**UNIVERSIDADE FEDERAL DA PARAÍBA**
**CENTRO DE INFORMÁTICA**
**CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO**

**ELAINE CRISTINA LINS DE SOUZA**

# AUTOMATIZAÇÃO DE OPERAÇÕES NA NUVEM SOB A ÓTICA DE SRE: UM ESTUDO DE CASO DO SLACKBOT HARRY BOTTER

**JOÃO PESSOA**
**2024**

ELAINE CRISTINA LINS DE SOUZA

# AUTOMATIZAÇÃO DE OPERAÇÕES NA NUVEM SOB A ÓTICA DE SRE: UM ESTUDO DE CASO DO SLACKBOT HARRY BOTTER

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia de Computação da Universidade Federal da Paraíba como requisito complementar para obtenção do título de Bacharel em Engenharia de Computação, sob orientação do professor Ms. Derzu Omaia.

João Pessoa
2024

**Catalogação na publicação Seção de Catalogação e Classificação**

S729a Souza, Elaine Cristina Lins de.
Automatização de operações na nuvem sob a ótica de SRE: um estudo de caso do slackbot Harry Botter / Elaine Cristina Lins de Souza. - João Pessoa, 2024.
65 f.

Orientação: Derzu Omaia.
TCC (Graduação) -
UFPB/CI.

1. Automação. 2. Devops. 3. SlackBot. 4. C I. Omaia, Derzu. II. Título.

UFPB /CI

CDU 004.8

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

ELAINE CRISTINA LINS DE SOUZA

**AUTOMATIZAÇÃO DE OPERAÇÕES NA NUVEM SOB A ÓTICA DE SRE: UM ESTUDO DE CASO DO SLACKBOT HARRY BOTTER**

Trabalho de conclusão de curso submetido à Banca Examinadora designada pelo Curso de Graduação em Engenharia de Computação da Universidade Federal da Paraíba como requisito para obtenção do grau de Bacharel em Engenharia de Computação.

BANCA EXAMINADORA

Assinatura: _____
Prof. Derzu Omaia
(Orientador)

Assinatura: _____
Prof. Rostand Edson Oliveira Costa

Assinatura: _____
Prof. Tiago Maritan Ugulino de Araújo

João Pessoa, 15 de Maio de 2024.

Dedico aos meus pais a realização desse sonho, pois me deram a vida, alimentaram meus sonhos, e trabalharam muito para que eu pudesse realizá-lo. Dedico também ao professor Derzu Omaia apoio e incentivo para a realização deste trabalho.

# AGRADECIMENTOS

Gostaria de expressar minha gratidão à Universidade Federal da Paraíba por me proporcionar recursos necessários para que eu pudesse estudar com qualidade e assim fazer minha pesquisa. Ao curso de Engenharia de Computação, juntamente com seus professores (as) e toda sua coordenação.

Ao Prof. Derzu Omaia, em especial, por aceitar ser meu orientador, além da atenção, paciência e pelos conhecimentos transmitidos para que eu pudesse realizar meu trabalho.

Aos meus pais Edson José de Souza e Gilvania Lins de Souza, por sempre apoiarem meus sonhos e por lutarem para que eu pudesse realizá-lo, além de terem me proporcionado força, amor, respeito e incentivo.

Aos meus amigos Letícia Leite e Stênio Ellison, por estar sempre ao meu lado, me incentivando e ajudando a concluir minha pesquisa.

E por fim a todas as pessoas que me auxiliaram a concluir essa etapa da minha vida.

"Quando o homem compreende a sua realidade, pode levantar hipóteses sobre o desafio dessa realidade e procurar soluções. Assim, pode transformá-la e o seu trabalho pode criar um mundo próprio, seu eu e as suas circunstâncias".

Paulo Freire
*Educação e mudança*

# RESUMO

O presente estudo trata sobre automatização de operações na nuvem sob a ótica de SRE (*Site Reliability Engineering*): Um estudo de caso do Slackbot Harry Botter. Tendo em vista que o tema, é fundamental para o curso de engenharia da computação porque oferece a oportunidade de estudar e entender como o desenvolvimento de software, a infraestrutura de TI e as práticas de gestão de serviços se relacionam. A fim de avaliar os benefícios obtidos na implementação de estratégias de automatização de operações na nuvem, através de um estudo de caso com um Slackbot em uma empresa de tecnologia. Para tanto, foi necessário: Identificar e descrever as práticas específicas de SRE utilizadas na automatização de operações na nuvem, destacando suas características e benefícios; Avaliar quantitativamente os benefícios obtidos com a implementação das práticas de SRE na automatização de operações na nuvem; Analisar os desafios organizacionais e culturais enfrentados durante a disponibilização do bot.

Realizou-se, então, uma pesquisa de natureza aplicada, com abordagens quantitativas, que se deram através de uma revisão bibliográfica, implementando técnicas de SRE para a construção de um bot, e após isso foi analisados os resultados da disponibilização do mesmo para a empresa. Diante disso, verificou-se que houve uma boa aderência do bot pela empresa, o que reduziu o tempo para realizar algumas operações na nuvem, criou uma nova padronização dessas ações, reduzindo assim erros humanos, e o custo operacional.

Sendo assim, foi possível concluir que o bot ajuda a melhorar nossa compreensão de como a utilização de chatbots podem ser úteis e garantir maior eficiência no dia-a-dia de uma empresa. Uma vez que, os resultados mostram que os bots não apenas podem resolver problemas de padronização e melhorar a eficiência das operações, mas também podem dar mais autonomia a equipes que não têm muita experiência com infraestrutura em nuvem.

**Palavras-chave**: Automação, Devops, SRE, SlackBot, ChatBot.

# ABSTRACT

This study deals with automating operations in the cloud from the perspective of SRE (Site Reliability Engineering): A case study of the Harry Botter Slackbot. Considering that the topic is fundamental for the computer engineering course because it offers the opportunity to study and understand how software development, IT infrastructure and service management practices are related. In order to evaluate the benefits obtained from implementing operations automation strategies in the cloud, through a case study with a Slackbot in a technology company. To this end, it was necessary to: Identify and describe the specific SRE practices used in automating cloud operations, highlighting their characteristics and benefits; Quantitatively evaluate the benefits obtained from implementing SRE practices in automating cloud operations; Analyze the organizational and cultural challenges faced when making the bot available.

An applied research was then carried out, with quantitative approaches, which took place through a bibliographical review, implementing SRE techniques for the construction of a bot, and after that the results of making it available to the company were analyzed. . Given this, the recipe was that there was a good adherence of the bot by the company, which affected the time to carry out some operations in the cloud, created a new standardization of these actions, thus reducing human errors and operational costs.

Therefore, it was possible to conclude that the bot helps to improve our understanding of how the use of chatbots can be useful and ensure greater efficiency in a company's day-to-day operations. Since, the results show that bots can not only solve standardization problems and improve the efficiency of operations, but they can also give more autonomy to teams that do not have much experience with cloud infrastructure.

**Keywords**: Automation, Devops, SRE, SlackBot, ChatBot.

# LISTA DE TABELAS E FIGURAS

# LISTA DE SIGLAS E ABREVIATURAS

**AWS –** Amazon Web Services

**BaaS** - Banking as a Service

**EKS** - Elastic Kubernetes Service

**IAC** - Infrastructure as a Code

**IaaS** - Infrastructure as Service

**K8S** - Kubernetes

**PaaS** - Platform as a Service

**S3** - Simple Storage Service

**SaaS** - Software as a Service

**SQS** - Simple Queue Service

**SRE –** Site Reliability Engineering

**SSO** - Single Sign-On

# SUMÁRIO

# 1. INTRODUÇÃO

As novas tecnologias têm um impacto na vida cotidiana da sociedade moderna, essas influenciam direta e indiretamente nos diversos aspectos que moldam o cotidiano de uma sociedade globalizada. Segundo Leandro (2007), “a utilização da internet se tornou indispensável nos dias de hoje e, além disso, esta utilização está tendo um vasto crescimento”. Uma vez que, “a internet é um meio de comunicação que permite, pela primeira vez, a comunicação de muitos para muitos em tempo escolhido e a uma escala global” (CASTELLS, 2003, p. 16). Esse processo reflete na inclusão de aparelhos eletrônicos na rotina do ser humano, já que a popularização das plataformas móveis como smartphones e tablets, adicionou um novo espaço de possibilidade para o uso da internet, ampliando sua presença em sociedade, nos mais diversos âmbitos do cotidiano, tais como: compras, serviços bancários, educação e até mesmo acesso a documentos pessoais.

Esse grande avanço na tecnologia, impacta diretamente as formas com que os softwares e sites são desenvolvidos. Esses, agora precisam garantir uma melhor escalabilidade, confiabilidade, segurança e otimização, e tudo isso, deve ser desenvolvido numa menor quantidade de tempo, e com um menor custo operacional. Favorecendo isso, ocorreu o surgimento de novas tecnologias, como, por exemplo, Kubernetes, Computação em Nuvens, dentre outras. Logo,“ o software desenvolvido com a expectativa de ser utilizado a longo prazo, tem seu ciclo de vida e eficiência rapidamente reduzidos, devido a desatualização de ferramentas, dependências, práticas, servidores, entre outros” (ARAÚJO E NASCIMENTO, 2022, p. 19).

Sendo assim, é nesse cenário, que o profissional de SRE (*Site Reliability Engineering*) deve atuar, visto que esse profissional, tem como principal objetivo fazer com que sites funcionem de maneira suave, eficiente e confiável, por meio da adoção de cultura e de práticas de engenharia no mundo de infraestrutura e operações (MULLER, 2023). Isso inclui a implementação de estratégias de automação, monitoramento contínuo, gestão de incidentes e a busca constante pela melhoria da escalabilidade, confiabilidade e segurança dos sistemas em um ambiente dinâmico e globalizado. Assim, o profissional de SRE desempenha um papel estratégico na adaptação e evolução dos sistemas digitais frente aos novos desafios tecnológicos decorrentes das grandes e demandas da sociedade contemporânea.

Visando abordar a problemática sobre, como a implementação de práticas de SRE,

aliadas à automatização de operações na nuvem, influenciam a eficiência e a confiabilidade das operações de TI, esse trabalho justifica-se pela relevância social na atualidade, especialmente no contexto da transformação digital e da crescente dependência de serviços online. Com a expansão da utilização de sites e a demanda por sistemas mais ágeis e confiáveis, a eficiência das operações de TI tornou-se crucial para empresas de todos os portes e setores. A implementação eficaz de práticas de SRE na automação de operações na nuvem não apenas impacta diretamente a qualidade dos serviços prestados, mas também influencia a competitividade das organizações no mercado global. Possuindo foco total na experiência do cliente através da estabilidade do produto e do ambiente, procurando antecipar possíveis problemas de instabilidade e indisponibilidade na produção (MUNIZ, OLIVEIRA e MULLER, 2023).

Este tema é essencial para o curso de Engenharia da Computação, pois possibilita a análise da interação entre desenvolvimento de software, infraestrutura de TI na nuvem e práticas de gestão de serviços. A pesquisa nesse campo proporciona noções valiosas sobre como as práticas de SRE podem aprimorar a eficiência e confiabilidade das operações de computação na nuvem, resultando em aplicações mais práticas de desenvolver e, posteriormente, mais seguras e eficientes. Além disso, o estudo de caso do Slackbot, Harry Botter, amplia a compreensão sobre a aplicação prática dos conceitos de SRE em cenários reais.

Durante o trabalho como SRE em uma empresa de BaaS, desenvolvendo automações e auxiliando os times de desenvolvimento de sites com todos os problemas de implementação e manutenção de infraestrutura, e com o intuito de facilitar um pouco mais esse trabalho, foi desenvolvido, o Harry Botter. Isso me ajudou a decidir esse tema. Sendo assim, tive a oportunidade de vivenciar os benefícios e os problemas da automatização de operações na nuvem a partir do ponto de vista de SRE. Ao compreender a complexidade e a necessidade desse assunto, percebeu-se a importância dessas práticas para garantir a eficiência e confiabilidade dos serviços em um ambiente extremamente dinâmico e exigente. Logo, decidiu-se por este tema porque tinha a oportunidade de aplicar os conhecimentos teóricos na prática, bem como para exaltar a necessidade de aprimorar continuamente os processos operacionais.

Diante disso, o objetivo dessa pesquisa é avaliar, sob a perspectiva de SRE, os benefícios obtidos na implementação de estratégias de automatização de operações na nuvem, mediante um estudo de caso com um Slackbot em uma empresa de tecnologia. De forma mais

específica, buscou-se:

- Identificar e descrever as práticas específicas de SRE utilizadas no desenvolvimento de um bot, destacando suas características e benefícios.
- Avaliar quantitativamente os benefícios obtidos com a implementação das práticas de SRE na automatização de operações na nuvem na empresa a qual o bot foi disponibilizado.
- Analisar os desafios organizacionais enfrentados durante a disponibilização do bot.

A abordagem utilizada consistiu em uma pesquisa básica, quantitativa e exploratória, além de uma revisão de estudos de caso e uma revisão de literatura. Apresentando uma visão geral sobre automatização de operações na nuvem sob a visão de SRE.

# 2. REFERENCIAL TEÓRICO

Esse capítulo apresenta toda a fundamentação teórica necessária para o embasamento deste trabalho, possuindo, conceitos e definições importantes para a compreensão do tema deste projeto.

## 2.1 Site Reliability Engineering - SRE

O *Site Reliability Engineering* (SRE), ou traduzido, Engenharia de Confiabilidade de Sites, é uma abordagem desenvolvida pela Google, para garantir a confiabilidade de sistemas em escala. “A Google designou um time dedicado de engenheiros de software para olhar o ambiente de produção com o objetivo de fazer com que seus sistemas funcionassem de maneira suave, eficiente, e confiável, por meio da adoção da cultura e de práticas da engenharia no mundo da infraestrutura e operações” (MUNIZ, OLIVEIRA e MULLER, 2023, p.2). Diante disso, podemos dizer que, com um foco na engenharia, automação e colaboração, os SREs trabalham para garantir serviços altamente disponíveis e resilientes.

Segundo Buffa (2021), o SRE representa, um conjunto de princípios, um conjunto de práticas, um conjunto de incentivos e um campo de atuação na disciplina maior de engenharia de software. Ou seja, combinando princípios de engenharia de software com operações de infraestrutura para criar sistemas altamente confiáveis, resilientes e escaláveis, mantendo um equilíbrio entre a inovação rápida e a estabilidade dos serviços, torna-se o principal objetivo do SRE.

Para atingir esse objetivo, o SRE aplica conhecimentos de programação, automação e design de sistemas, resolvendo desafios de disponibilidade e desempenho. Essa busca pela excelência operacional é um esforço contínuo e colaborativo, e a adoção das práticas do SRE pode contribuir significativamente para o sucesso dos serviços e infraestrutura. Já que, “o grande objetivo da adoção de SRE é fazer com que usuários finais se sintam satisfeitos ao usar produtos de software” (MUNIZ, OLIVEIRA e MULLER, 2023, p.3).

Contudo, em vez de se concentrar apenas em resolver problemas, os SREs se concentram em construir sistemas e processos para evitar problemas futuros, promovendo práticas de liberação de software que minimizem os efeitos prejudiciais em sistemas e serviços. Além disso, o uso de alertas e monitoramento sofisticados permite a detecção de problemas o mais rápido possível, possibilitando uma resposta rápida e eficiente, evitando

assim, possíveis incidentes. “Essas regras e práticas de trabalho ajudam a manter o foco no trabalho de engenharia, em oposição ao trabalho de operações” (ARAÚJO & NASCIMENTO, 2022, p. 24). Sabendo disso, podemos resumir os princípios de SRE em:

- Confiabilidade: Garantir que todas as aplicações sejam confiáveis e disponíveis para os usuários, com a maior segurança possível, monitoramento e proteção contra falhas.
- Eficiência: É preciso minimizar o tempo de inatividade, prevendo que todos os recursos atendem a diversos tipos de demandas, com um bom desempenho e eficiência.
- Automação: Visando aumentar a eficiência operacional, é preciso utilizar a programação para criar ferramentas e automação, principalmente de tarefas repetitivas.
- Observabilidade: Analisar o comportamento das aplicações, utilizando métricas, logs e rastreamentos para investigar o estado do sistema e identificar problemas ou incidentes é extremamente importante, porque é através disso, que é possível identificar, diagnosticar e resolver problemas de maneira eficaz quando eles surgem.
- Respostas a incidentes: Os SREs trabalham em conjunto com engenheiros de desenvolvimento para identificar e resolver problemas rapidamente.
- Provisionamento e Escalabilidade: É preciso que os sistemas possam ser facilmente escalados para acomodar um aumento na demanda de usuários, ajustando recursos e infraestrutura conforme necessário, evitando também recursos super provisionados.
- Tolerância a falhas: Projetar sistemas com redundância e tolerância a falhas para minimizar o impacto de problemas de hardware ou software, definindo previamente, taxas de sucesso no dia (*Service Level Indicator* - SLI), meta de taxas de sucesso (*Service Level Agreement* - SLA) e parâmetro de sucesso para os clientes (*Service Level Objective* - SLO).

Apesar de todos os pilares serem específicos para o SRE, e de não serem equivalentes, ainda ocorre bastante confusão com o termo DevOps, por haver bastante proximidade em seus conceitos, uma vez que eles são complementares e muitas vezes comparados devido à sua interseção em práticas e objetivos. Porém, Enquanto o SRE é direcionado ao desenvolvimento e entrega de melhorias e novas funcionalidades nos sistemas, assim como a mitigação de erros e bugs, o DevOps é uma cultura que conta com práticas que visam melhorar constantemente a entrega da qualidade do produto (ROCHA, 2021). Enquanto o SRE foca mais na confiabilidade do site, sob a perspectiva do cliente, o:

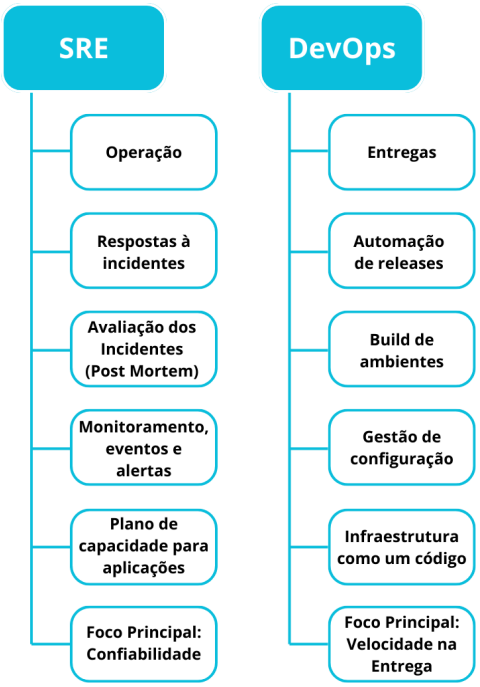
> “DevOps busca aumentar a cadência das entregas e estreitar a colaboração entre áreas envolvidas no desenvolvimento de um produto de software através da utilização

> automatizada de ferramentas, maximizando a qualidade a fim de atender à crescente demanda das áreas de negócio, pela disponibilização de novas funcionalidades para o usuário final, evitando assim o retrabalho" (MUNIZ, OLIVEIRA e MULLER, 2023, p. 3).

Ou seja, "DevOps é uma abordagem de cultura, automação e design de plataforma que tem como objetivo agregar mais valor aos negócios e aumentar sua capacidade de resposta às mudanças por meio de entregas de serviços rápidas e de alta qualidade" (ARAÚJO & NASCIMENTO, 2022, p.21).

O resumo das principais diferenças de SRE e Devops, pode ser visto na figura:

**Figura 01**. Principais diferenças de SRE e Devops.




Fonte: Imagem da Autora.

Em resumo, o SRE busca maneiras de melhorar sistemas, otimizar processos e garantir que a infraestrutura esteja pronta para lidar com volumes significativos de transações durante períodos de pico, lidando sempre com problemas. Enquanto o papel do Devops é criar maneiras de entregar o código de forma ágil e segura, além de criar padrões os quais o SRE deve seguir.

## 2.2 Infraestrutura como código

Infraestrutura com código, *Infrastructure as a Code*, IaaC ou IaC, são todos termos designados para uma forma de provisionar infraestrutura através de código, ao invés de ações manuais. Ou seja, "a ideia da IaaC é tratar os recursos de infraestrutura como se fossem

software em um código de programa, o que permite que a IaaC utilize as práticas de desenvolvimento de software provisionando e implantando infraestrutura de rede rapidamente e consistente" ( LIMA, JUCÁ, LEMOS & SILVA, 2021, p. 565.).

É através do IaC que templates(modelos) de infraestrutura são criados, e disponibilizados para que SREs e Devops possam provisionar recursos de infraestrutura de forma prática e segura. Além disso, controle de versões de alterações, facilidade em gerenciar grandes quantidades de infraestrutura, redução de custos, redução de erros, padronização de criação de recursos e mais rapidez nas entregas, são alguns benefícios resultantes da utilização de IaC.

Também é possível automatizar tarefas repetitivas, já que os IaCs podem ser integrados a métodos de entrega contínua (pipelines). Isso reduz muito o risco de erros e falhas, o que é crucial para as empresas. Uma vez que, a automação é uma meta fundamental em qualquer ambiente de computação. E a infraestrutura como código (IaC) é usada para automação de infraestrutura para criar ambientes (AWS-A, 2023).

Em suma, a infraestrutura como código (IaC) é uma abordagem que transforma a maneira como provisionamos e gerenciamos recursos de infraestrutura, simplificando os processos de infraestrutura, mas também impulsionando a inovação e a agilidade.

## 2.3 Computação na Nuvem

Apesar de atualmente o termo *Cloud Computing*, ou traduzido para Computação na Nuvem, estar em alta, ele não é um termo recente. Com o avanço da inserção da tecnologia no nosso cotidiano, empresas de TI precisaram adaptar suas aplicações, sejam softwares, sites ou aplicativos, para que esses pudessem ser acessados e utilizados de forma mais flexível e eficiente, com o menor custo possível. Para isso, elas passaram a utilizar serviços de computação sob demanda disponibilizados pela internet, que é justamente a definição de computação na nuvem.

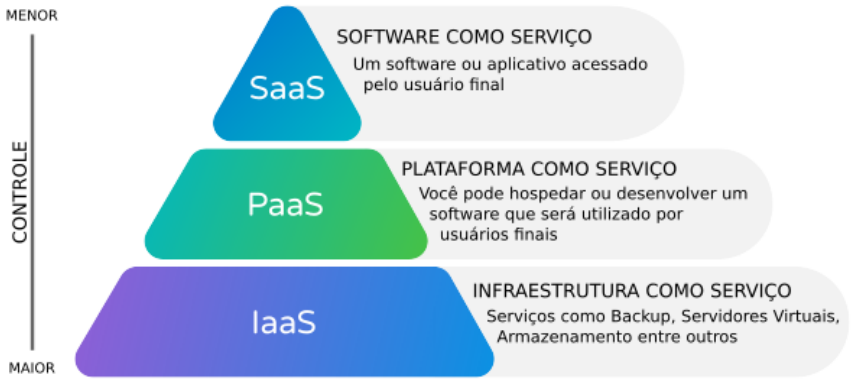
A computação em Nuvem, pode ainda ser dividida em:

- Nuvem Privada: Quando a nuvem é gerenciada e utilizada apenas pela organização que a desenvolveu, não estando disponível para uso externo.
- Nuvem Pública: Quando os recursos são disponibilizados para o público em geral;
- Nuvem Comunitária: Quando a nuvem é compartilhada por organizações que possuem um objetivo comum;

- Nuvem Híbrida: É uma mistura da nuvem privada com a pública.

Diante disso, surgiram alguns conceitos que ajudam a garantir esse objetivo, como por exemplo: Infraestrutura como serviço (IaaS); Plataforma como serviço (PaaS) e software como serviço (SaaS). Esses termos mudaram a maneira como as empresas lidam com suas necessidades de TI e todos eles estão inseridos no universo da computação na nuvem. Uma vez que, “ a computação em nuvem é uma tecnologia que permite que empresas de diversos portes e segmentos utilizem recursos computacionais como armazenamento, processamento e rede de forma mais eficiente e flexível, pagando apenas pelo que utilizarem” (FREITAS E NETO, 2023, p.2).

**Figura 02.** Hierarquia IaaS, PaaS e SaaS.




Fonte: Tecnomega, 2021.

“A nova arquitetura introduzida pela Cloud Computing permite que as organizações escolham o modelo adequado para a arquitetura dos seus aplicativos e onde armazenar seus dados” (VERAS, 2012, p. 30). Tudo isso, sem a necessidade de comprar servidores físicos de hardware e redes. Dessa forma:

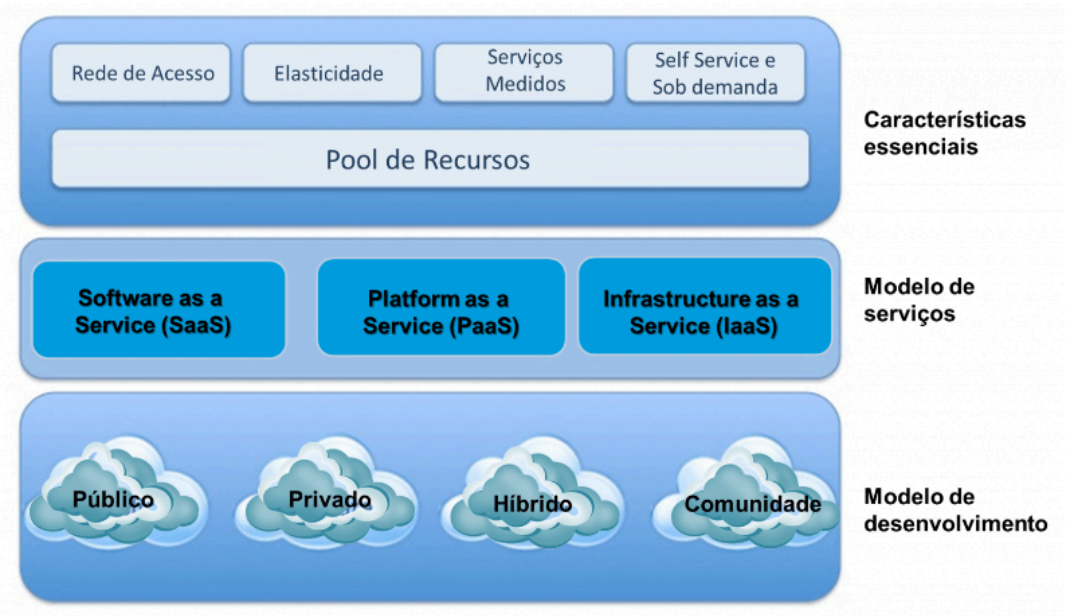
> “A redução de custos com o uso da nuvem é um dos principais pontos com relação a benefícios, pois há eliminação de investimentos em recursos que não são necessários o tempo todo, ou seja, na nuvem o pagamento pode ser conforme o uso e há elasticidade para crescer ou diminuir a quantidade de recursos conforme a necessidade do negócio em vez de fazer um investimento e deixá-lo ocioso para uso em determinados momentos” (MEIRELLES & VIEIRAS, 2015, p.1219).

“O crescimento do foco no negócio da organização contratante é outro benefício, visto que ao se contratar um provedor de serviços em nuvem as questões relacionadas a atualização da infraestrutura de TI serão de responsabilidade da empresa contratada” (DAVID, ALVES, NUNES & OLIVEIRA, 2022. p. 540). Além disso, a computação na nuvem, apresenta como

benefícios a escalabilidade, já que as empresas podem provisionar seus recursos conforme a capacidade que precisam naquele momento. Muitas vezes os recursos estão provisionados em vários locais (pool de recursos) distribuídos ao redor do mundo, por isso, é mais fácil recuperar dados perdidos em desastres, garantindo assim, uma maior segurança da informação.

Em resumo, a computação em nuvem pode ser explicada na figura 03.

**Figura 03**. Modelo NIST Computação em Nuvem.




Fonte: IUNES, 2017.

Embora a computação em nuvem apresenta benefícios, é importante ressaltar os riscos na adoção dessa tecnologia (VERAS, 2012). Esses riscos estão relacionados principalmente: A dependência da internet, já que sem ela, pode haver falhas de rede e conectividade; Estão sujeitos também à violação de segurança, justamente por dados sensíveis estarem expostos na internet. Por esse motivo, é extremamente importante adotar um conjunto de práticas de segurança, para utilizar esse conceito de forma confiável. Uma vez que, um dos maiores problemas que empresas podem enfrentar com os serviços baseados em computação em nuvem é o uso incorreto de ferramentas (TEBALDI, 2018).

## 2.4 Kubernetes

Segundo Silva (2023, p. 16), "a virtualização é uma prática altamente utilizada pelas empresas, e que ganhou ainda mais força com o crescimento da computação em nuvem. Essa

prática envolve outros dois conceitos: Máquina virtual e Container". As máquinas virtuais, permitem que um sistema operacional seja instalado mesmo sem servidor físico. Ou seja, "cada máquina virtual oferece um ambiente similar ao de uma máquina física, possibilitando assim que cada uma tenha seu próprio sistema operacional, aplicativos e serviços de rede" (SILVA, 2017, p. 14). Já o contêiner é uma unidade padrão de software que empacota o código e todas as suas dependências para que o aplicativo seja executado de forma rápida e confiável de um ambiente de computação para outro(DOCKER, 2024).

Como os contêineres possuem maior portabilidade e eficiência em relação à máquinas virtuais, voltados principalmente para a disponibilização de softwares em PaaS, eles passaram a ser mais utilizados nos últimos tempos. Dessa forma, passou a ser preciso criar formas de melhorar o gerenciamento de grandes quantidades de contêineres. Nesse cenário, surgiu o Kubernetes (K8s), que é um produto *Open Source* utilizado para automatizar a implantação, o dimensionamento e o gerenciamento de aplicativos em contêiner (KUBERNETES.IO, 2024). Ou seja, "o Kubernetes consiste num ambiente gerenciador de contêineres, com a finalidade de administrar o ciclo de vida de contêineres em nós num ambiente de cluster" (NETTO, LUNG, OLIVEIRA, LUIZ, RECH, JÚNIOR, 2017, P.447).

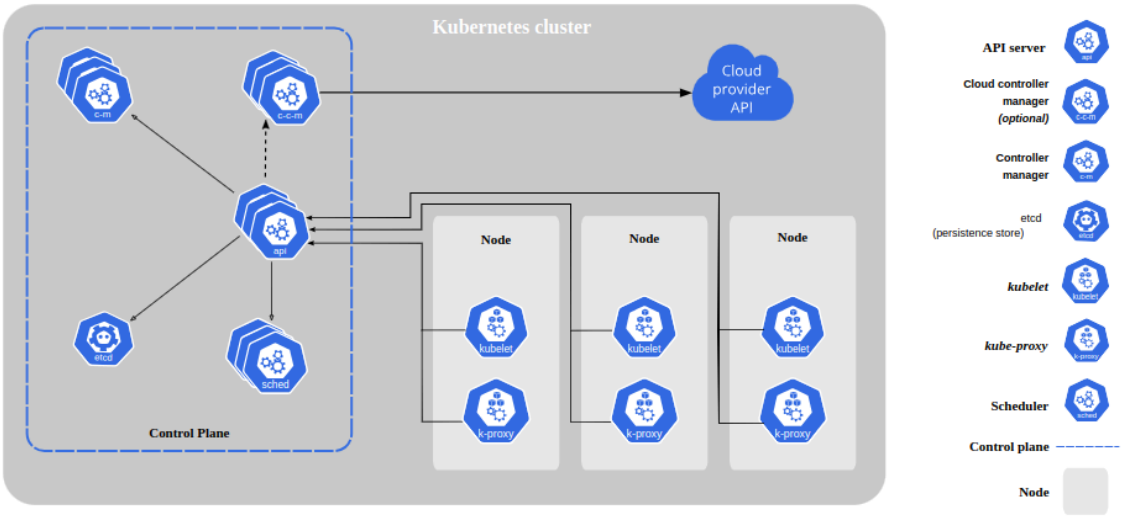
Dentre as funcionalidades providas pelo Kubernetes, pode-se citar como principais: "controle de admissão dos contêineres, balanceamento de recursos, descoberta de serviços entre os contêineres, publicação do serviço para acessos externos ao cluster e o balanceamento de carga entre os contêineres" (NETTO, LUNG, OLIVEIRA, LUIZ, RECH, JÚNIOR, 2017, P.447). Além disso, o Kubernetes também oferece escalabilidade automática, gerenciamento de armazenamento e atualizações de aplicativos sem tempo de inatividade, tornando-o uma ferramenta fortemente utilizada na orquestração de contêineres em ambientes de produção. Quanto ao seu funcionamento, pode-se dizer que:

> "Um cluster Kubernetes é formado de nós, que são as máquinas (físicas ou virtuais) nas quais containers são hospedados. A máquina principal contém componentes gerenciais, e pode ser replicada para tolerar faltas de parada. O POD é uma unidade mínima de gerenciamento no Kubernetes, e pode conter um ou mais containers. PODs recebem endereço de rede e são alocados em nós. Containers que estão no mesmo POD compartilham recursos e são mantidos no mesmo nó. Clientes contactam o cluster por meio de um firewall, que geralmente distribui pedidos aos nós segundo uma estratégia de balanceamento de carga. O proxy recebe pedidos de clientes e encaminha os pedidos ao POD. Se o POD estiver replicado, um balanceamento de carga interno ao nó distribui aleatoriamente o pedido a uma das réplicas. O kubelet gerencia PODs, imagens, containers e outros elementos do nó. O cAdvisor monitora recursos utilizados pelo container. O kubelet repassa as informações de

> monitoramento à máquina principal, que deve atuar quando necessário". (NETTO, LUNG, CORREIA, LUIZ,2016, P.4).

Podemos ver isso resumidamente, na figura 04:

**Figura 04**. Funcionamento de um cluster kubernetes.




Fonte: Kubernetes IO, 2024.

Segundo Ferrari (2021), o Kubernetes conduz um conjunto de processos independentes e combináveis, de modo a alcançar o estado desejado de execução, resultando em um sistema eficaz e robusto, resiliente e extensível, e seguro. No entanto, apesar de possuir diversos benefícios, sua utilização pode ser um pouco difícil para quem tem pouco conhecimento sobre computação em nuvem, por causa disso, surgiram plataformas para gerenciamento de kubernetes, que facilitam a implementação, gerenciamento e monitoramento de clusters kubernetes, como é o caso do Rancher. Que é uma plataforma unificada nativa da nuvem que ajuda as equipes a gerenciar seus Kubernetes, desde a infraestrutura até os aplicativos (Rancher, 2024).

## 2.5 AWS

Segundo Veras (2013), a *Amazon Web Services* (AWS), é a principal oferta de arquitetura do tipo cloud computing da atualidade, permitindo às empresas o acesso a serviços de infraestrutura sob demanda, reduzindo custos e riscos. Dessa forma, as empresas podem escalar recursos rapidamente conforme necessário, promovendo a agilidade e a eficiência operacional.

Segundo a AWS-B(2024), são mais de 200 serviços completos de datacenters em todo o mundo. Serviços esses que em sua maioria, são focados em computação, armazenamento de

dados, Machine Learning, Inteligência Artificial, Data Lakes, Análises, Infraestrutura de Redes, virtualização, entre outros. Alguns desses serviços são: Api Gateway, Cloudformation, DynamoDB, EKS, Lambda, Redis, IAM, S3, SQS, Cloudwatch.

### 2.5.1 Api Gateway

"Uma API é um conjunto de rotinas e padrões estabelecidos por um software para utilização das suas funcionalidades por aplicativos que não podem se envolver em detalhes da sua implementação, mas apenas usar seus serviços" (VERAS, 2013, p.13). Sabendo disso, o Amazon API Gateway é um serviço da AWS para criação, publicação, manutenção, monitoramento e proteção de APIs REST e WebSocket em qualquer escala (AWS-C, 2024). Ou seja, simplifica o processo de criação e manutenção de APIs, auxiliando os desenvolvedores a terem menos preocupações relacionadas à infraestrutura de comunicação.

### 2.5.2 Cloudformation

O AWS CloudFormation possibilita a criação de recursos na nuvem com base em *templates*, diminuindo o tempo de provisionamento com segurança e garantindo gerenciamento e escalabilidade(ANDRADE, 2022). Em outras palavras, através do cloudformation podemos pressionar diversos recursos de uma única vez, por meio de infraestrutura como código, e salvar esses recursos em stacks(pilhas). Possibilitando assim, a implementação de entrega contínua.

### 2.5.3 CloudWatch

Segundo a AWS-D(2024), o Amazon CloudWatch monitora os recursos da AWS e as aplicações executadas na AWS em tempo real. Ou seja, é basicamente um repositório de métricas, que coleta e fornece automaticamente dados das aplicações e recursos, para que esses possam ter suas performances analisadas e monitoradas.

### 2.5.4 DynamoDB

O DynamoDB é um banco de dados não relacional (NoSQL) da AWS totalmente na cloud. Uma solução escalável, altamente disponível, facilmente gerenciável, serverless (sem servidor) e segura (CHELES, 2022). Algumas das principais características desse recurso, é que ele é orientado a Documento ou Chave-valor. Por ser sem servidor, ele é mais rápido do que os bancos de dados

habituais. Além disso, possui escalabilidade automática e integração com outros serviços AWS.

### 2.5.5 Amazon EKS

O *Amazon Elastic Kubernetes Service* (Amazon EKS) é um serviço gerenciado que elimina a necessidade de instalar, operar e manter o seu próprio ambiente de gerenciamento do Kubernetes na AWS (AWS-E, 2024). Algumas das principais características desse recurso, é que ele possui uma boa facilidade de implantação e escalabilidade, possui integração com outros serviços AWS. Além disso, tem um gerenciamento simplificado, com alta disponibilidade e resiliência.

### 2.5.6 Lambda

O AWS Lambda é um serviço de computação sem servidor e orientado a eventos que permite executar código para praticamente qualquer tipo de aplicação ou serviço de backend sem provisionar ou gerenciar servidores(AWS-F, 2024). Algumas das principais características desse recurso, é que ele possui suporte a várias linguagens de programação, integração com eventos e serviços da AWS, possui escalabilidade automática, e não possui servidor.

### 2.5.7 Elasticache para Redis

Segundo a AWS-G (2023), o *Amazon ElastiCache* para Redis é um armazenamento de dados na memória extremamente rápido para potencializar aplicativos em tempo real em escala de Internet criado com base no Redis de código aberto e compatível com suas APIs. Algumas das principais características desse recurso, é que ele possui armazenamento em memória no formato chave-valor, operações atômicas, geoespacial e busca por índice, e cache distribuído.

### 2.5.8 IAM Roles

Uma IAM Role é uma identidade que pode ser criada no AWS *Identity and Access Management* (AWS IAM) e ter permissões atribuídas a ela diretamente ou via políticas do IAM (BORTOLETTO & MORAES, 2022). É através de uma roles, que podemos definir quais privilégios um recurso ou usuário vai ter na AWS. Esse recurso é fundamental para garantir um maior controle de acessos, e assim uma maior segurança para as aplicações hospedadas na AWS.

### 2.5.9 S3

O S3 (Simple Storage Service), é um serviço de armazenamento de dados na nuvem da AWS, altamente escalável e confiável (LECHETA, 2014, p.40). O S3 possui foco em escalabilidade, disponibilidade, segurança e performance, tudo isso com um custo extremamente acessível(MACHADO, 2020). Além disso, com o S3 você consegue gerenciar melhor seus dados, criando algumas políticas de segurança e de armazenamento, e também consegue utilizar outros recursos da AWS para fazer o processamento e monitoramento desses dados.

### 2.5.10 SQS

Segundo a AWS-H(2024), o Amazon Simple Queue Service (SQS) permite que você envie, armazene e receba mensagens entre componentes de software em qualquer volume, sem perder mensagens ou precisar que outros serviços estejam disponíveis. Além de possuir integração nativa com outros serviços da AWS, o SQS, possui uma escalabilidade automática, um bom gerenciamento de filas, entregas confiáveis, e modelos diferentes de entregas de mensagens.

## 2.6 Automação e Chatbots

“O dia a dia de um profissional de SRE deveria ser dividido em duas partes, sendo a primeira identificando e resolvendo problemas, e a segunda criando automações para solução de problemas comuns” (MUNIZ, OLIVEIRA e MULLER, 2023, p. 7). Em outras palavras, umas das principais habilidades do SRE é identificar como facilitar e otimizar processos, promovendo a máxima autossuficiência e desempenho possível de um software, removendo trabalhos manuais e repetitivos do sistema.

É aí que entra o conceito de toil, que ainda segundo esses autores, é um trabalho manual ineficiente que pode ser reduzido mediante um serviço automatizado e confiável. Qualquer trabalho manual, repetitivo e baseado em critérios de decisão de objetivos pode ser considerado toil e consequentemente passível de automação. Sendo assim, a automação consiste no uso de equipamentos e softwares para simplificar e agilizar os processos internos da organização, reduzindo a quantidade de toils, sendo extremamente necessário para um ambiente corporativo ágil e confiável, reforçando a colaboração entre os times de SRE e Engenharia.

É diante desse cenário que surgem os ChatBots, que são programas de computador que muitas vezes, utilizam inteligência artificial (IA) para simular conversas humanas por meio de mensagens de texto ou fala e que podem ser encontrados em uma variedade de plataformas, como sites, aplicativos de mensagens instantâneas, redes sociais e sistemas de atendimento ao cliente. Eles auxiliam na criação de automação de tarefas repetitivas, sendo muitas vezes integrados a sistemas de monitoramento, gerenciamento e comunicação devido à sua capacidade de interagir de forma natural com os usuários, podendo ser integradas a ferramentas como: Slack, Microsoft Teams, Opsgenie, entre outras. “Essas integrações se tornam valiosas no processo de comunicação, tendo o chatbot como agente integrador de informações fornecidas por diversas ferramentas presentes no ecossistema da empresa” (MUNIZ, OLIVEIRA e MULLER, 2023, p. 61). Dessa forma, os chatbots são capazes de fornecer respostas rápidas a perguntas frequentes, realizando diagnósticos preliminares de incidentes e mesmo executando ações corretivas em tempo real. Reduzindo a carga de trabalho repetitiva e manual dos profissionais de SRE.

## 2.7 Trabalhos Relacionados

A Engenharia de Confiabilidade de Sites tem demonstrado ser útil em várias situações em uma empresa de tecnologia. Porém, é importante ter em mente que nem sempre é possível obter benefícios com esse método. Portanto, é necessário fazer uma análise aprofundada. Além disso, é notável que pesquisas anteriores se concentraram em tipos muito específicos de automação, o que deixou uma lacuna significativa em relação à utilização de bots. Esta diferença mostra o papel crucial deste trabalho, pois se concentrou exclusivamente no papel de SRE em compreender as necessidades de uma empresa e de aplicar técnicas de automação utilizando conceitos da área. Dessa forma, nesta seção, fornecemos uma revisão dos estudos mais importantes sobre o uso da automação em uma variedade de campos, com ênfase nos trabalhos Engenharia de confiabilidade de site para aplicativos móveis IOS em pequenas e médias empresas indústrias de escala (KAVYASHREE; SUPRIYA; LOKESH, 2019), Utilização de chat bots em LLMS para automação de testes de Software (OLIVEIRA, 2024), Chatbot: uma visão geral sobre aplicações inteligentes (JÚNIOR; CARVALHO, 2018).

Em (OLIVEIRA, 2024), foi realizada uma pesquisa bibliográfica e de estudo de casos, com o objetivo realizar uma busca por soluções eficientes e eficazes para o processo de Testes Automatizados, principalmente para sistemas de grande escala, a partir da utilização de ChatBots. O Autor analisou as ferramentas ChatGPT 3.5, Google Bard, Aria Opera e

Microsoft Bing, e Perplexity AI sob a plataforma de testes de Login/Cadastro sa SMART NX, em diversos cenários. Como resultado, o autor informa que ChatGPT teve melhor desempenho como ferramenta para auxiliar iniciantes em testes automatizados, ressaltando a importância de como uma ferramenta de automatização pode ser importante para garantir uma maior confiança nos processos de testes.

(JÚNIOR; CARVALHO, 2018), realizou uma pesquisa também bibliográfica, e um estudo de caso, visando identificar quais são as principais características que definem bots, e simular uma interação desses aplicativos com um usuário humano. Buscando apresentar as aplicações nas quais os bots mais se destacam. Sendo assim, no estudo de caso, os autores consideraram que seriam apresentadas algumas pessoas relacionadas entre si, utilizando as ferramentas SWI-Prolog, API Dialogflow, para criar uma assistente virtual, utilizando as tecnologia de Processamento de Linguagem Natural e Aprendizagem de Máquina, com o intuito de demonstrar na prática, o funcionamento dos chatbots. Diante disso, o autor conclui que os bots estão cada vez mais se tornando presentes em tarefas do cotidiano. Atuando de forma ubíqua na vida das pessoas, auxiliando-as em tarefas desde as mais simples até as mais complexas.

A confiabilidade do software é essencial para garantir a excelência do software, de acordo com o estudo de (KAVYASHREE; SUPRIYA; LOKESH, 2019). Assim, usando uma pesquisa bibliográfica e estudos de caso, os autores definiram métricas para estabelecer testes de confiabilidade na indústria de pequena e média escala. Logo, para garantir a confiabilidade contínua durante o desenvolvimento, eles desenvolveram o Halodoc, um aplicativo que incorporava os conceitos de SRE em aplicativos móveis. Além disso, vários elementos são abordados: avaliação de riscos da aplicação, conveniência da aplicação, desempenho e eficiência, corrigibilidade, gerenciamento do design da aplicação, monitoramento e notificação de problemas e análise pós-mortem. Os autores concluem que a adoção da técnica SRE melhorou a confiabilidade da aplicação móvel, enfatizando a importância de ferramentas e processos adequados, além da colaboração entre equipes, para alcançar esse objetivo.

Todos os trabalhos ressaltam o potencial da utilização de automações e de práticas de SRE para garantir confiabilidade e agilidade de processos. No entanto, é importante notar que nenhum dos estudos teve como foco criar automações implementando práticas de SRE, com o intuito de facilitar ainda mais as operações na nuvem. Porém, todos os trabalhos trazem valiosas contribuições para a área de SRE, impulsionando o uso criativo e adaptativo de automações para otimizar os processos de desenvolvimento de tecnologia.

# 3. METODOLOGIA

Aqui serão abordados todos os aspectos metodológicos da pesquisa realizada, descrevendo-se os procedimentos necessários e úteis para avaliar os desafios enfrentados e os benefícios obtidos na implementação de estratégias de automatização de operações na nuvem, mediante um estudo de caso com um Slackbot em uma empresa de tecnologia.

Esse estudo tem por finalidade realizar uma pesquisa de natureza aplicada, uma vez que utiliza conhecimento da pesquisa básica para resolver problemas. Para alcançar os objetivos propostos e melhor apreciação deste trabalho, foram utilizadas as abordagens quantitativa. Segundo Manzato e Santos (2012), os métodos de pesquisa quantitativa, são utilizados quando se quer medir opiniões, reações, sensações, hábitos e atitudes etc. de um público-alvo mediante uma amostra que o represente de forma estatisticamente comprovada. Isto não quer dizer que ela não possa ter indicadores qualitativos. Uma vez que, " uma pesquisa de natureza busca dar respostas a questões muito particulares, específicas, que precisam de elucidações mais analíticas e descritivas" (OLIVEIRA, 2020, p. 02 apud RODRIGUES, OLIVEIRA e SANTOS, 2021, p. 157).

Com intuito de conhecer a problemática sobre a área de estudo foi realizada uma pesquisa descritiva, já que, é a partir da coleta de dados de eventos anteriores, que o pesquisador irá conseguir descrever o que aconteceu, gerando um novo conhecimento. A pesquisa foi realizada entre os meses de setembro de 2023 e abril de 2024. E para obtenção dos dados necessários, foi utilizado um estudo de caso. "A estratégia metodológica do estudo de caso tem como base questões de pesquisa tanto do tipo "qual" ou "como", que podem gerar análises descritivas inferenciais, quanto do tipo "por que", de natureza explicativa" (SÁTYRO e ALBUQUERQUE, 2020, p.6).

Assim, por meio de um estudo de caso e análise de resultados, foram identificados benefícios operacionais potenciais por meio da pesquisa exploratória e descritiva.

## 3.1 Contextualização da Empresa

A empresa *Banking as a Service*(Banco como serviço), o qual o Harry Botter foi desenvolvido e disponibilizado, possui aproximadamente 1.500 funcionários, sendo pelo menos, metade da área de tecnologia. Todos os produtos desenvolvidos nela, encontram-se hospedados em servidores na cloud, principalmente na AWS. Para que os times de

desenvolvimento tenham suporte de infraestrutura existem times de SRE, Devops, Cloud, segregados por unidades de negócio.

Apesar de alguns times utilizarem tecnologias diferentes, existem algumas comuns neles todos, como, por exemplo, a utilização de kubernetes, e para visualização dos cluster, utiliza-se, principalmente, o Rancher. Além disso, a maioria dos times utiliza tecnologias como armazenamento em cache com o Redis; Salvamento de logs no cloudwatch; Versionamento de código no github; Lambda para armazenar código sem servidor; Criação de infraestrutura utilizando IaaC; Todo os times utilizam o slack como principal meio de comunicação; E para gerenciamento de atividades é utilizado o jira, que é uma ferramenta que permite o monitoramento de tarefas e acompanhamento de projetos, garantindo o gerenciamento de todas as suas atividades em único lugar (JIRA, 2024).

Por causa desse compartilhamento de tecnologias, os times de desenvolvimento de software, solicitavam aos times de SRE, muitas ações para criar, alterar, excluir e até mesmo gerenciar a maioria desses recursos de cloud. Resultante disso, o número de chamados ( tickets de atimento) cresciam cada vez mais, a medida em que a empresa crescia. Sendo assim, os times de SRE tinham muitas vezes que parar de focar em projetos estruturantes para atender essas demandas repetitivamente. Além disso, muitas vezes ao atender essas demandas, devido à intensidade de trabalho, alguns erros eram cometidos pelos profissionais de SRE, causando retrabalho, falhas de processos, e até mesmo falha de segurança, principalmente em ambientes produtivos.

Outro erro bastante comum, era o da falta de verificação, por parte dos SREs, dos formulários de segurança. Uma vez que, para realizar qualquer ação em ambientes de produção, era preciso que o profissional abrisse um formulário no jira, informando todos os procedimentos que seriam feitos, e se eles iriam causar indisponibilidade ou intermitência na aplicação. Esse formulário, deveria ser aprovado por um profissional de controle de qualidade, analista de riscos, e pelo líder direto do profissional que iria executar a ação. Dessa forma, era garantido que mesmo que ocorresse algum erro na aplicação, todos os clientes da empresa já estariam cientes que algum imprevisto iria acontecer, e um plano de ação para a correção daquele teria que ser seguido.

Diante de tais problemas, foi verificado quais eram os que aconteciam com forma mais frequente, e que poderiam ser resolvidos mediante automações. Nessa análise, constatou que um dos principais pedidos dos times de desenvolvimento para os times de SRE era de acesso aos clusters no Rancher, Gerenciamento de throttlings nas APIs das aplicações, Inclusão de Clusters no Rancher e exclusão de chaves em clusters Redis. Dessa forma, o

Harry Botter foi pensando em resolver tais problemas, reduzindo assim os acionamentos para os times de SRE, e garantindo maior segurança e otimização de processos.

### 3.2 Revisão bibliográfica

Essa etapa para o desenvolvimento do trabalho deu-se com a pesquisa e discussão teórica, uma vez que proporciona o atributo essencial para a compreensão do tema estudado. Recorrendo a autores que fundamentam os conceitos abordados como: SRE, Computação em Nuvem, Automação e Otimização operacional.

Objetivando-se em colher dados de outros autores, a fim de transformá-los em informações para a elaboração do desenvolvimento do trabalho. Dentre as ações nesta etapa, estão a leitura da bibliografia e a fundamentação teórica.

### 3.3 Escolha das Ferramentas

Garantir escalabilidade, redução de custos com infraestrutura, maior segurança, dentre outros benefícios, é uma das principais características da computação em nuvem ou *cloud computing*. “A computação em nuvem é uma tecnologia que permite que empresas de diversos portes e segmentos utilizem recursos computacionais como armazenamento, processamento e rede de forma mais eficiente e flexível, pagando apenas pelo que utilizarem” (FREITAS e NETO, 2023, p.1).

Por esse motivo, empresas começaram a aderir esse formato em alta escala, ocasionando em um grande aumento de plataformas que buscam auxiliar a construção de infraestrutura, tais como: Google Cloud Platform, Microsoft Azure, a Amazon Web Services (AWS) e IBM Cloud. Segundo Silva (2023, p. 19), “a AWS é uma das principais provedoras de serviços em nuvem, oferecendo uma ampla gama de recursos e soluções para empresas e desenvolvedores”. Oferecendo serviços de computação em nuvem confiáveis, escaláveis e acessíveis (AWS-I, 2023).

Além disso, a empresa onde o bot foi testado, utiliza como principal ferramenta de comunicação, o Slack. O Slack além de possuir uma grande aderência na empresa, possui diversas possibilidades para o desenvolvimento de bots e outros tipos de automação (Slack, 2024). Segundo Olawanle (2023), ao integrar um Slackbot ao seu fluxo de trabalho, você pode economizar tempo e aumentar a produtividade. Uma vez que é uma plataforma de mensagens instantâneas que permite a inclusão de plugins e de outros aplicativos

personalizados que podem ser utilizados de forma intuitiva. Logo, para facilitar e garantir agilidade, escolheu-se essa plataforma para que os times de engenharia e de produtos pudessem utilizar a automação.

Como linguagem de programação, foi utilizado o python 3.12. Que é uma linguagem de programação de alto nível, ou seja, com sintaxe mais simplificada e próxima da linguagem humana, utilizada nas mais diversas aplicações, como desktop, web, servidores e ciência de dados (MELO, 2021).

Por fim, o armazenamento do código da automação, bem como da infraestrutura, foi versionado no github da organização da empresa. Visto que, o GitHub é um serviço baseado em nuvem que hospeda um sistema de controle de versão (VCS) chamado Git. Ele permite que os desenvolvedores colaborem e façam mudanças em projetos compartilhados enquanto mantêm um registro detalhado do seu progresso (ANDREI, 2023). Por ser um repositório privado, e particular da empresa, o mesmo não pode ser disponibilizado.

### 3.4 Implementação de técnicas

Considerando, aplicar conceitos de SRE, as técnicas e pilares utilizados no desenvolvimento do projeto foram:

- Versionamento de código, visto que o código da automação do Harry Botter foi versionado no github;
- Elaboração de infraestrutura como código, toda a infraestrutura do Harry Botter foi criada através de IaaC;
- Como um dos principais objetivos do bot era fazer manipulações na nuvem, e como toda sua arquitetura encontra-se na AWS, foram utilizados conceitos de cloud na arquitetura;
- Foram desenvolvidas algumas automações específicas para a necessidade da empresa.
- Integração com APIs de nuvem, uma vez que para realizar alguns serviços, foram utilizados APIs de outras plataformas, como é o caso do Rancher.

### 3.5 Análise das métricas

A quarta etapa consistiu na análise dos dados coletados, durante o período de 1 ano após a publicação do bot, e a conclusão acerca do que foi coletado para concluir o objetivo do trabalho. Nesta etapa, foi desenvolvida a relação entre o que foi constatado na fase de

desenvolvimento do Slackbot, a fim de traduzir as informações obtidas para a proposta do projeto.

Para medir o impacto da utilização do bot, foram obtidas algumas métricas, com o intuito de analisar quais foram os benefícios da inserção de automações no cotidiano dos times de desenvolvimento de sites, e quais são as possíveis melhorias a serem feitas a fim de otimizar ainda mais a automação. As métricas aqui analisadas serão:

- Quantidades de requisição para cada serviço do bot;
- Quantidades de requisições com sucesso e com erro;
- Economia de tempo;
- Custo operacional;

# 4. O HARRY BOTTER

Nesse capítulo, será abordado as principais características e funcionalidades, do Slackbot, o Harry Botter, o qual é o principal objeto de estudo desse trabalho.

## 4.1 Funcionalidades Principais

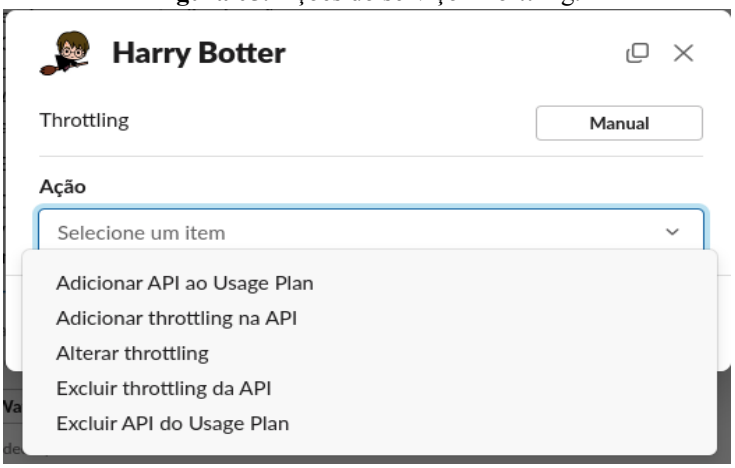
O Harry Botter possui ao todo 4 serviços: Aplicação de throttling em API Gateway, Exclusão de chave em cluster Redis, Adicionar permissão em cluster Kubernetes no Rancher, Incluir um cluster kubernetes no rancher.

### 4.1.1 Throttling em API Gateway

O Throttling é um limiar de solicitações de uma API para ocorrer uma taxa de transferência mais fluida e eficiente. Com o throttling é possível configurar o controle de utilização e cotas para as suas APIs para ajudar a protegê-las da sobrecarga de numerosas solicitações.

Para utilizar o serviço de throttling do Harry Botter, basta o usuário pesquisar na barra de pesquisa por throttling, ou em algum chat digitar "/throttling". Assim que ele utilizar um desses dois comandos, um formulário do tipo modal. Nesse formulário, o usuário deverá preencher quais são as ações que irá querer realizar, que são Adicionar API ao Usage Plan, Adicionar Throttling na API, Alterar throttling, Excluir throttling da API e Excluir API do Usage Plan.

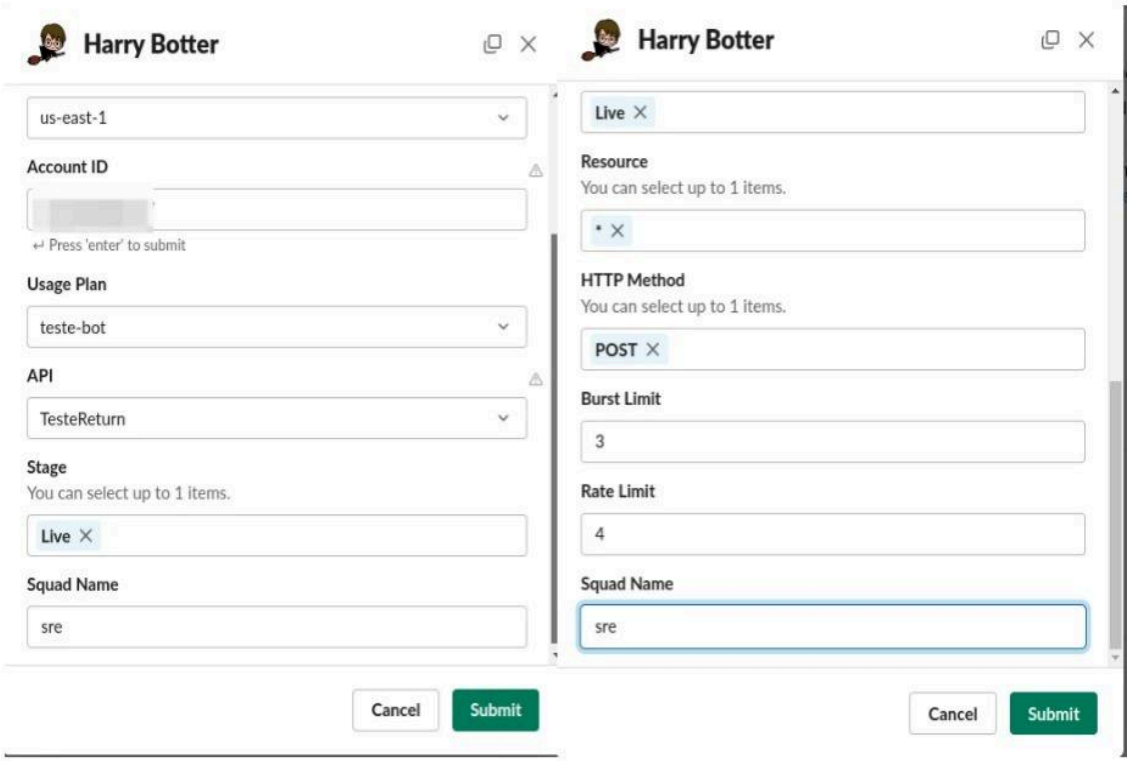
**Figura 05.** Ações do serviço Thorttling.




Fonte: Imagem da Autora.

Além disso, ele deverá colocar dados específicos da API gateway que ele quer manipular, tais como: ID da conta AWS, Nome da API, Método, e Resource Path. Por fim, deverá informar o time (squad) a qual ele pertence.

**Figura 06.** Modal do Throttling.




Fonte: Imagem da Autora.

Após submetido, o Slack irá enviar uma requisição com os dados para a api, que por sua vez irá executar o código com a automação do Harry Botter, que irá fazer o que o usuário solicitou.

### 4.1.2 Importação de Clusters no Rancher

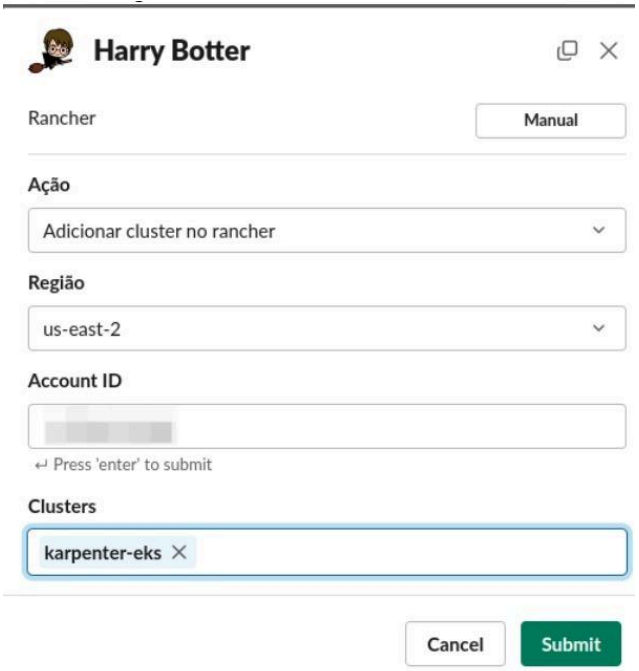
Como mencionado no referencial teórico, o Rancher é uma plataforma open source para gerenciar infraestrutura de Docker e Kubernetes em produção, assim como efetuar deploy de aplicativos usando Docker. O deploy pode ser local ou em servidores remotos (Azure, AWS). E por esse motivo os clusters de outros servidores, podem ser importados no Rancher. Buscando dar autonomia para os times de engenharia e produtos, e até mesmo para times de SRE de outras BUs, além de

garantir uma padronização e agilidade de processos, essa funcionalidade foi implementada.

Para que o usuário possa utilizar esse serviço, basta ele digitar Rancher na barra de pesquisa, ou utilizar o comando "/Rancher" em algum chat, que o modal irá ser carregado para preenchimento.

No modal em questão, o usuário deverá preencher qual a opção, o ID da conta AWS e qual o cluster que deverá ser incluído no Rancher.

**Figura 07.** Modal do adicionar cluster no rancher.




Fonte: Imagem da Autora.

Após a submissão do formulário, se tudo ocorrer corretamente, o cluster poderá ser importado no Rancher, e em poucos minutos ele poderá ser utilizado na plataforma. Com esse processo o usuário não precisa se autenticar localmente no cluster através de SSO, e nem gerar token no Rancher. Isso garante além de agilidade de processos, maior segurança, pois podemos centralizar acesso de administrador para SREs com maior grau de senioridade.

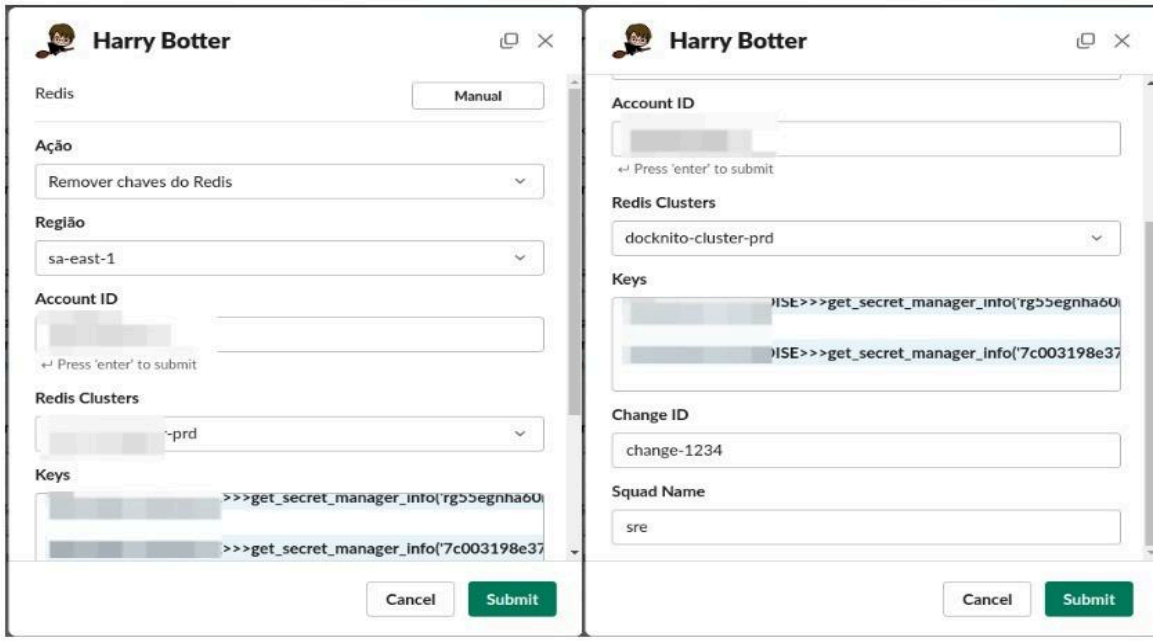
### 4.1.3 Exclusão de Chave Redis

Assim como mencionado anteriormente, o Redis, que significa Remote Dictionary Server, é um datastore de chave-valor rápido e de código aberto na memória (sistemas de softwares em que o código fonte é disponibilizado

publicamente). Ele oferece tempos de resposta inferiores a um milissegundo, permitindo milhões de solicitações por segundo para aplicações em tempo real em setores como jogos, tecnologia de anúncios, serviços financeiros, saúde e IoT.

Muitas vezes para garantir o bom funcionamento de uma aplicação, é preciso excluir alguma chave, portanto, para utilizar esse serviço, basta o usuário digitar Redis na barra de pesquisa ou utilizar o comando "/Redis". Logo em seguida o formulário será carregado, e o usuário deverá preencher dados específicos para a sua necessidade, tais como: Qual é o ID da conta AWS onde está o cluster, qual é o cluster onde está hospedado a chave, qual ou quais são as chaves a serem excluídas, e por fim, o time (squad), a qual pertence.

**Figura 08.** Modal do Redis.




Fonte: Imagem da Autora.

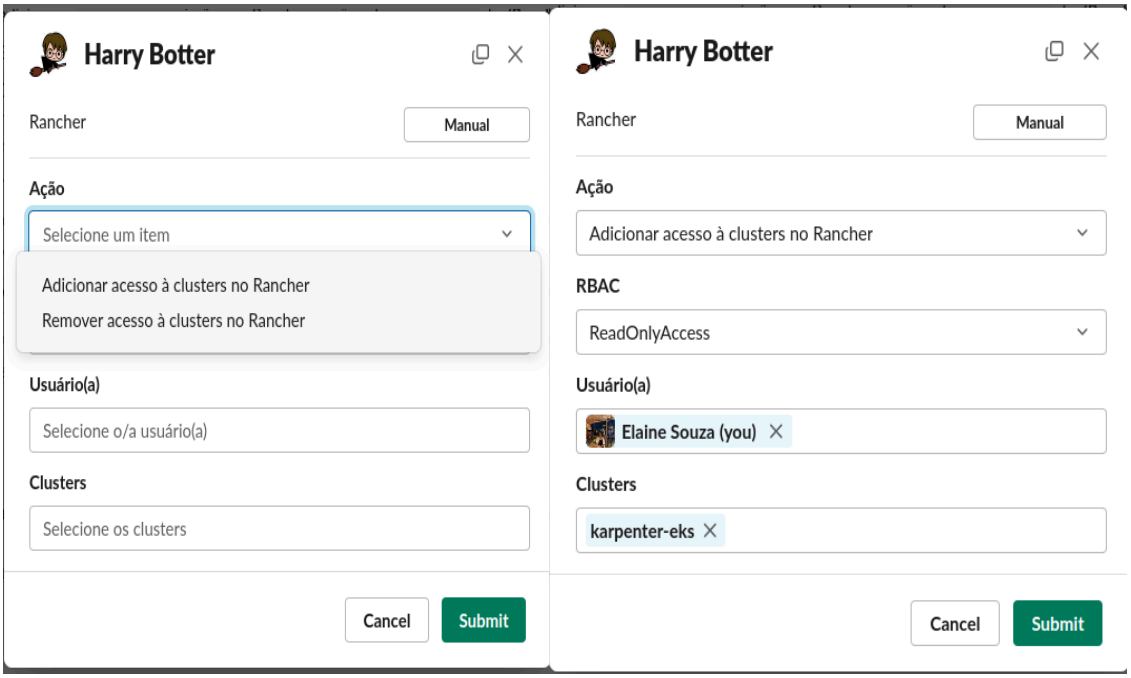
Após submetido, em poucos segundos a chave será excluída. Então, essa funcionalidade evita que chaves sejam excluídas erroneamente, já que o usuário precisa saber previamente qual é a chave precisa ser excluída, e confirmar através da seleção. Além de garantir uma padronização do processo, por ser rápido, em cenários de incidentes, problemas poderão ser corrigidos de forma ágil e confiável, sem a necessidade de acionamento de um analista de SRE. Visto que o próprio time responsável pela aplicação pode fazer a exclusão. Outra vantagem de utilizar esse serviço, é que para exclusão de chaves em ambientes produtivos, ele faz a validação se formulário de segurança está devidamente aprovado pelo líder do time responsável

pela aplicação, por um analista de Controle de Qualidade, e por um Analista de Risco, coisa que muitas vezes, os profissionais de SRE deixavam passar despercebidos.

### 4.1.4 Adicionar/Remover Permissões de Clusters no Rancher

Após um cluster ser importado no Rancher, é preciso que seja concedido permissões para os usuários terem acesso a ele. Essa funcionalidade visa então gerenciar o controle desses acessos, bem como fazer a adição/remoção deles. Para utilizar esse serviço, basta digitar na caixa de pesquisa Permissions, ou utilizar o comando "/Permissions". Após solicitado o modal será carregado e as informações sobre qual ação (Adicionar ou remover), qual o perfil de acesso que vai desde apenas leitura até de edição completa de componentes do cluster, qual o usuário que terá sua permissão manipulada. E por fim, qual o cluster onde será concedida ou removida a permissão.

**Figura 09.** Modal do Permissions.




Fonte: Imagem da Autora.

É válido lembrar que existe um controle no acesso das permissões, apenas os líderes técnicos, e SREs, podem dar um nível de acesso superior ao de leitura.

Então, após o envio da requisição, o acesso é concedido/removido, em poucos segundos, e garante que todos possuam de alguma forma acesso aos clusters de forma rápida e mais uma vez sem necessidade de acionamento de um analista de SRE.

## 4.2 Visão Geral da Arquitetura

Através do *workspace*(área de trabalho) do Slack, o usuário aciona o bot. Ele pode encontrar todos os serviços através da barra de pesquisa, ou por meio do comando "/nome-do-serviço". Após a escolha do serviço, o Slack encaminhará uma requisição para uma *API Gateway* pública (HarryBotterApi), onde a autenticação é realizada mediante validação do token, uma espécie de senha, do aplicativo do Harry Botter no Slack. Sendo assim, requisições de clientes não autorizados ou em formato de payload (dados transmitidos na requisição), diferente do Slack, serão negadas com um código de resposta *Forbidden* (403 - Proibido). E todas as requisições são registradas em grupos de logs do *CloudWatch* para análises posteriores.

Uma vez autenticada e aprovada, a API invoca uma função lambda **(harry-botter-trigger)**, responsável por executar, de forma *serverless* (sem servidor), o código do *frontend*. Ou seja, nesta função estão programados todos os modais, tipo de caixa de diálogo (pop-up), que serão exibidos ao usuário. Dependendo do serviço selecionado, consultas a outros recursos são realizadas para exibir informações relevantes na tela. Por exemplo, o serviço Permissions exibe em tempo real todos os clusters no Rancher para o usuário. Quando o modal é aberto, uma requisição é enviada para o Rancher para buscar os clusters existentes, que serão listados em tempo real no modal. Os demais serviços funcionam de forma similar, buscando informações em recursos terceiros ou em outras contas e exibindo-as em tempo real. Por causa disso, o lambda de frontend precisa de algumas permissões específicas, e isso é liberado na role **(RoleHarryBotter),** que é uma identidade de acesso com permissões políticas de acesso.

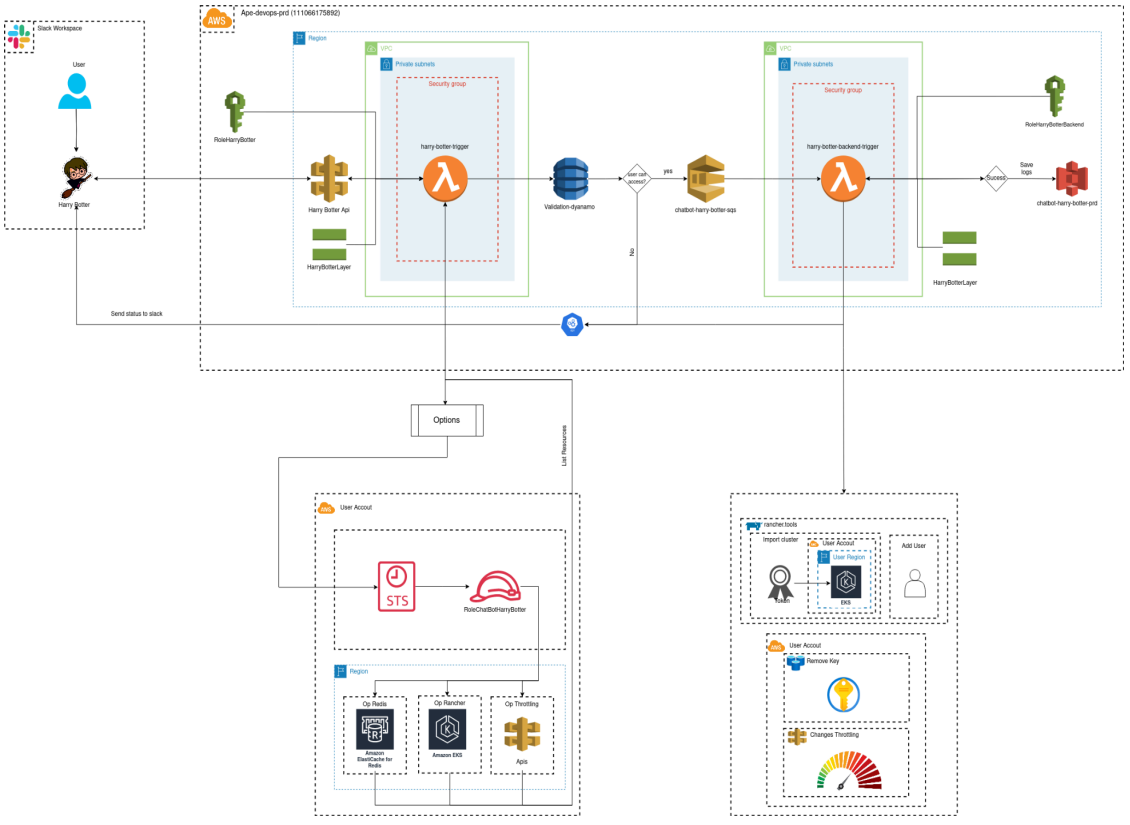
Após o usuário submeter a requisição preenchendo todos os campos do modal, a depender do serviço, é verificado em uma tabela *dynamo***(validation-harry-botter)** se aquele usuário tem ou não permissão para utilizar aquele serviço do bot. Se não houver, ele é avisado via Slack**(canal default do bot)** que não possui a permissão. Porém, se houver permissão, todas as informações que ele preencheu é enviada para uma fila SQS**(chatbot-harry-botter-sqs)**, que irá engatilhar um outro

lambda**(harry-botter-backend-trigger)** responsável por fazer o processamento da solicitação do usuário. Essa função também requer permissões específicas, como por exemplo, permissão para assumir roles em outras contas AWS, diferentes dá conta que hospeda o código do Harry Botter, quando necessitar manipular algum recurso nessa outra conta. Portanto, nessas outras contas, é preciso ter uma role que permita ser assumida pela role do Harry Botter e que possua as permissões necessárias para fazer as manipulações.

Por fim, o usuário é notificado no canal padrão do bot no Slack sobre o processamento bem-sucedido ou não da solicitação. Em caso afirmativo, os dados são armazenados em um bucket S3 **(chatbot-harry-botter-prd)**.

A arquitetura completa pode ser vista na figura 10:

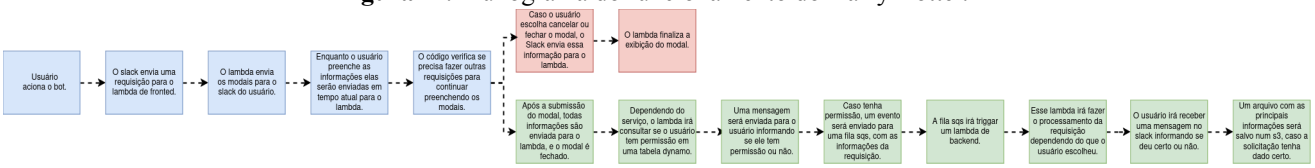
**Figura 10**. Arquitetura do Harry Botter.




Fonte: Imagem da Autora.

Em resumo, o fluxo que acontece na arquitetura acima, está explicado nos blocos da figura 11.

**Figura 11**. Fluxograma do funcionamento do Harry Botter.



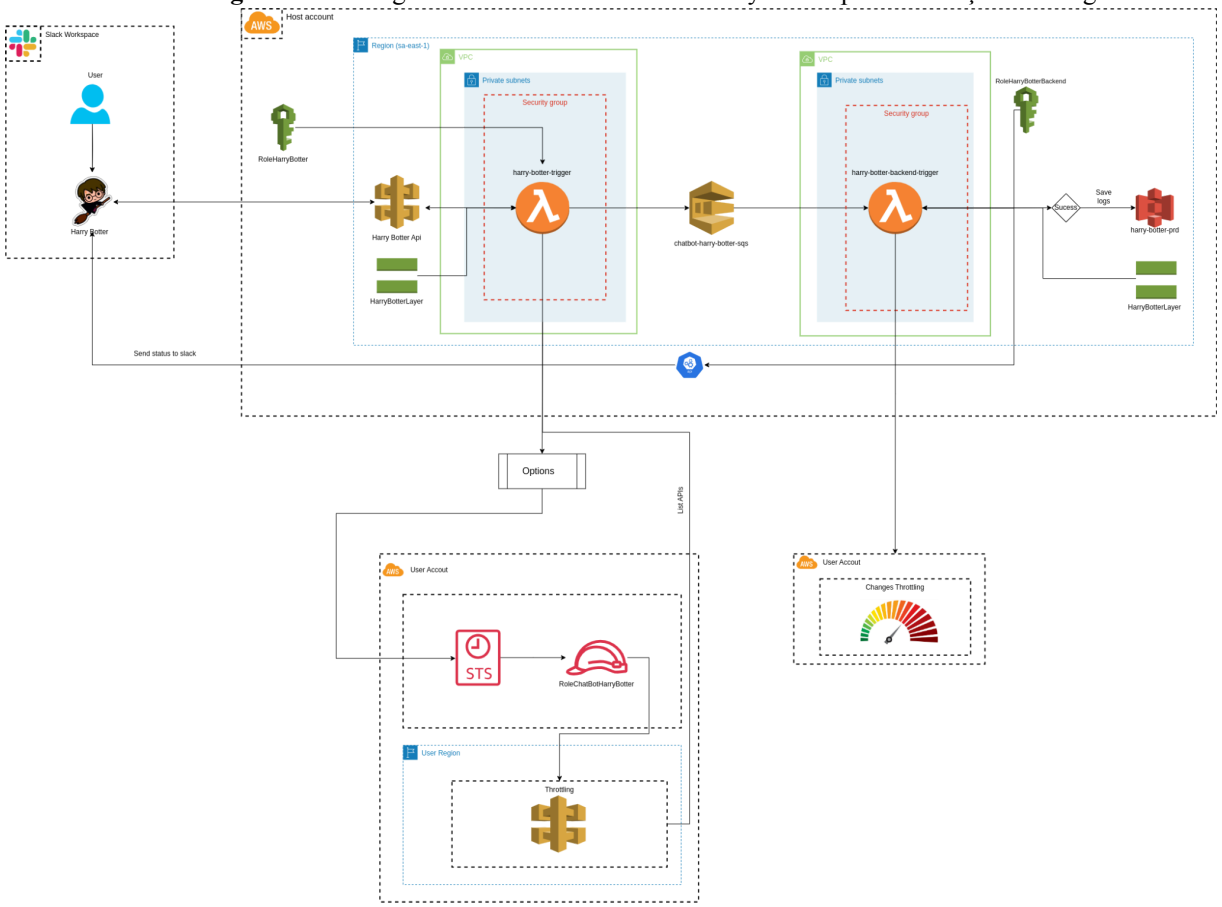

Fonte: Autora.

### 4.2.1 Infraestrutura do serviço Throttling

Quando o usuário aciona, através do slack, o serviço de throttling, uma requisição é enviada para a HarryBotterApi, que irá engatilhar o lambda harry-botter-trigger, esse lambda enviará uma requisição para o slack com o modal de exibição, onde o usuário deverá selecionar os valores iniciais de ação, região e a conta AWS onde está a api, cujo o throttling será configurado. E assim que a conta AWS for digitada, o lambda pegará todas as informações preenchidas até àquele momento, e se conectará à essa conta através da role RoleHarryBotter, buscará então, todas as APIs presentes na conta. Em tempo real, o lambda listará essas APIs na tela do usuário para ele selecionar sua API. Após selecionada, o lambda consultará mais uma vez o que foi preenchido, acessará novamente a conta AWS, e dessa vez irá buscar e exibir para o usuário todos os métodos (Method), recursos (Resource Path) e estágios (stage) dessa API, para que o usuário possa fazer sua escolha.

Após o usuário preencher todos os dados, e apertar no botão de submissão (submit), o lambda de frontend enviará os valores de todos os campos para outro lambda de backend, o harry-botter-backend-trigger, via uma fila SQS. Nesse lambda, será processada todos as informações, conforme o usuário selecionou no campo de ação. Por exemplo, caso ele tenha selecionado a opção, adicionar throttling à API Gateway, o lambda executará um script que irá acessar a conta, o método no estágio e paths da api gateway selecionada e configurar os valores limites que o usuário preencheu. Após isso, o lambda de backend enviará uma mensagem para o usuário, informando se a solicitação foi feita com sucesso ou não. E caso a solicitação tenha tido sucesso, será gerado um arquivo com todos os dados dessa requisição e ele será salvo no bucket chatbot-harry-botter-prd. Essa arquitetura pode ser vista na figura 12.

**Figura 12**. Fluxograma do funcionamento do Harry Botter para o serviço throttling.




**Fonte:** Imagem da autora.

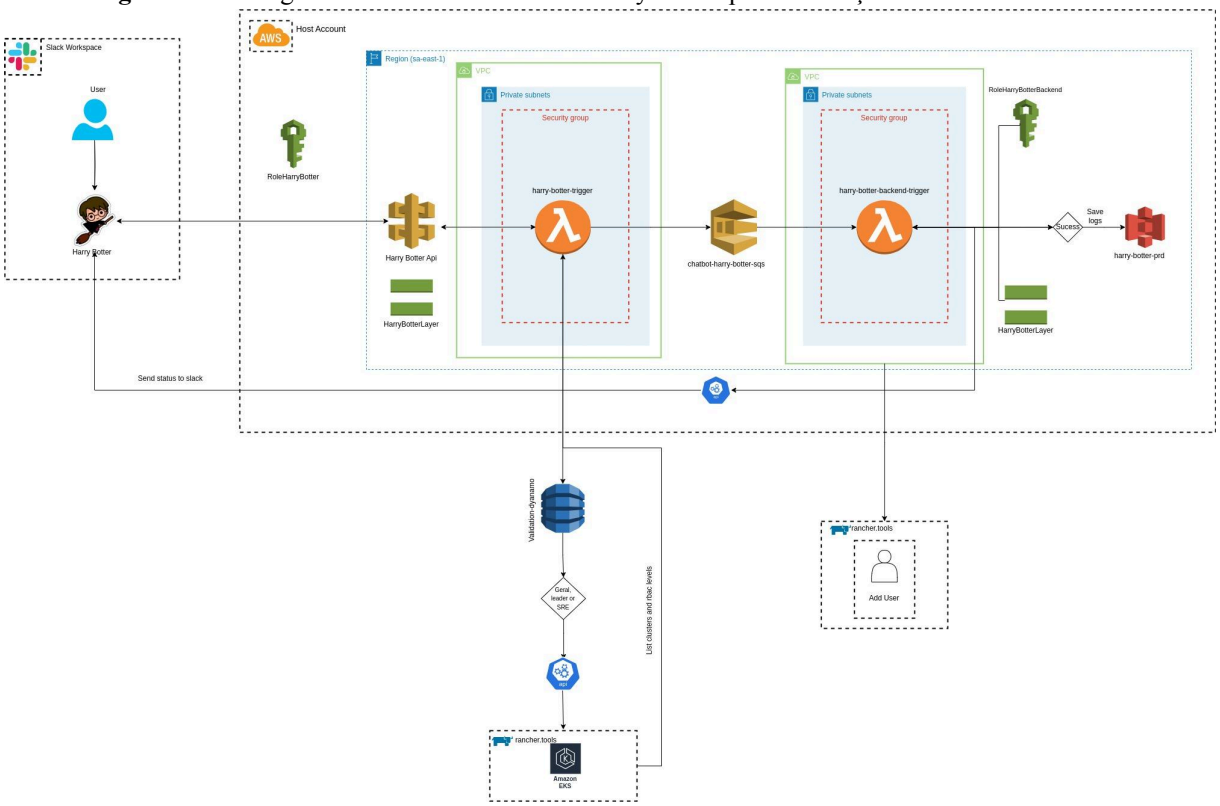
### 4.2.2 Infraestrutura do serviço Permissions

Uma vez que o serviço de Permissions é acionado, uma requisição é enviada para a HarryBotterApi, que irá engatilhar o lambda harry-botter-trigger, esse lambda enviará uma requisição para o slack com o modal de exibição, onde o usuário deverá selecionar os valores iniciais de ação. Assim que o usuário seleciona se quer adicionar ou remover permissão, o lambda irá pegar o ID desse usuário, e irá consultar na tabela validation-harry-botter se aquele usuário encontra-se na tabela, e se ele é SRE. Caso o usuário não esteja na tabela, no campo Rbac no modal irá aparecer apenas a opção ReadyOnly (apenas leitura), se o usuário estiver na tabela e não estiver como SRE, além da opção de ReadyOnly, irá aparecer a opção PowerUser(ações simples de edições), que já é um nível maior de acesso. E se o usuário for SRE, deverá aparecer uma terceira opção de ClusterOperator (ações complexas de edições), essa permissão

já têm maiores políticas de edições no Rancher. Toda essa verificação acontece em tempo real e é dessa forma que é feito o controle de acesso nos clusters.

Após a submissão, todo o processo é bem parecido com o do serviço throttling, mudando apenas o script de ação, que dessa vez ao invés de configurar throttlings em uma conta AWS, ele utilizará a API do rancher para inserir ou remover à permissão conforme os dados preenchidos pelo usuário. Essa arquitetura simplificada, pode ser vista na figura 13.

**Figura 13**. Fluxograma do funcionamento do Harry Botter para o serviço Permissions.




**Fonte:** Imagem da autora.

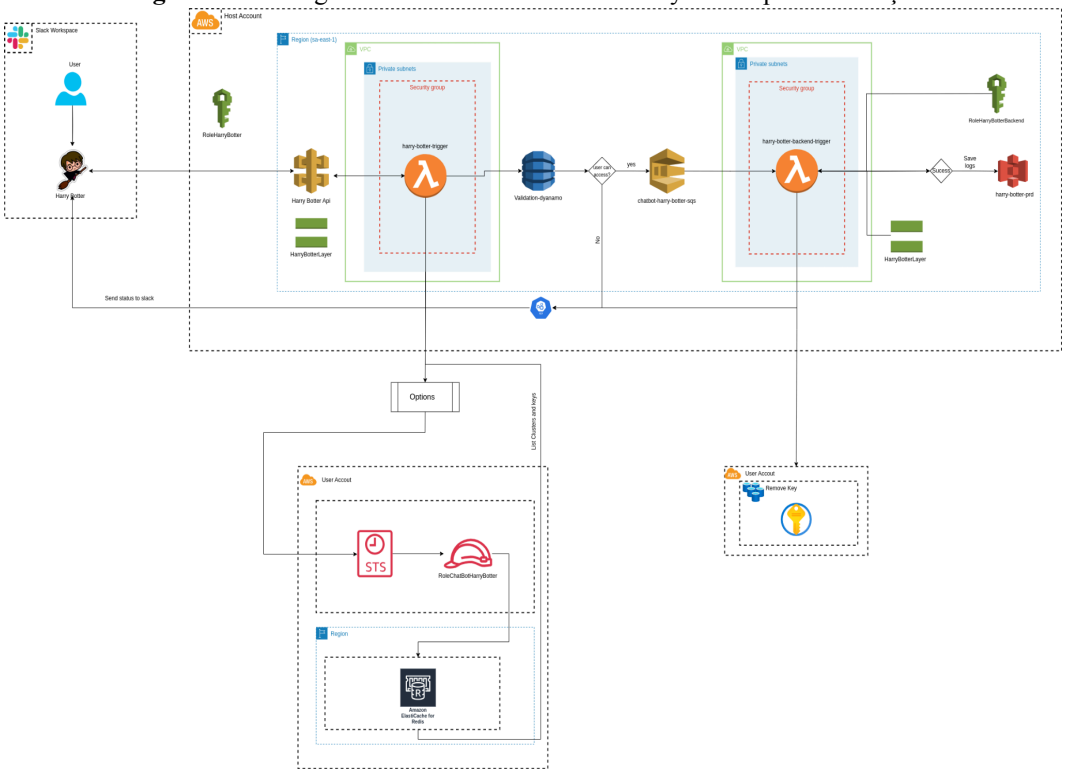
### 4.2.3 Infraestrutura do serviço Redis

Acionado o comando do Redis, de forma similar aos serviço do throttling, uma requisição é enviada para a HarryBotterApi, que irá engatilhar o lambda harry-botter-trigger, esse lambda enviará uma requisição para o slack com o modal de exibição, onde o usuário deverá selecionar os valores iniciais de ação, região e id da conta AWS onde encontra-se o cluster Redis. E assim que a conta AWS for digitada, o lambda pegará todas as informações preenchidas até àquele momento, e se conectará à essa conta através da role RoleHarryBotter, e buscará todos os clusters Redis na

conta. Em tempo real o lambda listará esses clusters na tela do usuário para ele selecionar. Após selecionado, o lambda consultará mais uma vez o que foi preenchido, acessará novamente a conta AWS, e dessa vez irá buscar todas as informações desse Redis, se conectará nele, carregará todas as chaves presentes no cluster selecionado, partir do que o usuário digitar no campo key (chave), e irá listar na tela do usuário todas as opções conforme o que o usuário for digitando.

Após preencher todos os campos e submeter o modal, o lambda de frontend irá consultar a tabela dynamo para ver o usuário em questão está inserido nela, caso não esteja, será enviado uma mensagem informando que ele não tem acesso para utilizar aquele serviço, mas se ele possuir, as informações serão enviadas para o lambda do backend, que irá se conectar no cluster, e excluir as chaves que o usuário selecionou. Caso o cluster que o usuário selecionou seja produtivo, antes de excluir a chave, o código do lambda irá consultar, através da API do jira, se o formulário de segurança foi devidamente aprovado ou não, se não tiver sido, a chave não é excluída, e o usuário é informado que o formulário precisa ser aprovado. E se tiver tudo certo, a chave será excluída. Após a exclusão da chave, o processo é o mesmo aos demais serviços. A arquitetura desse processo pode ser vista na figura 14.

**Figura 14**. Fluxograma do funcionamento do Harry Botter para o serviço Redis.

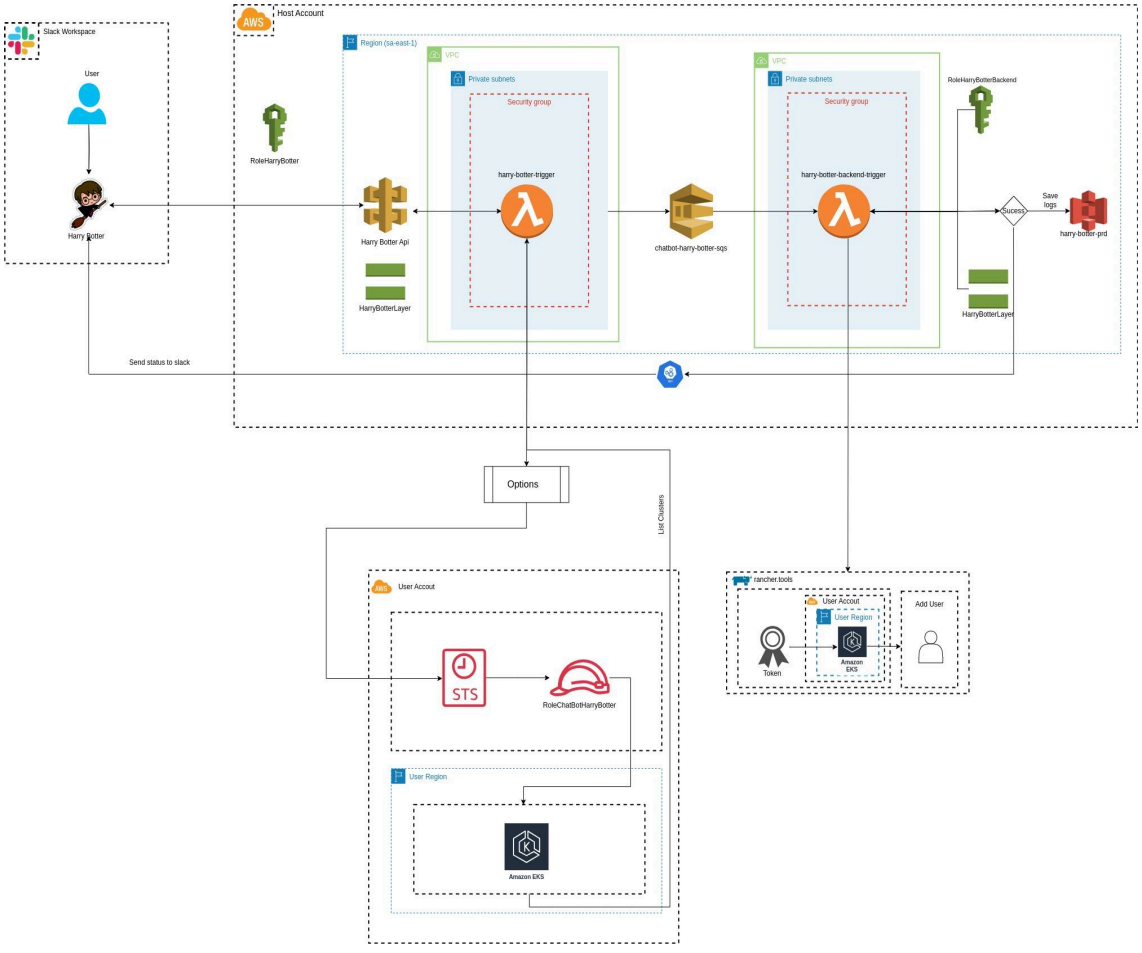



**Fonte**: Imagem da autora.

### 4.2.4 Infraestrutura do serviço Rancher

A parte inicial desse serviço funciona similar aos demais, principalmente ao do serviço Redis, porém ao invés de listar clusters Redis, essa opção irá listar para o usuário, os clusters EKS da conta selecionada. E após submetido, e enviado para o backend, o código irá acessar o Rancher, e através da API irá importar o cluster. O Rancher por sua vez, irá gerar um endpoint (URL) com um pacote de recursos que precisaram ser instalados no cluster. Em poucos segundos, o lambda irá armazenar essa url que muda a cada cluster, se conectará na conta AWS onde está o cluster, pegará todas as informações dele, e se conectará para instalar o endpoint fornecido pelo lambda. Dessa forma, a importação é concluída, e o processo final é igual aos demais serviços.

**Figura 15**. Fluxograma do funcionamento do Harry Botter para o serviço Rancher.




**Fonte:** Imagem da autora.

### 4.2.5 Custo de toda infraestrutura

Atualmente o custo mensal de toda a infraestrutura do Harry Botter é  de 0,11 centavos de dólar. Para chegar nesse valor, foram analisados os recursos que estão fora do nível gratuito da AWS, que são S3, SQS, Dynamo, Lambda e Api Gateway.

Segundo a documentação oficial de custos do S3 da AWS-J (2023), o custo de um bucket do tipo standard nos primeiros 50 TB/mês é de 0,023 USD por GB. Como o bucket, é usando apenas para logs, e não por isso não chega ter nem 1GB, ficando com o custo de 0,023, o custo mensal é de 0,023 USD.

Para o serviço de SQS, a AWS-K (2023), prevê que o primeiro milhão de solicitação por mês, é gratuito, porém a depender do tamanho e conteúdos das mensagens, uma requisição, pode se tornar quatro solicitações. Além disso, a transferência de dados por mês pode gerar o custo de 0,09 USD por GB (nos primeiros 10 TB/mês). Logo, como o Harry Botter não atinge toda essa quantidade de solicitações, e a quantidade de dados não é relativamente alta, o custo para esse serviço é baixo. Outro serviço que poderia gerar custos na infraestrutura do Harry Botter, o DynamoDb, saí sem custos consideráveis. Pois, segundo AWS-L (2023), para gravação de dados, o custo é de 1,25 USD por milhão de unidades de solicitação de gravação. Já para solicitações de leitura, o valor é de 0,25 USD por milhão de unidades de solicitação. Além disso, para os primeiros 25 GB armazenados por mês são gratuitos usando a classe de tabela DynamoDB Standard. Como o Harry Botter ainda não possui uma quantidade de solicitações que ultrapassem essas margens, os custo para o DynamoDB é nulo.

Segundo a AWS-M (2024), os primeiros 25 GB armazenados por mês são gratuitos usando a classe de tabela DynamoDB Standard. Sabendo disso, concluímos que como o consumo do Harry Botter não ultrapassa esse valor, o custo para o lambda, pelo menos enquanto não ultrapassar essas medidas, será de 0 USD. A Api Gateway, último recurso precificado da arquitetura do bot, tem como nível gratuito, um milhão de chamadas recebidas, e se esse valor for excedido, será precificado conforme a quantidade de solicitações mensais, por exemplo, até 333 milhões de solicitações mensais, o preço será de 3,50 USD por milhão (AWS-N, 2023). Ou seja, como o Harry Botter ainda não bateu um milhão de requisições, seu ainda não foi gerado custos para esse recurso, porém, uma vez ultrapassado esse valor, será gerado custo por mês a partir dos 333 milhões de requisições.

Diante do exposto, e baseando-se na arquitetura e na quantidade de requisições que o bot recebe mensalmente, dificilmente seu custo irá aumentar significativamente. Além disso, esse custo obtido, é derivado da taxa do S3 com SQS acumuladas a cada mês durante, após a instalação do mesmo em ambiente produtivo. Os demais recursos, portanto, não tiveram custos desde a criação do bot, pois não atingiram os limites mínimos.

## 4.3 Código

Atualmente, o código do Harry Botter está dividido nos repositórios do frontend e backend. Eles possuem um arquivo chamado main.py que possui a função lambda_hander, compartilhada para todos os serviços. Também possui uma pasta service, dentro dela estão os arquivos com funções gerais, que podem ser utilizadas para qualquer serviço, como por exemplo a função que envia uma requisição para o slack exibir um modal, atualizar o modal, enviar mensagens, ou até mesmo a função que pega os valores que estão sendo preenchidos em tempo real.

Além disso, possui pastas com arquivos específicos para cada serviço, as quais são: Permissions, Redis, Throttling, Rancher. Nessas pastas estão, por exemplo, as funções com os códigos que consultam as APIs em outras contas, que se conectam nos clusters Redis e buscam as chaves presentes neles, o código que faz uma requisição para o Rancher com o intuito de consultar quais os clusters estão inseridos nele, e entre outros códigos. Para realizar essas automações, são utilizadas algumas bibliotecas do python, que são: boto3, redis, requests, json, os, kubernetes. Ademais, nesses repositórios também estão inseridos toda a infraestrutura, que foi desenvolvida por meio de código, em Cloudformation, serviço da AWS que permite que a infraestrutura seja criada por IaaC e organizadas em pilhas.

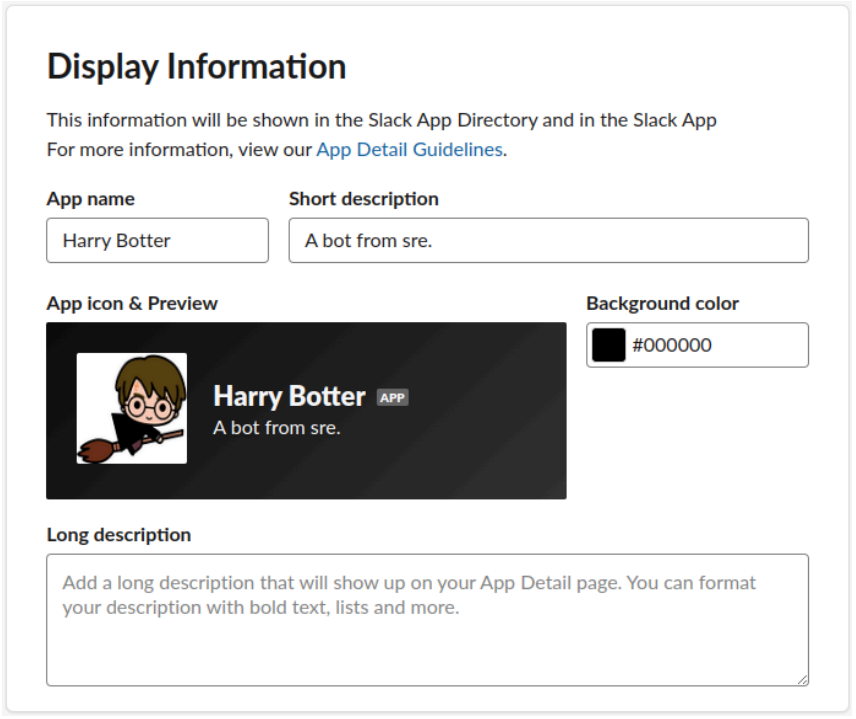
## 4.4 Slack Developer

O Slack possui uma ferramenta para que desenvolvedores possam criar seus próprios aplicativos dentro do Slack. Essa funcionalidade tem o intuito de Desenvolver com a plataforma do Slack permite conectar seu workspace a ferramentas, fontes de dados e processos que fazem sua organização fluir sem empecilhos (Slack, 2024).

Por tanto, foi criado um o aplicativo no Slack app, e por default ele vai criar a App Credentials, assim, foi necessário configurar apenas o **Display Information,** com o nome,

uma descrição, cor de background e imagem de perfil do Harry Botter. Ficando da seguinte maneira:

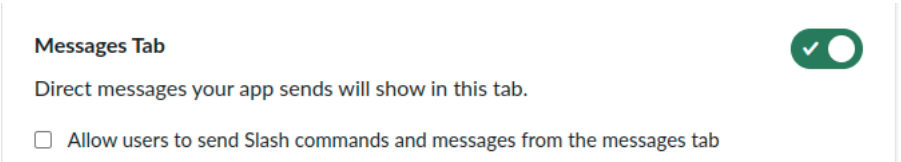
**Figura 16.** Informação sobre o aplicativo do Harry Botter.




Fonte: Imagem da Autora.

Em App Home, foi habilitado para o bot aparecer sempre como online e para as mensagens serem exibidas no chat do app do Harry Botter.

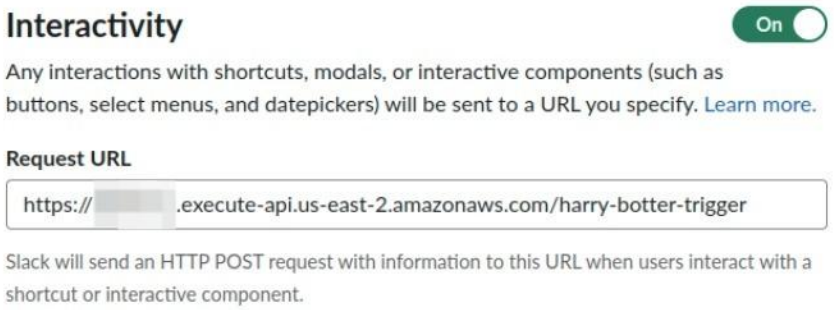
**Figura 17.** Configuração de disponibilidade do Harry Botter.




Fonte: Imagem da Autora.

Foi habilitado o **Interactivity** e passado em **Request URL** o endpoint da api gateway que foi criado para engatilhar o lambda function com o código da automação.

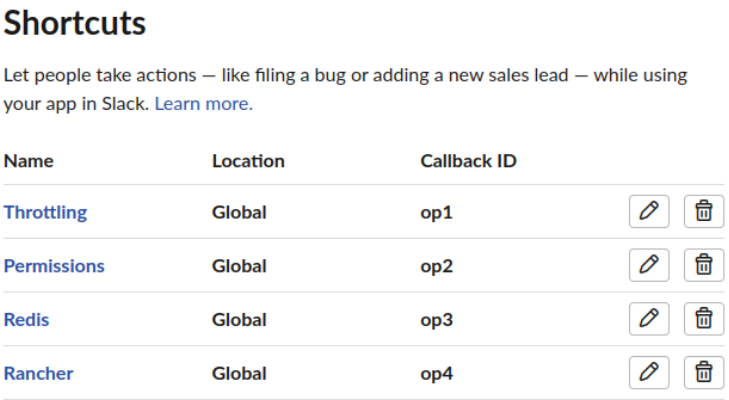
**Figura 18**. Configuração de webhook.




Fonte: Imagem da Autora.

No campo **Shortcuts,** é definido os serviços (opções) que o bot irá fornecer e um Callback ID, que é necessário para programar qual ação o bot deverá fazer quando o usuário escolher um shortcut.

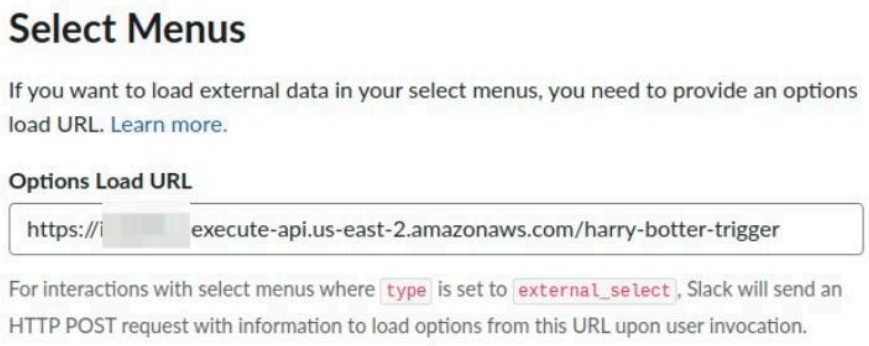
**Figura 19**. Configuração dos serviços.




Fonte: Imagem da Autora.

Em **Select Menus,** foi passado um endpoint de uma api para o carregamento de menus externos(que precisam fazer uma requisição fora do lambda), como por exemplo o menu de clusters do Permissions, que faz uma requisição para o Rancher listar os clusters. No caso do Harry Botter, a api é a mesma que colocamos anteriormente.

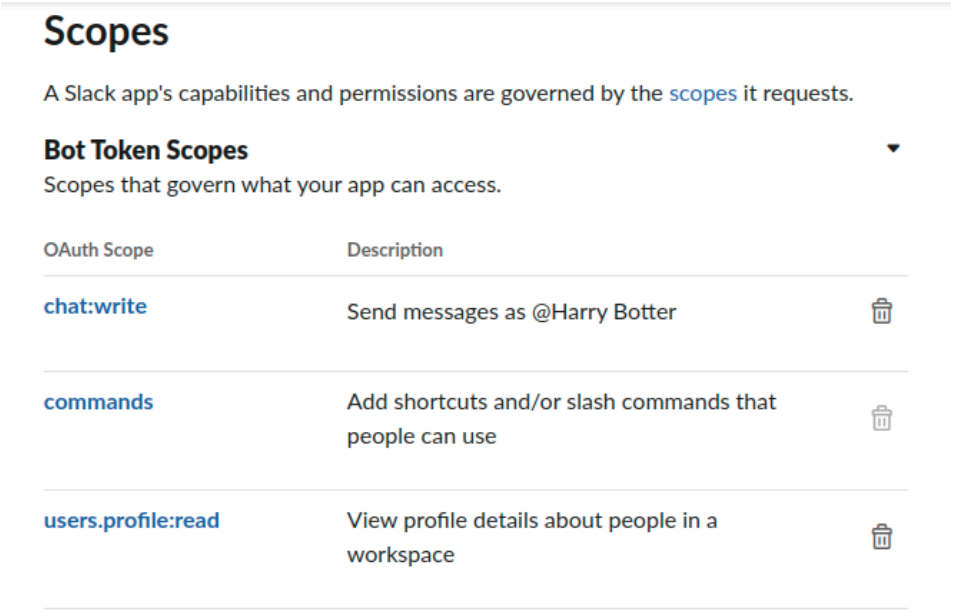
**Figura 20.** Configuração do Menu Externo.




Fonte: Imagem da Autora.

Por fim, em **Scopes**, será colocado as permissões que o nosso bot terá do Slack, que são:

**Figura 21.** Configuração das permissões do bot.




Fonte: Imagem da Autora.

Após fazer as configurações acima, o bot foi instalado no Workspace da Empresa.

# 5. RESULTADOS E MÉTRICAS DE DESEMPENHO

Segundo Muniz, Oliveira e Muller(2023), uma métrica é uma medida quantificável usada para avaliar o estudo de um processo, uma ação ou uma estratégia específica, ou seja, é a medida para acompanhar o que dá certo ou não no trabalho. Sabendo disso, as métricas aqui analisadas foram retiradas do log group (cloudwatch) do lambda onde está o código do Harry Botter, bem como, dos logs armazenados no bucket s3 após uma ação realizada com sucesso. Essas métricas irão determinar o sucesso e efetividade do bot.

No cenário desse estudo de caso, serão analisados quantas vezes os modais foram abertos, quantas vezes os modais foram enviados, qual a taxa de sucesso e erro, qual o tempo e o custo operacional;

## 5.1 Quantas vezes os modais foram abertos

Essa métrica, diz respeito à quantidade de vezes que um usuário ao menos cogitou utilizar o Harry Botter, fazendo uma requisição, tendo enviado ela ou não. A query utilizada para obter a quantidade desse tópico foi:

**Figura 22.** Query de quantas vezes o modal foi aberto.

```
1  fields @timestamp, @message, @logStream, @log
2  | filter @message like 'shortcut' and @message like ‘op1’
3  | sort @timestamp desc
4  | stats count(*) as total
```

Fonte: Imagem da Autora.

Na figura 22, contamos a quantidade total em que as palavras shortcut ( método padrão utilizado pelo Slack para informar que uma requisição foi aberta). E também, procuramos pela opção escolhida pelo usuário ( op1 é o serviço throttling, op2 é o serviço Permissions, op3 é o serviço redis, op4 é o serviço do Rancher).

Após essa definição, a query foi executada e os seguintes dados foram obtidos para análise:

**Tabela 01.** Tabela de resultados métrica 1.

| Métrica/Serviço | Throttling | Permissions | Redis | Rancher |
|---|---|---|---|---|
| **Quantidade de Abertura** | 135 | 2865 | 591 | 144 |

Fonte: da Autora.

Na tabela 01, é possível perceber que a linha refere-se à métrica aqui analisada, e em cada coluna é o valor da quantidade total em que os usuários chegaram a abrir o bot para cada serviço.

## 5.2 Quantas vezes os modais foram enviados

Para saber a quantidade de requisições que foram submetidas, ou seja, enviada para o harry botter processar, bastou substituir o ‘shortcut' da query por ‘view_submission' e também os valores do op1, op2, op3 e op4. Que foram substituídos por ‘throttling’, ‘permissions’ , ‘redis’ ou ‘rancher’. Por exemplo:

**Figura 23.** Query de quantas vezes o modal foi enviado.

```
1 fields @timestamp, @message, @logStream, @log
2 | filter @message like 'view_submission' and @message like ‘permissions’
3 | sort @timestamp desc
4 | stats count(*) as total
```

Fonte: Imagem da Autora.

Após essa definição, a query foi executada e os seguintes dados foram obtidos para análise:

**Tabela 02.** Tabela de resultados métrica 2.

| Métrica/Serviço | Throttling | Permissions | Redis | Rancher |
|---|---|---|---|---|
| **Quantidade de. Submissões** | 87 | 2050 | 289 | 72 |

Fonte: Imagem da Autora.

Na tabela 02, é possível perceber que a linha refere-se à métrica em questão, e em cada coluna é o valor da quantidade total em que os usuários chegaram a submeter o modal do bot para cada serviço.

## 5.3 Qual a taxa de sucesso e erro

Para saber as taxas de sucesso, foi analisado os logs armazenados no bucket s3 o chatbot-harry-botter-prd, dentro desse bucket existem pastas para cada serviço, e dentro de

cada uma delas, um log específico. Uma vez que, esse bucket armazena todas as requisições executadas com sucesso, só bastou buscar a quantidade total de logs armazenados.

Para saber as taxas de erro, bastou subtrair a quantidade de requisições submetidas de cada serviço, pela quantidade de requisições com sucesso.

Após essa definição, a query foi executada e os seguintes dados foram obtidos para análise:

**Tabela 03.** Tabela de resultados métrica 3.

| Métrica/Serviço | Throttling | Permissions | Redis | Rancher |
|---|---|---|---|---|
| **Taxa de Sucesso** | 86 | 2026 | 218 | 27 |
| **Taxa de Erro** | 1 | 24 | 71 | 45 |

Fonte: Imagem da Autora.

Na tabela acima, é possível perceber que cada linha refere-se às métricas em questão, e em cada coluna é o valor da quantidade total sucessos e erros das requisições enviados pelos usuários. Nota-se também, que a quantidade de erros, seja por falta de conhecimento do usuário, ou por alguma falha no processo, é consideravelmente baixa, provando assim, que o bot possui uma operação. Percebe-se também que o serviço do Rancher é aquele com uma maior proporção de erro, isso pode estar atribuído principalmente ao fato de que para que esse serviço funcione, o cluster EKS em questão precisa atender alguns requisitos, que passam muitas vezes despercebidos pelos usuários.

## 5.4 Tempo de execução

A partir de experiências reais dos usuários, e de testes realizados pelo time de SRE, conseguimos obter um tempo médio de utilização para cada serviço do bot. Esse tempo também considerou o primeiro contato de um usuário com bot, precisando ler o manual de utilização. Pelo bot ser de fácil entendimento, o principal cenário simulado e cronometrado, foi o cenário em que o usuário já possuía algum conhecimento sobre as informações necessárias para passar nos modais.

Além disso, foi mensurado quanto tempo um SRE levaria para executar manualmente cada um desses serviços. Fazendo isso, é possível estimar a economia em termos monetários, quanto de tempo, proporcionadas pela automatização. Podendo, também, ser considerado todos os benefícios lucrativos para a empresa, como a redução de erros, redução de toil, o

aumento da eficiência operacional, maior autonomia para os times de engenharia e redução de acionamento. Permitindo compreender não apenas os custos diretos, mas também os impactos positivos que a implementação do bot pode trazer para a organização.

Para calcular o tempo médio de execução do bot, foi considerado que um usuário iria utilizá-lo pela primeira vez, após ter lido o manual de funcionamento. Já para contabilizar o tempo de execução manual, foi feito todos os procedimentos anteriores à disponibilização do bot, ou seja, no antigo padrão. Por exemplo, para fazer alterações em throttling, primeiramente era feito um código iac em cloudformation, em seguida esse código era versionado no github, que por sua vez engatilhava uma pipeline de entrega contínua, passando por vários passos de compilação e validações de segurança, até iniciar a criação do recurso na AWS. Já para calcular o tempo de execução manual para adicionar permissão de um usuário do Rancher, foi levado em condieração que o SRE teria que se conectar à uma VPN ( Virtual Private Network), ou seja, o SRE primeiro teria que se conectar à uma rede privada, acessar o dashboard do Rancher, procurar dentre os muitos clusters da empresa qual era o cluster que foi solicitado, e inserir diretamente pelas configurações o usuário.

Já para excluir uma chave em um cluster Redis, foi considerado que o SRE, teria também que se conectar à uma VPN, acessar a conta AWS onde encontra-se o cluster que possui a chave a ser excluída, buscar todas as informações de login, se conectar ao cluster via terminal, e executar o comando de exclusão de chaves para excluir cada chave de uma única vez. O processo manual de inclusão de cluster no Rancher, no entanto, é bem parecido com o de excluir chaves no cluster Redis. O SRE teria que se conectar em uma VPN, ir na conta AWS, buscar informações de login no cluster EKS, acessar o dashboard do Rancher, incluir o cluster lá para que um token e uma url com os pacotes de importação fossem criados, e posterior a isso, o SRE teria que executar o token e pacote no cluster e aguardar a importação ser concluída.

Após essa definição e execução de processos, foi feita uma simulação onde os seguintes dados foram obtidos para análise:

**Tabela 04.** Tabela de resultados métrica 4.

| Métrica/Serviço | Throttling | Permissions | Redis | Rancher |
|---|---|---|---|---|
| **Tempo médio de Execução do Bot** | 2 minutos | 40 segundos | 1,1 minutos | 30 segundos |
| **Tempo de Execução Manual** | 20 minutos | 8 minutos | 15 minutos | 10 minutos |

Fonte: Imagem da Autora.

Na tabela 04, é possível perceber que a linha refere-se às métricas em questão, e em cada coluna é o valor da quantidade média de tempo de utilização do bot, e do tempo para ação manual de cada serviço que o Slackbot proporciona. Pode-se perceber, ainda, que a quantidade de tempo de utilização do bot é consideravelmente menor do que o tempo para fazer a mesma ação manualmente.

## 5.5 Custo Operacional

Considerando o baixo custo para manter o Harry Botter, e também, na redução de tempo para realizar as ações, conseguimos estimar qual seria a economia resultante disso e do menor acionamento do time de SRE

Foi utilizado o site Glassdoor, que é um site de avaliação de empresas e cargos, para consultar o valor médio do salário de um analista de SRE e para um Desenvolvedor (usuário do bot). Segundo o site, no ano de 2024, o salário de um SRE é de 7 à 13 mil reais e para um desenvolvedor é de 3 à 8 mil reais, por tanto, os salários médios para essa estimativa serão de 10 mil reais e 5.500 reais. Considerando também que a quantidade de horas trabalhadas é de 40 horas semanais, temos que o salário por hora é de um profissional de SRE é de 62,50 reais, e de um desenvolvedor é de 34,37 reais.

Sabendo disso, e sabendo a quantidade de requisições com sucesso, e o tempo de execução da ação no bot e manualmente, realizamos um cálculo para saber o custo operaciona (CO) de execução, em horas, do bot e manualmente. Que consistiu em:

$$CO = salário/hora \times (taxa\ de\ sucesso \times qt.\ de\ execução\ em\ horas)$$

Considerando que o bot é utilizado principalmente pelos desenvolvedores, o custo operacional de execução do bot foi realizado com os valores médios do salário dos desenvolvedores, já os de execução manual, foi calculado com o salário médio dos SRE. Já para calcular a Economia Total (ET), foi realizado o seguinte cálculo.

$$ET = CO\ bot - CO\ manual - (Custo\ da\ infraestrutura\ em\ reais \div 4)$$

Estimamos assim, que o resultado do custo operacional durante um ano foi:

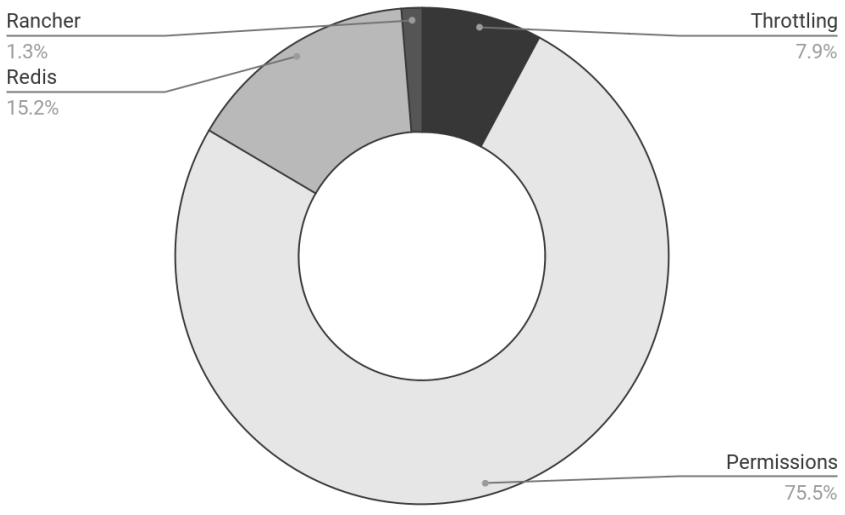
**Tabela 05.** Tabela de resultados métrica 5.

| Métrica/Serviço | Throttling | Permissions | Redis | Rancher |
|---|---|---|---|---|
| **Custo Operacional Exec. Bot** | 98,53 Reais | 663,71 Reais | 137,36 Reais | 7,73 Reais |
| **Custo Operacional Exec. Manual** | 1.791 Reais | 16.883,33 Reais | 3.406,25 Reais | 281,25 Reais |
| **Custo da infraestrutura** | 1,32 Dólares/ano 6,89 Reais ( em maio de 2024) | | | |
| **Economia Total** | 1.690,78 Reais | 16.217,92 Reais | 3.267,2 Reais | 271,8 Reais |

Fonte: Imagem da Autora.

Na tabela acima, podemos notar o quão significativamente baixo é o custo operacional do bot em comparação com o custo da execução manual. Isso evidencia a considerável economia que a empresa obtém ao implementar automações para tarefas específicas e repetitivas. Além disso, se considerarmos todas as horas que um analista de SRE pouparia ao evitar operações manuais, podendo direcionar seu foco para outros projetos, implementação de correção de problemas, desenvolvimento ou até mesmo prevenir crises, invasões ou outras situações que poderiam gerar prejuízos ou aumentar os lucros da empresa. Fica claro o valor estratégico das automações no ambiente empresarial.

**Figura 24.** Gráfico com porcentagem da economia operacional.




Fonte: Imagem da Autora.

## 5.6 Análise

Segundo Muniz, Oliveira e Muller (2023), a autonomia dos times e a facilidade dos comandos se tornam muitas vezes o ponto forte da adoção de bots, trazendo velocidade para os times. Tendo em vista isso, e que um dos principais papéis do SRE é trabalhar em estreita colaboração com equipes de produto, engenharia, operações e segurança para implementar mudanças controladamente, antecipar possíveis problemas, otimizar ações repetidas, e garantir que as aplicações funcionem de maneira confiável e eficiente. Tudo isso priorizando a confiabilidade e a excelência operacional para proporcionar uma experiência segura e eficiente. E com base nos dados coletados em cada uma das métricas, podemos observar que este bot está tendo um impacto positivo na empresa em que está operando.

Percebemos pela quantidade de acessos que o bot foi bem aderido pelos times de engenharia e produto da empresa, mudando assim, um pouco da cultura organizacional, já que esses times passaram a ter maior autonomia para gerenciar suas aplicações. Confirmou-se então que, tornar a execução de processos o mais simples e o mais fácil quanto possível, é a melhor alternativa para promover mudanças na cultura organizacional (MUNIZ, OLIVEIRA e MULLERO, 2023). Além disso, comprovou-se ter uma boa otimização de tempo nas operações, visto que o tempo de utilização do bot é até mais de 10 vezes menor do que o tempo que um SRE levaria para fazer a mesma ação manualmente.

Outro benefício encontrado na utilização do bot é que os profissionais de SRE podem direcionar seu foco para outros projetos estratégicos, já que passam a ser menos acionados. Além do mais, em caso de incidentes, a resolução pode ser feita de forma mais rápida. Com menos acionamentos do time de SRE, reduzindo toils, além dos ganhos operacionais, a empresa também obtém economia monetária, ampliando os benefícios proporcionados pela implementação do bot. Segundo Muniz, Oliveira e Muller(2023), para eliminar toils, é preciso que os seguintes passos sejam seguidos: Eliminar dúvidas dos solicitantes, padronizar e automatizar processos do time de SRE, tornar o processo self-service, eliminação de processos. Por tanto, podemos dizer que o toil dessas ações foram eliminadas por completo, visto que ocorreu a eliminação de dúvidas dos usuários através da disponibilização de manuais de utilização do bot, os processos anteriormente foram padronizados e automatizados, e disponibilizados para os times de engenharia utilizarem conforme precisarem, ou seja, self-service. Além disso, eliminamos por completo os processos retirando-o das mãos do time de SRE, evitando assim, redução de erros humanos. Dessa forma, provou-se também que, a cultura do Chat Bots é uma grande aliada na busca pela

eficiência de processos, onde times técnicos podem interagir de forma mais autônoma e consistente, evitando erros operacionais e processos repetitivos desnecessários (MUNIZ, OL

# 6. CONSIDERAÇÕES FINAIS

A pesquisa desenvolvida observou do ponto de vista do SRE, os benefícios da implementação de um Slackbot para ações na nuvem em uma empresa de tecnologia. Os objetivos do estudo foram alcançados, visto que houveram diversos benefícios após o time de SRE disponibilizar o Harry Botter para os times de engenharia e produtos. Com isso, ficou claro que manter um bot operacional de maneira eficiente pode proporcionar uma considerável autonomia para equipes com pouco conhecimento em infraestrutura em nuvem, além de mitigar problemas de padronização e otimizar os tempos de operação comuns na empresa. Além disso, percebemos que como o bot teve bastante aderência pelos times de engenharia e produto, não houve muitos desafios.

Nesse sentido, observou-se que, apesar de alguns desafios, como requisitos específicos para determinados serviços, a implementação eficaz dessas estratégias resultou em uma operação mais eficiente e confiável. Foi possível ver, principalmente como serviço de Adicionar/Remover Permissões, que quantidade de vezes que os times de engenharia chegaram ao menos a cogitar utilizar o bot, com a quantidade de vezes que eles submeteram as requisições, não tiveram uma diferença muito grande. Esses valores comprovaram que após a implementação os times de SRE tiveram maior liberdade para focar em tarefas estruturantes. Além disso, após a implementação do bot houve uma redução considerável de erros, indicando que o Harry Botter operou de maneira confiável, apesar de alguns desafios, como os relacionados ao serviço Rancher, que exigia requisitos específicos de cluster EKS. Além dessa redução de erros, um dos principais benefícios provenientes do bot, foi o custo operacional, que proporcionou à empresa uma redução, segundo a estimativa resultante da análise das métricas, de  mais de 20 mil reais. Esses resultados destacam não apenas a eficácia do bot em melhorar a operação e reduzir erros, mas também seu impacto financeiro positivo para a organização.

Ao final do estudo, foi possível concluir que ele ajuda a melhorar nossa compreensão de como os bots podem ser úteis no dia-a-dia de uma empresa. Os resultados mostram que os bots não apenas podem resolver problemas de padronização e melhorar a eficiência das operações, mas também podem dar mais autonomia a equipes que não têm muita experiência com infraestrutura em nuvem. Além disso, é possível notar que eles também ajudam a reduzir os erros, o orçamento da empresa e os custos operacionais.

Essas descobertas são importantes para empresas de tecnologia que querem ser mais eficientes e usar melhor seus recursos. Elas mostram como a automação, usando bots, pode trazer melhorias significativas. Ademais, destacam que não é só questão técnica, mas também organizacional. É importante que as diferentes equipes trabalhem juntas para implementar essas soluções de forma integrada. No fim das contas, este estudo ajuda a melhorar as práticas de operações em nuvem, oferecendo ideias úteis para tornar os sistemas mais eficientes e confiáveis em ambientes tecnológicos complexos.

Diante de tais considerações, recomenda-se para trabalhos futuros: Criar um monitor para que todas essas métricas sejam exibidas de formas reais e automáticas; Fazer melhorias no serviço do Rancher, visto que esse é o que possui a maior taxa de erros; Criar um novo serviço de criação de clusters EKS, para que os times de engenharia tenham ainda mais autonomia.

# REFERÊNCIAS


ANDRADE, L. **Hands-on: como desenvolver uma infraestrutura como código usando AWS CloudFormation.** Medium, 2022. Disponível em: <https://medium.com/itautech/hands-on-como-desenvolver-uma-infraestrutura-como-c%C3%B3digo-usando-aws-cloudformation-af85e6fd4d01#:~:text=Acesse%20o%20Console%20AWS%20e>. Acesso em: 15 de abril 2024.

ANDREI, L. **O Que é GitHub e Para Que é Usado?** hostinger. 2023. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-github>. Acesso em 10 de fevereiro de 2024.

ARAÚJO, C. V. F. NASCIMENTO, L. D. F. Modernização de software: Um estudo de caso sobre a aplicação de práticas de DevOps. UNB, 2022. Disponível em: <https://bdm.unb.br/bitstream/10483/34517/1/2022_CaioViniciusAraujo_LucasDoNascimento_tcc.pdf>. Acesso em: 23 de abril de 2024.

AWS-A. **O que é Infraestrutura como código?** — Explicação sobre a IaC — AWS. 2023. Disponível em: <https://aws.amazon.com/pt/what-is/iac/#:~:text=A%20automa%C3%A7%C3%A3o%20%C3%A9%20uma%20meta>. Acesso em: 23 Março. 2024.

AWS-B. **O que é AWS?** Como funciona o Amazon Web Services, 2023. Disponível em: <https://aws.amazon.com/pt/what-is-aws/?nc1=f_cc>. Acesso em: 14 de abril 2024.

AWS-C. **O que é o Amazon API Gateway?** - Amazon API Gateway, 2024. Disponível em: <https://docs.aws.amazon.com/pt_br/apigateway/latest/developerguide/welcome.html>. Acesso em: 13 de abril de 2024.

AWS-D. Amazon CloudWatch. Disponível em: <https://aws.amazon.com/pt/cloudwatch/>. Acesso em: 4 de abril de 2024.

AWS-E. **O que é o Amazon EKS? - Amazon EKS, 2024**. Disponível em: <https://docs.aws.amazon.com/pt_br/eks/latest/userguide/what-is-eks.html>. Acesso em: 05 de abril de 2024.

AWS-F. **O que é o AWS Lambda?** - AWS Lambda, 2024. Disponível em: <https://docs.aws.amazon.com/pt_br/lambda/latest/dg/welcome.html>. Acesso em: 04 de abril de 2024.

AWS-G. Amazon Elasticache for Redis, 2023. Disponível em: <https://aws.amazon.com/pt/elasticache/redis/>. Acesso em: 04 de abril de 2024.

AWS-H . **O que é o Amazon Simple Queue Service?** - Amazon Simple Queue Service, 2024. Disponível em: <https://docs.aws.amazon.com/pt_br/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html>. Acesso em: 06 de abril 2024.

AWS-I. **Serviços de computação em nuvem - Amazon Web Services (AWS)**. Disponível em: <https://aws.amazon.com/pt/>. Acesso em: 10 de fevereiro de 2024.

AWS-J. **Definição de preço do Amazon S3.** 2023**.** Página inicial. Disponível em: <https://aws.amazon.com/pt/s3/pricing/>. Acesso em: 10 de fevereiro de 2024.

AWS-K. **Definição de preço do Amazon SQS**. 2023. Página inicial. Disponível em: <https://aws.amazon.com/pt/sqs/pricing/>. Acesso em: 10 de fevereiro de 2024.

AWS-L. **Preço para capacidade sob demanda**. 2023. Página inicial. Disponível em: <https://aws.amazon.com/pt/dynamodb/pricing/on-demand/>. Acesso em: 10 de fevereiro de 2024.

AWS-M. **AWS Lambda.** 2024. Página inicial. Disponível em: <https://docs.aws.amazon.com/pt_br/whitepapers/latest/how-aws-pricing-works/aws-lambda.html>. Acesso em: 10 de fevereiro de 2024.

AWS-N. **Preço do Amazon API Gateway**. 2023. Página inicial. Disponível em: <https://aws.amazon.com/pt/api-gateway/pricing/>. Acesso em: 10 de fevereiro de 2024.

BORTOLETTO, F. MORAES, F. **Uso de IAM Roles no desenvolvimento de aplicações na AWS**. AWS, 2022. Disponível em: <https://aws.amazon.com/pt/blogs/aws-brasil/uso-de-iam-roles-no-desenvolvimento-de-aplicacoes-na-aws/>. Acesso em: 05 de abril. 2024.

BUFFA, L. H. **TORNANDO ACESSÍVEL A CULTURA DEVOPS A PEQUENAS EMPRESAS E STARTUPS**. PUC Goiás, 2021. Disponível em: <https://repositorio.pucgoias.edu.br/jspui/handle/123456789/2861>. Acesso em: 22 de abril 2024.

CASTELLS, Manuel, 1942- C344g **A galáxia da internet: reflexões sobre a internet, os negócios e a sociedade** / Manuel Castells; tradução Maria Luiza X. de A. Borges; revisão Paulo Vaz. – Rio de Janeiro: Zahar, 2003.

CHELES, P. **DynamoDB: o que é e como usar o banco de dados da Amazon?** UDS, 2022. Disponível em: <https://uds.com.br/blog/aws-amazon-dynamodb-o-que-e/>. Acesso em: 06 de abril 2024.

DAVID, D. J. ALVEZ. C. A. M. NUNES, R. R. OLIVEIRA, R. M. **Benefícios e Riscos do Uso da Computação em Nuvem no Setor Público: uma análise baseada em artigos disponibilizados em bases dados acadêmicas de 2017 a 2021.** RISTI, 2022. Disponível em: <https://www.researchgate.net/publication/362918198_Beneficios_e_Riscos_do_Uso_da_Computacao_em_Nuvem_no_Setor_Publico_uma_analise_baseada_em_artigos_disponibilizados_em_bases_dados_academicas_de_2017_a_2021>. Acesso em: 23 de Março de 2024.
DOCKER. **What is a Container?** Docker, 2024. Disponível em: <https://www.docker.com/resources/what-container/>. Acesso em: 13 de abril 2024.

FREITAS, O. NETO, L. **COMPUTAÇÃO EM NUVEM: UMA BREVE REVISÃO BIBLIOGRAFICA**. **energia**, v. 3, 2023. Disponível em: <https://repositorio.ufersa.edu.br/items/5846175d-de61-42ad-8d38-7812a647ae9c>. Acesso em: 13 de abril 2024.

FERRARI, P. E. **Benefícios do Kubernetes: quais são os principais?** Programathor, 2021. Disponível em: <https://programathor.com.br/blog/beneficios_kubernetes/>. Acesso em: 14 de abril 2024.

Google - Site Reliability Engineering. Disponível em: <https://sre.google/sre-book/table-of-contents/>. Acesso em: 22 de abril 2024.

IUNES, P. J. P. **Modelo NIST ⋆** Colaborae, 2017. Disponível em: <https://colaborae.com.br/blog/2017/01/30/modelo-nist/>. Acesso em: 04 de abril de 2024.

JIRA. **Jira Software**. 2024. Disponível em: <https://www.atlassian.com/br/software/jira>. Acessado em 25 de maio de 2024.

JÚNIOR, C. F. C. CARVALHO, K. R. S. A. **Chatbot: uma visão geral sobre aplicações inteligentes.** Revista Sítio Novo, 2018. Disponível em: <https://sitionovo.ifto.edu.br/index.php/sitionovo/article/view/140/86>. Acesso em 20 de maio de 2024.

KAVYASHREE, N. SUPRIYA, M. C. LOKESH M. R. **Site reliability engineering for IOS mobile application in small-medium scale industries**, 2019. Global Transitions Proceedings. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666285X21000935?ref=pdf_download&fr=RR-2&rr=8884ca2febac00ec> Acesso em 20 de maio de 2024.

**Kubernetes.** Disponível em: <https://kubernetes.io/pt-br/>. Acesso em: 13 de abril 2024.

KUBERNETES.IO. **Kubernetes Components**. Kubernetes.io, 2024 Disponível em: <https://kubernetes.io/docs/concepts/overview/components/>. Acesso em: 13 de abril 2024.

LEANDRO, D.R. **O universo multimídia e a Psicologia: um diálogo entre a tecnologia e o emocional humano.** Trabalho de Conclusão de Curso. Criciúma: Curso de graduação em Psicologia, Universidade do Extremo Sul Catarinense, 2007.

LECHETA, R. R. **AWS para Desenvolvedores: Aprenda a instalar aplicações na nuvem da Amazon AWS**. Novatec Editora, 2014.

LIMA, A. C. JUCÁ, S. C. S. LEMOS, P. B. S. SILVA, S. A. **APLICAÇÃO DOS CONCEITOS SDN, DEVOPS E INFRAESTRUTURA COMO CÓDIGO NO PROCESSO DE ENSINO E APRENDIZAGEM DE CIÊNCIAS E ENGENHARIAS.** EnciBio, 2021. Disponível em: <https://www.conhecer.org.br/enciclop/2021C/aplicacao.pdf>. Acesso em: 06 de abril 2024.

MACHADO, G. **O que é AWS S3?** treinaweb, 2020. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-aws-s3>. Acesso em: 06 de abril 2024.

MANZATO, A.; ADRIANA, P.; SANTOS, B. **A ELABORAÇÃO DE QUESTIONÁRIOS NA PESQUISA QUANTITATIVA.** INF - UFSC, 2012. Disponível em: <https://www.inf.ufsc.br/~vera.carmo/Ensino_2012_1/ELABORACAO_QUESTIONARIOS_PESQUISA_QUANTITATIVA.pdf>. Acesso em: 05 de abril. 2024.

MEIRELLES, F. S. VIEIRA, C. S. **Computação em Nuvem: Análise bibliométrica da produção científica sobre os fatores que influenciam as empresas no seu uso.** Revista Eletrónica Gestão e Serviços, 2015. Disponível em: <https://www.researchgate.net/publication/293192132_Computacao_em_Nuvem_Analise_Bibliometrica_da_Producao_Cientifica_Sobre_os_Fatores_que_Influenciam_as_Empresas_no_Seu_Uso>. Acesso em: 4 de abril de 2024.

MELO, D. **O que é Python?** [Guia para iniciantes]. 2023. Disponível em: <https://tecnoblog.net/responde/o-que-e-python-guia-para-iniciantes/>. Acesso em: 10 de fevereiro de 2024.

MUNIZ, A.; OLIVEIRA, T.; MULLER, M. **Jornada SRE no Brasil: Unindo conceitos e práticas da engenharia de confiabilidade para melhorar a experiência do cliente.** Editora Brasport, 2023.

NETTO, H. LUNG, L. C. CORREIA, M. LUIZ, A. F. **Replicação de Máquinas de Estado em containers no Kubernetes: uma Proposta de Integração**. SBRC, 2016. Disponível em: <https://sbrc2016.ufba.br/downloads/SessoesTecnicas/152295.pdf>. Acesso em: 23 de Março de 2024.

NETTO, H. V. LUNG, L. C. OLIVEIRA, C. P. LUIZ, A. F. RECH, L. O. JÚNIOR, J. R. B. NETTO, H. V. et al. **Coordenação de Containers no Kubernetes: Uma Abordagem Baseada em Serviço.** SBRC, 2017. Disponível em: <https://sol.sbc.org.br/index.php/sbrc/article/view/2634/2596>. Acesso em: 23 de Março de 2024.

OLAWANLE, J. **Como Criar um Slackbot com Node.js e a API da Kinsta para Gerenciamento de Sites.** 2023. Disponível em: <https://kinsta.com/pt/blog/criar-Slackbot-gerenciamento-sites/>. Acesso em: 11 abr. 2024.

OLIVEIRA, C. M. **UTILIZAÇÃO DE CHAT BOTS BASEADOS EM LLMS PARA AUTOMAÇÃO DE TESTES DE SOFTWARE**. 2024, UFOP. Disponível em: <https://monografias.ufop.br/bitstream/35400000/6440/3/MONOGRAFIA_Utiliza%c3%a7%c3%a3oChatsBots.pdf> Acessado em 20 de maio de 2024.

RANCHER. **Why Rancher.** Rancher, 2023. Disponível em: <https://www.rancher.com/why-rancher>. Acesso em: 14 de abril 2024.

ROCHA, A. **Site Reliability Engineering | Saiba mais sobre a metodologia de SRE**. opservices, 2021. Disponível em: <https://www.opservices.com.br/site-reliability-engineering/#:~:text=Os%20times%20SRE%20do%20Google>. Acesso em: 23 Março. 2024.

RODRIGUES, T. D. DE F. F.; OLIVEIRA, G. S. DE; SANTOS, J. A. DOS. **AS PESQUISAS QUALITATIVAS E QUANTITATIVAS NA EDUCAÇÃO**. Revista Prisma, v. 2, n. 1, p. 154–174, 25 dez. 2021. Disponível em: <https://revistaprisma.emnuvens.com.br/prisma/article/view/49/41>. Acesso em: 23 de janeiro de 2024.

SÁTYRO, N. G. D.; D'ALBUQUERQUE, R. W. **O que é um Estudo de Caso e quais as suas potencialidades**. Sociedade e Cultura, v. 23, 18 maio 2020. Disponível em: <https://revistas.ufg.br/fcs/article/view/55631/34815>. Acesso em: 24 de janeiro de 2024.

SILVA, B. E. L. S**ERVIÇOS DE SEGURANÇA EM COMPUTAÇÃO EM NUVEM USANDO A PLATAFORMA AMAZON WEB SERVICE (AWS)**. PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS. 2023. Disponível em: <https://repositorio.pucgoias.edu.br/jspui/bitstream/123456789/5862/1/TCC2_Bruno%20Emilio%20Luiz%20Silva-rag.pdf>. Acesso em: 10 de fevereiro de 2024.

SILVA, G. A. **COMO GARANTIR QUE UM CLUSTER KUBERNETES POSSUI COBERTURA DE FALHAS CONTINUAMENTE NA CLOUD? O USODE CHAOS ENGINEERING NA ESTEIRA DE ENTREGA CONTÍNUA.** UFPE, 2022. Disponível em: <https://repositorio.ufpe.br/handle/123456789/47224> Acesso em: 15 de abril 2024.

SILVA, W. F. **UTILIZANDO VIRTUALIZAÇÃO BASEADA EM CONTAINERS PARA CRIAÇÃO DE LABORATÓRIOS PRÁTICOS DE DISCIPLINAS NA ÁREA DE TI**. UFC, 2017. Disponível em: <https://repositorio.ufc.br/bitstream/riufc/29549/1/2017_tcc_wfsilva.pdf>. Acesso em: 14 de abril 2024.

Slack. **Desenvolva apps com as ferramentas de desenvolvedor do Slack**. Disponível em: <https://Slack.com/intl/pt-br/help/articles/13345326945043-Desenvolva-apps-com-as-ferramentas-de-desenvolvedor-do-Slack>. Acesso em: 24 fevereiro de 2024.

TEBALDI, P. C. **Computação em nuvem | Quais as principais vantagens e riscos?** Opservice, 2018. Disponível em: <https://www.opservices.com.br/riscos-da-adocao-da-computacao-em-nuvem/>. Acesso em: 06 de abril 2024.

SATO, D. **DevOps na prática: Entrega de software confiável e automatizada**. Editora Casa do Código, 2014.

Slack. Slack, 2024. Página inicial. Disponível em: <https://Slack.com/intl/pt-br>. Acesso em: 29 de abril de 2024.

TECNOMEGA, B. **Conheça Os 3 Modelos Cloud: IaaS, PaaS E SaaS**. Tecnomega, 2021. Disponível em: <https://tecnomega.com.br/blog/conheca-os-modelos-iaas-paas-saas/>. Acesso em: 13 de abril 2024.

VERAS, M. **Cloud Computing: nova arquitetura da TI**. Brasport, 2012.