Cupom, Pedidos e Autenticação, disponível em <https://github.com/PedroPadilhaPortella/GeekShopping_PA/blob/main/sprint-1/arquitetura-monolito-zoom.png>

Para validar os itens desenhados acima, foi desenvolvida uma POC bem simples, com as funcionalidades de:

* Gerenciar produtos
* Adicionar produtos ao carrinho
* Utilizar um cupom de desconto
* Fazer um pedido
* Ver meus pedidos
* Criar uma conta ou fazer login

O código fonte da POC pode ser encontrado no Github, na Branch *feature/monolith*:

<https://github.com/PedroPadilhaPortella/GeekShopping/tree/feature/monolith>

E as imagens das telas podem ser encontradas no Github, em:

<https://github.com/PedroPadilhaPortella/GeekShopping/tree/feature/monolith/screens>

### 2.1.2 Lições Aprendidas

* Não estava planejado, mas achei necessário adicionar mais diagramas, como o diagrama de atividades e o diagrama de sequência, para ter melhor visibilidade do fluxo do sistema, e poder desenhar de forma mais assertiva, a arquitetura.
* Tentei usar o Asta UML para fazer os diagramas, mas não consegui baixar a versão Community, então achei melhor fazer no Draw.io mesmo, e foi até mais simples.
* O desenvolvimento de uma POC foi muito útil, pois me deu uma visão melhor de como separar os serviços, e deu uma visão muito boa de quais funcionalidades podem ser extraídas para cada microsserviço no futuro.
* Para configurar a parte de login e cadastro de forma mais rápida, eu utilizei uma biblioteca chamada Duende.IdentityServer.AspNetIdentity, que já a parte de gerenciamento de usuários, cadastro e login, utilizando o IdentityServer.

## 2.2 Sprint 2

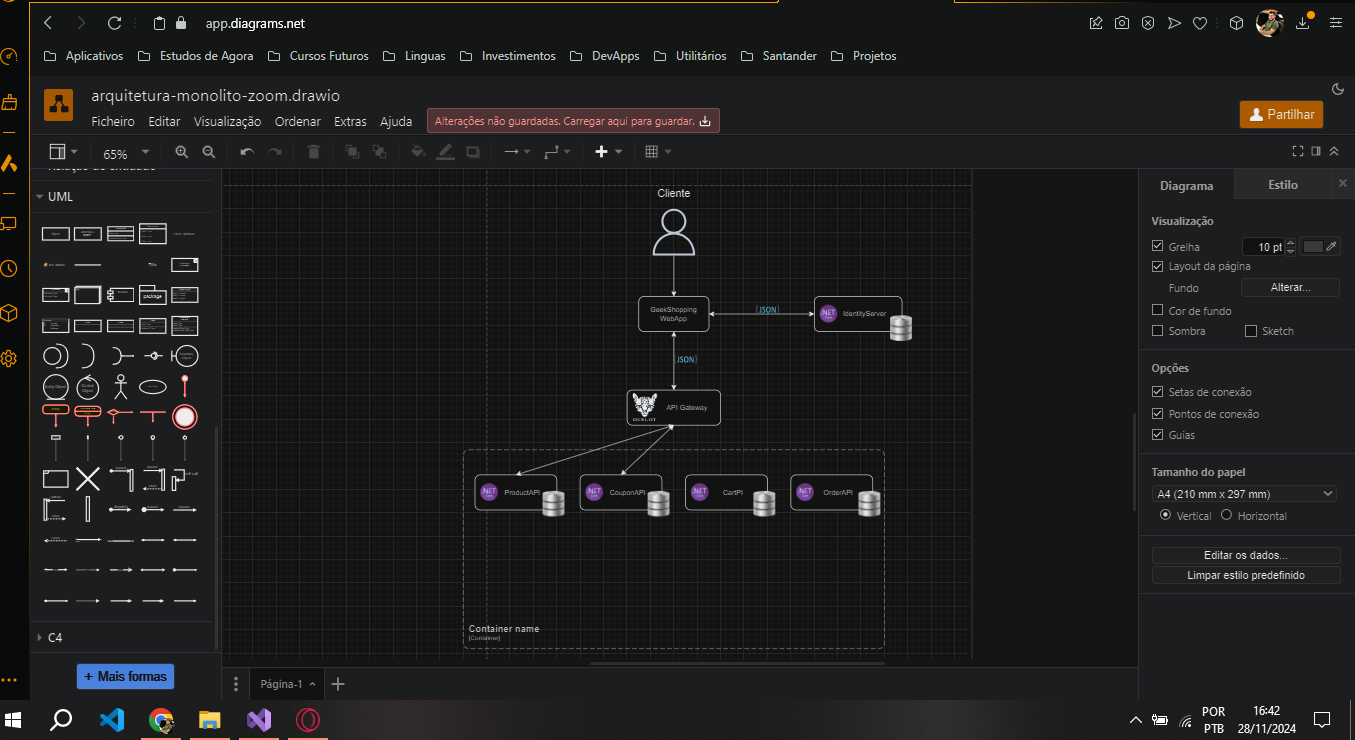
### 2.2.1 Solução

#### Evidência do planejamento:

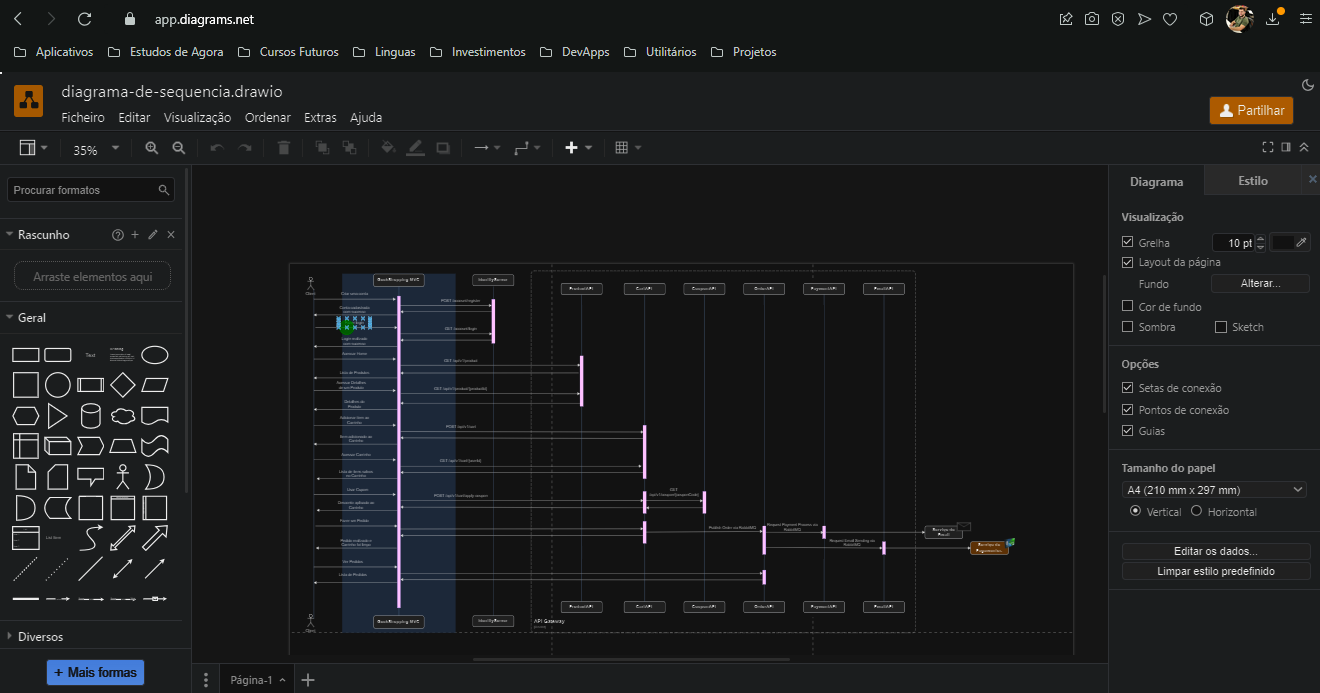


#### Evidência da execução de cada requisito:

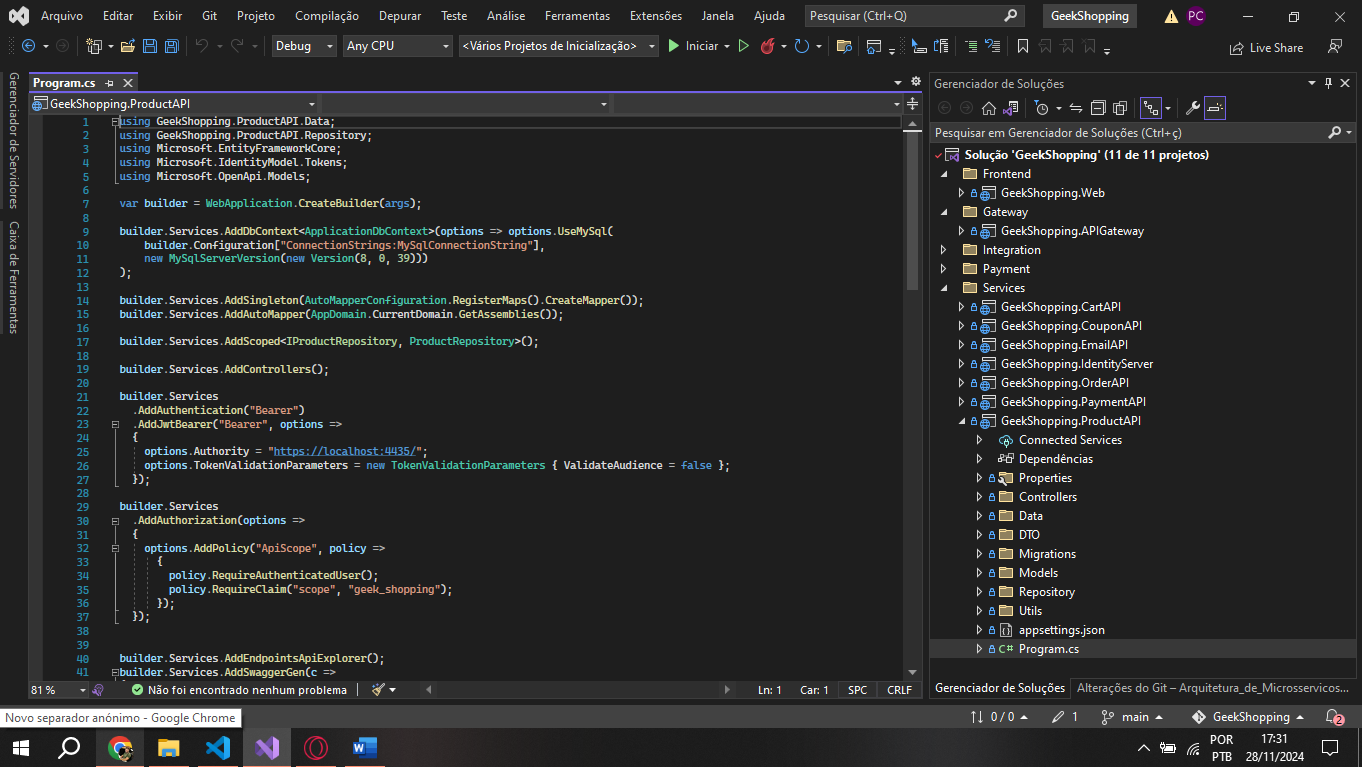
Como evidência da execução de cada requisito, estão disponíveis todos os arquivos gerados, via Draw.io e MySQL Workbench, no Github https://github.com/PedroPadilhaPortella/GeekShopping\_PA/tree/main/sprint-2



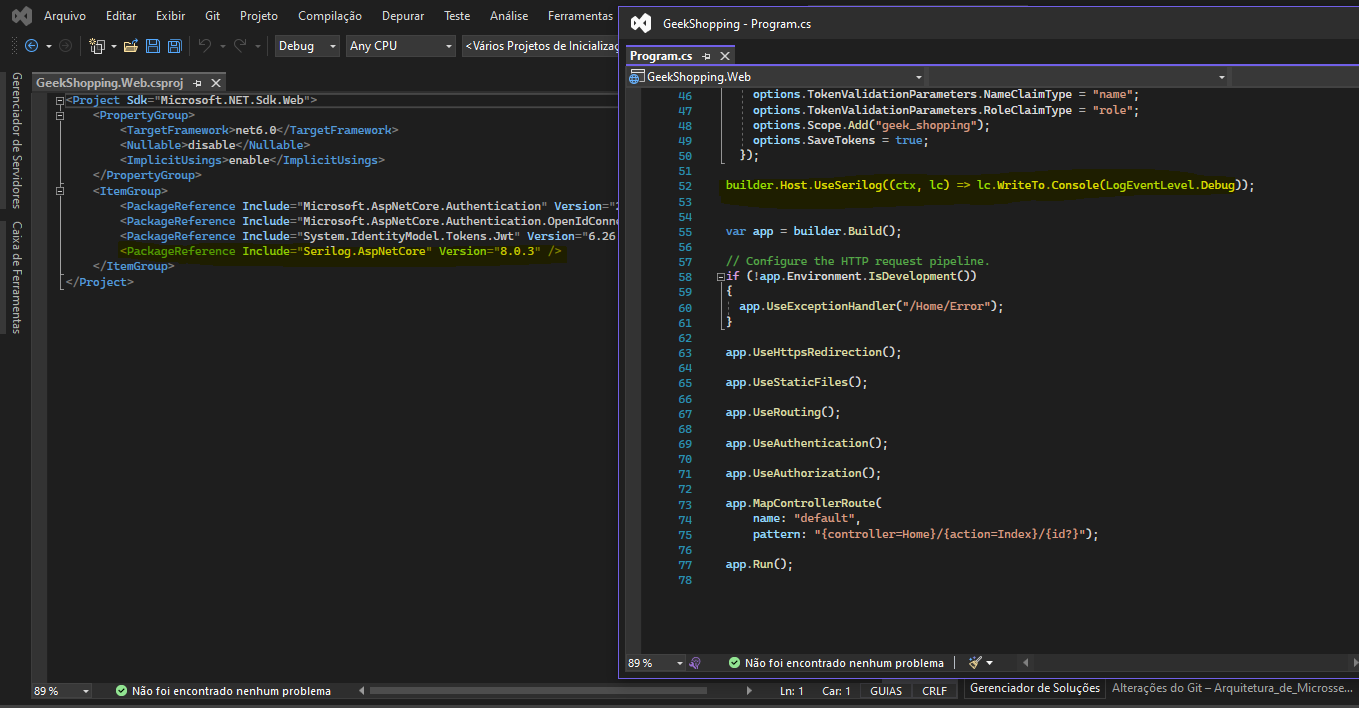
Evidência de execução do diagrama arquitetural, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-2/arquitetura-microsservicos.drawio



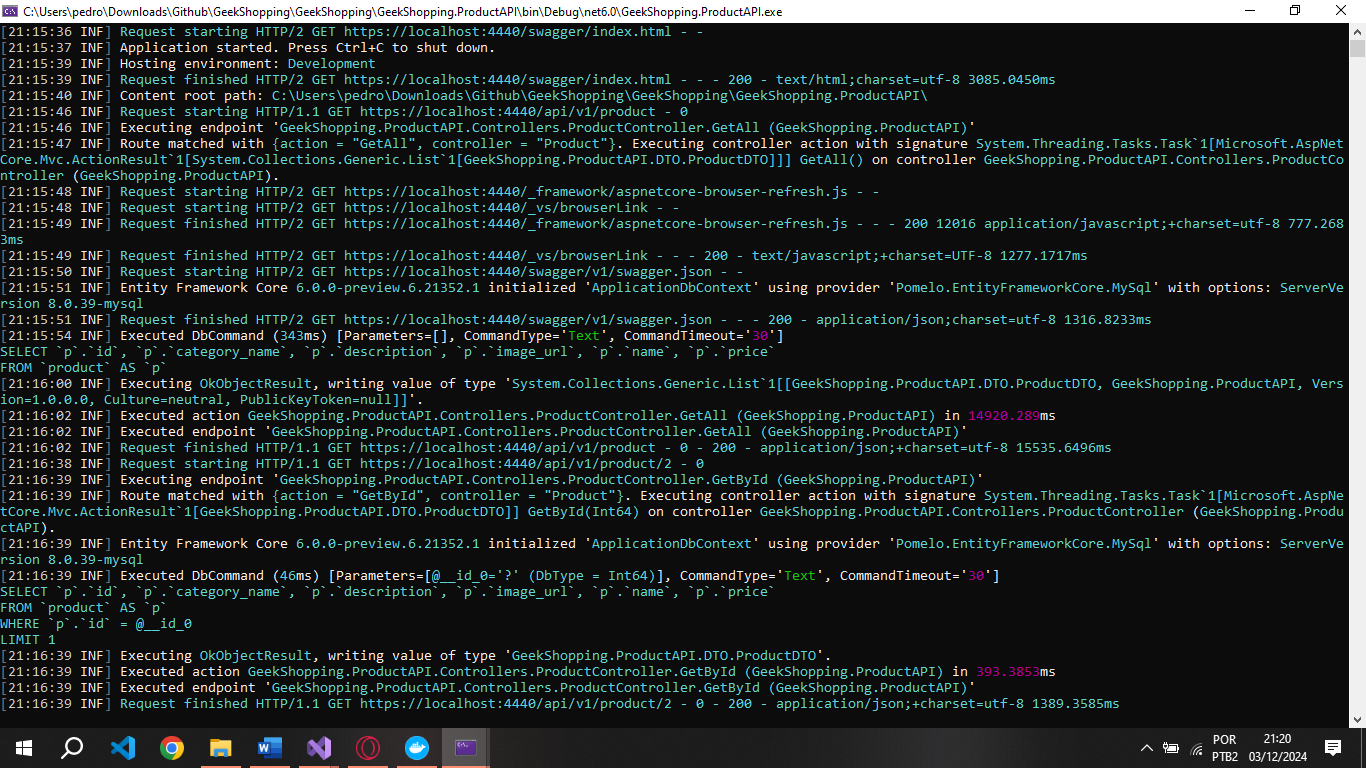
Evidência de execução do diagrama de sequência, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-2/diagrama-de-sequencia.drawio



Evidencia de desenvolvimento da migração para microsserviços, disponível em: https://github.com/PedroPadilhaPortella/GeekShopping/tree/feature/microsservices

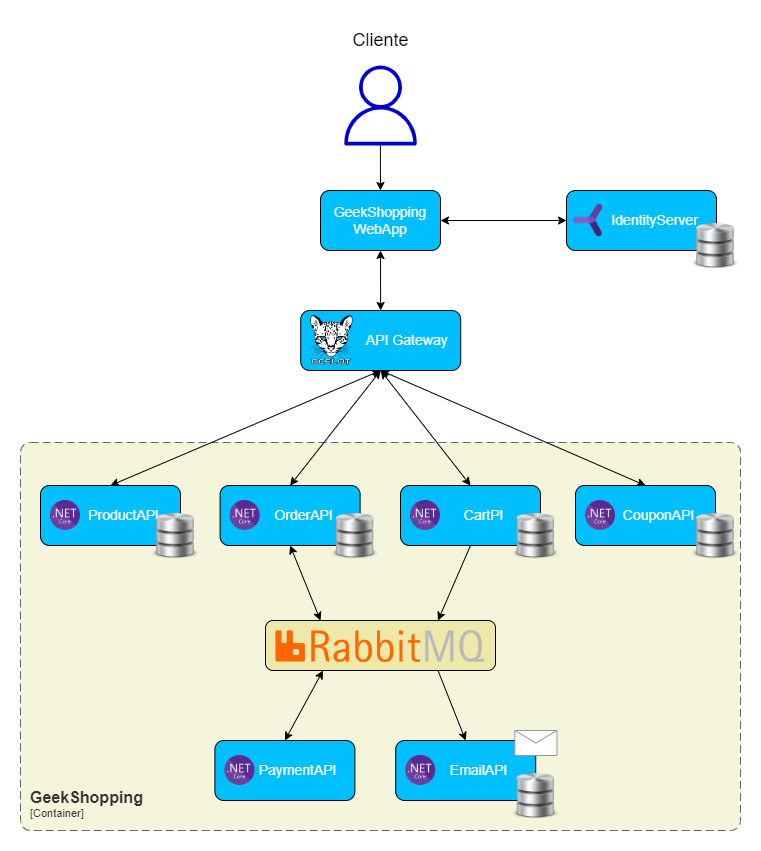


Evidencia da implementação de uma ferramenta de Logging, no caso, o Serilog.



Evidencia da implementação do Serilog no projeto, na imagem acima, os logs da Api de Produtos.

#### Evidência dos resultados:



Visão da Arquitetura em Microsserviços do sistema, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-2/arquitetura-microsservicos.png

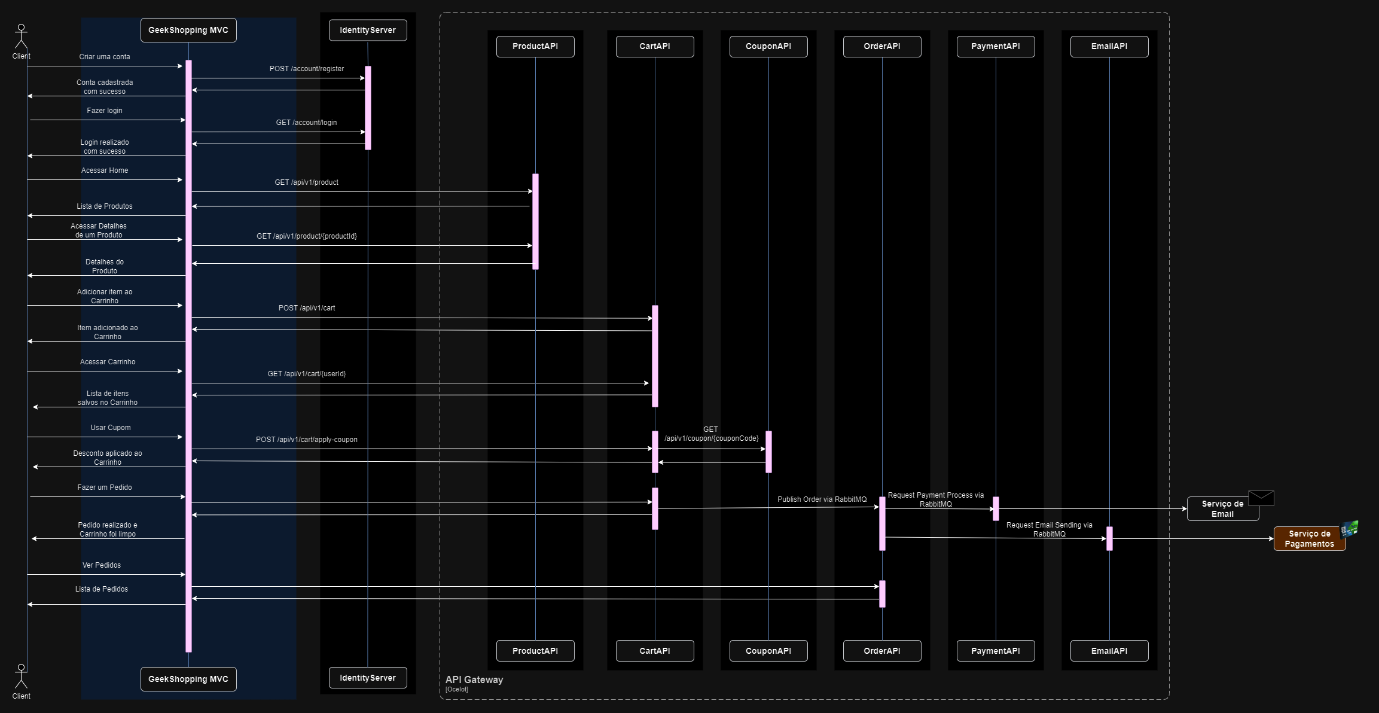


Diagrama de sequência do Geek Shopping E-commerce, versão em microsserviços, apresentando toda a sequência da aplicação, desde fazer o login em sua conta, até adicionar produtos ao carrinho, fazer um pedido e acompanhar esse pedido, disponível em https://github.com/PedroPadilhaPortella/GeekShopping\_PA/blob/main/sprint-2/diagrama-de-sequencia.png

Para validar os itens desenhados acima, foi utilizada a POC desenvolvida anteriormente, onde as funcionalidades do sistema foram quebradas em micro serviços, com as funcionalidades de:

* Product API (Gerenciar produtos do sistema)
* Cart API (Gerenciar produtos do carrinho de cada cliente)
* Coupon API (Gerenciar o uso de cupons de desconto)
* Order API (Fazer um pedido e gerenciar pedidos feitos)
* Identity Server WebApp (Manter a lógica de criar contas e login separadas)

Não houve nenhuma alteração significativa da parte de UI, nem nas tabelas do banco de dados.

O código fonte pode ser encontrado no Github, na Branch *feature/microsservices*:

https://github.com/PedroPadilhaPortella/GeekShopping/tree/feature/microsservices

### 2.2.2 Lições Aprendidas

* Foi bem simples configurar o Serilog como biblioteca de Logging, ele funcionou muito bem o os serviços desenvolvidos em .Net Core.
* Achei interessante utilizar o RabbitMQ para o envio mensagens assíncronas que poderiam demorar para trazer uma resposta, como os serviços de Pedidos, Pagamentos e Email, que são serviços que podem demorar para executar suas operações, já que lidam com serviços de terceiros.
* Utilizei o RabbitMQ já pelo Docker, o que, no futuro, até facilitaria a implantação dele na nuvem.
* Utilizei o MailTrap como servidor de emails de teste, já que a maioria das outras soluções que eu encontrei eram pagas.
* Não foi necessário fazer a modelagem do banco de dados novo, pois os campos não foram alterados, nem a relação entre as tabelas.

## 2.3 Sprint 3

### 2.3.1 Solução

#### Evidência do planejamento:

#### Evidência da execução de cada requisito:

#### Evidência dos resultados:

### 2.3.2 Lições Aprendidas

# 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.