Índice

1. INTRODUÇÃO

Este documento define diretrizes para o desenvolvimento seguro de software adotado pela empresa Kaffa que visa garantir que os produtos e aplicações desenvolvidas sejam de fato seguras, de ponta a ponta.

2. OBJETIVO

O objetivo é definir diretrizes de boas práticas a serem adotadas, tornando o processo de concepção de sistemas mais confiável, estável, e protegido contra ameaças, prezando sempre pela confidencialidade, integridade e disponibilidade das informações.

Este documento atende aos controles A.8.25 (Ciclo de vida de desenvolvimento seguro) e A.8.26 (Requisitos de segurança de aplicações) da norma NBR ISO/IEC 27001:2022, assegurando que práticas de segurança da informação estejam integradas em todas as fases do desenvolvimento de software.

3. APLICAÇÃO E ÁREAS ENVOLVIDAS

A aplicação desta política abrange o Desenvolvimento de Sistemas das Diretorias de Produto & Tecnologia, Implantação e Sucesso do Cliente Kaffa. Esta política também se aplica ao uso de serviços internos de provisionamento e manutenção utilizados no processo de desenvolvimento, incluindo repositórios de código, repositórios de artefatos e dependências, gerenciamento de vulnerabilidades, ferramentas de integração contínua e demais serviços de apoio ao ciclo de vida do software, todos gerenciados internamente pela organização.

4. DEFINIÇÕES E PREMISSAS

***SQL Injection*:** É uma forma de ataque em sistemas, realizado via *interface*, no qual o usuário informa trechos de *SQL* em campos de texto (ou até mesmo em telas de *login* ou de pesquisa), alterando a consulta prevista pelo desenvolvedor,

sendo que o atacante poderá receber privilégios especiais ou poderá manipular indevidamente o banco de dados.

Encriptação: O processo de converter dados em um formato cifrado, usando um algoritmo e uma chave para proteger a informação.

Decriptação: O processo de converter dados cifrados (ou encriptados) em seu formato original.

Cifrador: algoritmo ou ferramenta que realiza a encriptação e a decriptação.

Chave: Sequência de *bits* utilizada como um "código secreto" pelo cifrador, para realizar encriptação e/ou decriptação de dados. A única maneira de descobrir uma chave deve ser por força-bruta; tentando todas as alternativas no espaço de chaves possíveis. O algoritmo utilizado pelo cifrador deve garantir que chaves longas e complexas tornem a descoberta por tentativa e erro impraticável devido ao grande número de combinações.

Cifrador Simétrico: É um cifrador que usa a mesma chave para encriptação e decriptação. Mais rápidos, em geral. Exemplos: *Advanced Encryptation Standard* (AES)18 e *Data Encryptation Standard* (DES)19.

Cifrador Assimétrico: É um cifrador que usa chaves diferentes, uma pública e uma privada para encriptação e decriptação. Mais lentos, mas com usos para assinatura e verificação de autenticidade. Exemplos: *Rivest-Shamir-Adleman* (RSA)20 e *Elliptic Curve Cryptography* (ECC)21.

***Hash* Criptográfico:** Função matemática que transforma dados de qualquer tamanho em uma sequência de tamanho fixo. Essa sequência é única para cada entrada, ou seja, se você passar os mesmos dados pela função, sempre obterá o mesmo resultado. A função é unidirecional, ou seja, impossibilita a reversão

para se obter os dados originais. Além disto é rápida, segura e garante o mesmo resultado para a mesma entrada.

Cifrador de bloco: Cifrador que opera sobre blocos de *bits* de tamanho fixo, encriptando um bloco por vez.

***WS-Reliable Messaging*:** Protocolo para entrega segura de mensagens *SOAP*.

***SOAP*:** Protocolo para troca de mensagens em formato *XML*.

***REST*:** Protocolo para comunicação entre sistemas utilizando os métodos do protocolo *HTTP*.

***V*PN** (Virtual Private Network): Rede privada virtual que estabelece um canal de comunicação seguro e criptografado sobre uma rede pública, permitindo o acesso remoto a recursos internos da organização de forma protegida.

SSH (Secure Shell): Protocolo de rede criptográfico utilizado para autenticação e comunicação segura entre sistemas, permitindo acesso remoto e transferência de dados de forma protegida por meio de pares de chaves criptográficas.

CI/CD (Continuous Integration / Continuous Delivery): Práticas de engenharia de software que automatizam a integração, validação e entrega de código. A integração contínua realiza verificações automatizadas a cada alteração submetida ao repositório, enquanto a entrega contínua automatiza o processo de disponibilização do software nos ambientes.

Pipeline: Conjunto sequencial e automatizado de etapas executadas sobre o código-fonte, que pode incluir compilação, execução de testes, análise de qualidade, validação de convenções e publicação de artefatos.

5. RESPONSABILIDADES

Atividade	Responsável
Responsável por maximizar o valor do produto ao representar as necessidades do negócio e dos stakeholders. Atua na definição, priorização e gestão do backlog, incluindo a elaboração e o refinamento das histórias de usuário (User Stories) com critérios de aceite claros. Trabalha de forma colaborativa com o time de desenvolvimento durante o refinamento e planejamento, contribuindo para o entendimento das demandas, mas sem atribuir estimativas, que são de responsabilidade do time. Participa da validação das entregas, garantindo aderência aos requisitos definidos e ao valor esperado. Em projetos que contem com o papel de analista de negócios, atua de forma complementar, alinhando necessidades e garantindo a tradução adequada dos requisitos para o backlog do produto.	*Product Owner (PO)*
Responsável por analisar, revisar e aprovar políticas relacionadas ao desenvolvimento de sistemas, validando documentações para publicação e propondo melhorias à alta direção. Atua para garantir o alinhamento estratégico, a alocação de recursos e a evolução contínua dos processos, promovendo eficiência, qualidade e inovação nas entregas da área.	Product Delivery Manager
Responsável por transformar histórias em soluções funcionais com qualidade. Todos colaboram para entregar valor ao negócio de forma contínua,	Equipe de Desenvolvimento, Líder Técnico e Manutenção

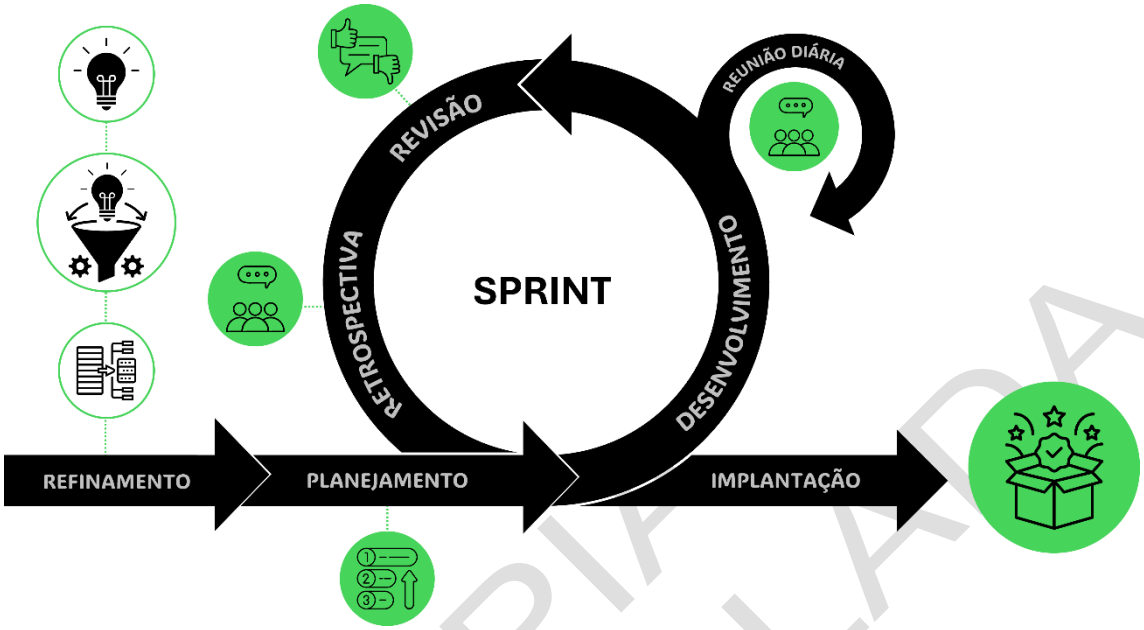
desenvolvendo, testando, corrigindo e implementando as soluções para os Produtos Kaffa, seguindo os preceitos desta política e boas práticas de desenvolvimento seguro. Armazenar códigos-fonte dos sistemas desenvolvidos de maneira segura e realizar o gerenciamento das bases de dados. O Líder Técnico atua como referência técnica do time, sendo responsável por orientar decisões de arquitetura e desenvolvimento, garantir a aplicação de boas práticas de desenvolvimento e segurança da informação, apoiar a resolução de problemas complexos e promover a evolução técnica da equipe. Também assegura o alinhamento das soluções com os padrões definidos, contribuindo para a qualidade, segurança e sustentabilidade dos sistemas.	
Responsável por planejar, executar e monitorar atividades relacionadas à garantia e ao controle da qualidade, assegurando a conformidade de produtos, serviços e processos com os padrões estabelecidos. Atua na identificação e análise de não conformidades, na definição e acompanhamento de ações corretivas e preventivas, bem como na validação de requisitos e critérios de aceitação. Promove a melhoria contínua por meio da revisão de processos, indicadores e práticas de qualidade, contribuindo para a eficiência operacional, a mitigação de riscos e a satisfação dos clientes e das partes interessadas.	Analista de Qualidade

Responsável por planejar, coordenar e supervisionar a execução de projetos, assegurando o cumprimento de prazos, custos e qualidade estabelecidos. Atua no gerenciamento de riscos, na alocação eficiente de recursos e no alinhamento entre as partes interessadas, garantindo a entrega de resultados conforme os objetivos estratégicos. Promove a melhoria contínua das práticas de gestão, contribuindo para a eficiência operacional e o sucesso dos projetos da organização.	Gerente de Projeto
Responsável por identificar, analisar e documentar as necessidades de negócio, traduzindo requisitos em soluções que agreguem valor à organização. Atua na interface entre as áreas de negócio e tecnologia, garantindo o entendimento claro das demandas e a aderência às expectativas dos stakeholders. Contribui para a melhoria contínua de processos, apoiando a tomada de decisão e assegurando que as soluções implementadas estejam alinhadas aos objetivos estratégicos da empresa.	Analista de Negócio
Responsável por garantir recursos e suporte estratégico para iniciativas de melhoria contínua.	Alta Direção

6. DESCRIÇÃO DA POLÍTICA

6.1. METODOLOGIA APLICADA

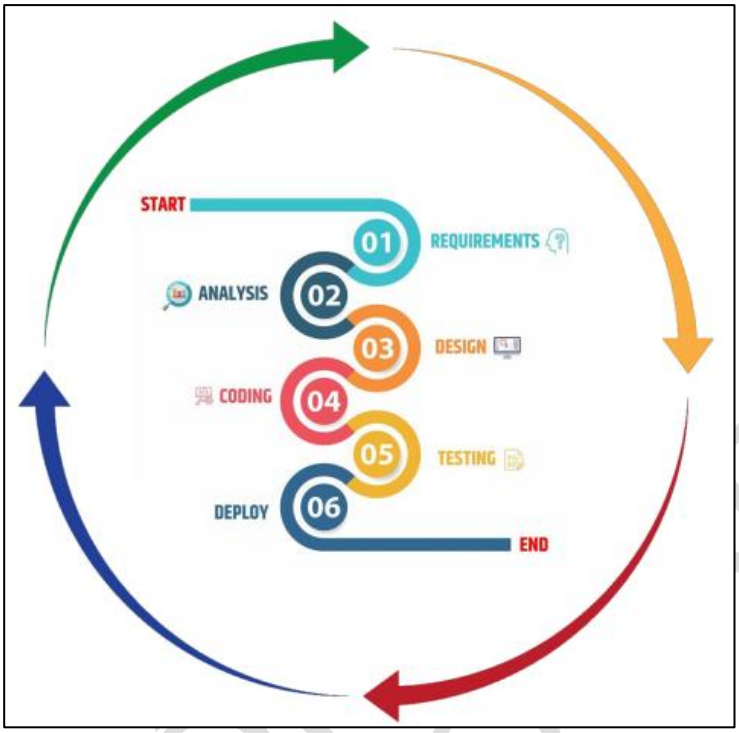
A Kaffa desenvolve seus produtos e *API's* utilizando a metodologia ágil com práticas do *framework Scrum* conforme exemplificado na figura abaixo:



Scrum é um *framework* leve que ajuda pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos. O *Scrum* é baseado no empirismo e *lean thinking*. O empirismo afirma que o conhecimento vem da experiência e da tomada de decisões com base no que é observado. O *lean thinking* reduz o desperdício e se concentra no essencial.

É um *framework* iterativo e incremental, que possui 3 pilares centrais: transparência, inspeção e adaptação.

A Kaffa utiliza o *SCRUM* adaptado para a realidade e cultura da empresa, e ao ciclo de vida do desenvolvimento de software (*SDLC – Software Development Life Cycle*):



A gestão da metodologia, incluindo planejamento de sprints, acompanhamento de tarefas, registro de histórias e controle de backlog, é realizada por meio da ferramenta Azure DevOps.

O ciclo padrão de sprint é de duas semanas. Em situações emergenciais, como correções críticas ou demandas de alta prioridade, podem ser executados planejamentos emergenciais com ciclos reduzidos de até uma semana, garantindo agilidade na resposta sem comprometer o controle e a rastreabilidade do processo, em contrapartida em situações com feriados durante a semana ou situações de ausência de parte do time a sprint pode ser adaptada para 3 semanas.

Ao final de cada sprint é realizada a Retrospectiva, cerimônia na qual o time avalia o processo de trabalho, identifica pontos de melhoria e define ações para aprimorar a eficiência e qualidade das entregas nos próximos ciclos.

6.2. GESTÃO DE MUDANÇAS

Para garantia da estabilidade, segurança e continuidade dos serviços e aplicações Kaffa, adotamos a prática de gestão de mudanças. Com ela conseguimos atuar na evolução e correção das nossas soluções de forma controlada, transparente e previsível.

As mudanças realizadas no sistema são identificadas, classificadas, avaliadas quanto ao impacto, priorizadas, gerenciadas e aplicadas seguindo as etapas do ciclo de desenvolvimento seguro e de engenharia de software.

Os *Product Owners (POs) e Analistas de negócio (ANs)* em conjunto com os clientes/*stakeholders*, são responsáveis por definir as mudanças que serão executadas no sistema através de demandas internas, regulatórias, legais ou de mercado. Após a decisão de incorporação de uma alteração no sistema são realizadas diversas atividades que contemplam a especificação, codificação, teste e validação da mudança proposta.

A alteração de código segue um processo formal de aprovação. O desenvolvedor submete um Merge Request (MR) no GitLab ou no Azure DevOps, a depender do setor, que é obrigatoriamente revisado por outros membros da equipe por meio de Code Review.

Em paralelo à revisão, é realizado um scan de vulnerabilidades no código fonte utilizando três ferramentas open-source, cada uma responsável por uma camada de análise:

- **GitLeaks** — detecção de secrets. Verifica o código em busca de credenciais expostas (chaves de API, tokens, senhas, strings de conexão) que possam ter sido commitadas indevidamente.

- **Semgrep** — análise estática de código (SAST). Examina o código fonte em busca de padrões inseguros e vulnerabilidades de implementação, como injeção, validação inadequada de entrada e uso incorreto de APIs sensíveis.

- **Trivy** — análise de composição de software (SCA). Inspeciona as dependências do projeto em busca de vulnerabilidades conhecidas (CVEs) nas bibliotecas e pacotes de terceiros utilizados.

As vulnerabilidades identificadas pelas três ferramentas são centralizadas e armazenadas no DefectDojo, uma plataforma open source para gestão de vulnerabilidades e testes de segurança, que consolida os resultados e permite o acompanhamento e o tratamento das ocorrências de forma rastreável.

Concluído o scan, chega-se à fase de aprovação (approve), obrigatória para que a mesclagem seja efetivada. Todo o ciclo de vida do MR (desde a criação até a incorporação) é notificado por e-mail aos envolvidos, garantindo transparência e rastreabilidade em cada etapa do processo.

Após esse processo, é gerada uma release final, que será homologada e, se aprovada, considerada para envio à produção.

Além da aprovação via Code Review, a incorporação de alterações ao repositório principal é restrita a colaboradores com perfil de Maintainer (mantenedor).

Como camada adicional de controle, o acesso aos repositórios de código exige o uso de VPN para conexão aos serviços internos e cadastro prévio de chave SSH para autenticação. Cada desenvolvedor possui acesso restrito apenas aos repositórios dos projetos para os quais foi previamente provisionado com perfil de desenvolvedor, garantindo que o controle de mudanças se inicie desde o acesso ao código-fonte.

Com todas as etapas concluídas e validadas, a funcionalidade ou evolução desenvolvida pode ser implantada em ambiente de produção, mantendo o registro de versionamento, *changelog* e diretriz de *rollback*.

Cada uma das etapas está descrita na próxima seção.

6.3. FASES DO CICLO DE DESENVOLVIMENTO

Os processos de desenvolvimento estão descritos nos fluxogramas P-DP-01 - Desenvolvimento do Produto e Implementação de Melhorias EGW,P-DP-02-Comitê de Demandas de Melhorias EGW, P-DP-03-Correção de Bugs EGW, P-DP-02 - Desenvolvimento do Produto PGR e P-DI-01- Desenvolvimento – Implantação e estão detalhados nas seguintes etapas:

6.3.1. Entendendo o Projeto e Coletando Informações

Nesta fase ocorre a análise e entendimento dos principais objetivos e necessidades do projeto pelo Analista de Negócios, a partir de demanda encaminhada pelo Gerente de Projetos (GP). Então, busca-se reunir/organizar:

- Informações que podem ser encaminhadas a partir de um pedido de um cliente/*stakeholder, feedback* de usuários, mudanças regulatórias ou legais, análise de mercado ou concorrência e/ou incidentes;
- Se há restrições tecnológicas e regulatórias;
- Se há prazos e expectativas de data.

6.3.2. Entendendo o Escopo e Definindo Grandes Funcionalidades

Na Análise de Alto Nível e a Identificação de features o PO deve garantir que o escopo do trabalho seja bem definido e compreendido. As features são identificadas para representar funcionalidades, com base nas informações recebidas dos clientes ou stakeholders. Em seguida, o PO discute o tema com o

time técnico levantando riscos e dependências, e promovendo uma visão clara do impacto técnico.

6.3.3. Estimando Esforços e Alinhando Expectativas

Em formato síncrono (preferencialmente) ou assíncrono, o time técnico e o PO atuam de forma colaborativa para gerar uma visão inicial do esforço necessário para cada feature. Pode-se utilizar técnicas de macro estimativa, como *T-shirt sizing* (P, M, G, GG), *story points* aplicados as features, comparações com projetos anteriores, planning poker, horas ou até mesmo *PERT*. O objetivo é garantir que todos os participantes alinhem suas percepções e cheguem a um consenso. As estimativas resultantes devem ser registradas durante o rito de planning (planejamento).

Importante: para a estimativa, são consideradas as atividades de geração e refinamento de *backlog*, desenvolvimento, testes (unitários, integração e funcional) e implantação.

6.3.4. Identificando Riscos e Preparando Soluções

O *PO* e o time técnico, identificam possíveis riscos que possam impactar o projeto, incluindo: riscos técnicos (Exemplo: uso de tecnologias novas ou integrações complexas), e riscos de negócio (Exemplo: mudanças regulatórias ou requisitos incertos). São discutidas as estratégias para mitigar esses riscos; e caso necessário, o *PO* e o time podem adicionar *buffers* (margens de segurança) às estimativas, garantindo maior previsibilidade na entrega.

6.3.5. Criando e Validando Protótipos

Em projetos que demandam validação visual ou de usabilidade, antes do desenvolvimento, o *PO* e a equipe de design, quando necessária, elaboram protótipos de alta ou baixa fidelidade (a combinar entre eles). Esses protótipos são compartilhados com o analista de negócios que é quem realiza o registro da

necessidade. Com base nas sugestões recebidas, os protótipos são revisados e ajustados. Após as revisões, o PO solicita a aprovação formal dos protótipos ao *analista de negócios,* garantindo o alinhamento necessário antes do início do desenvolvimento.

6.3.6. Aprovando o Projeto

O *analista de negócios* consolida a Lista de Entregas, Macro Estimativa e a Avaliação de Riscos e Incertezas à demanda tanto no âmbito de desenvolvimento da parte de produto quanto da parte de projetos, para comunicar o GP e, em seguida busca a aprovação do cliente/*stakeholde*r para início do projeto. Se novos detalhes surgirem entre a interação com o GP, ou cliente/stakeholder, a estimativa é revisada e refinada, com todas as alterações devidamente registradas para manter o histórico de decisões.

6.3.7. Gerando o backlog

Com a aprovação da macro estimativa e do protótipo, a equipe de projeto inicia a organização do backlog de projeto, estruturando as entregas necessárias para viabilizar a implementação junto ao cliente.

Paralelamente, a equipe de produto é responsável pela criação e gestão do backlog de produto, onde o PO decompõe as necessidades em histórias de usuário, podendo incluir também histórias técnicas ou facilitadoras com apoio do time técnico, abrangendo arquitetura, infraestrutura, automação, pipelines DevOps, APIs, integrações, segurança e escalabilidade.

Os itens de cada backlog são priorizados de forma independente, considerando valor de negócio, complexidade e dependências. À medida que a equipe de produto conclui suas entregas, estas são disponibilizadas para a equipe de projeto, que as utiliza na execução e entrega final ao cliente, garantindo um fluxo coordenado entre as frentes.

IMPORTANTE: Todo o material gerado sobre e para gestão das demandas se dividem entre os *softwares*:

- DefectDojo: Plataforma open source para gestão de vulnerabilidades e testes de segurança.
- Jira: software para gestão estruturada de tarefas, manutenções e projetos de desenvolvimento de *software*.
- SharePoint: é uma plataforma colaborativa baseada na web que permite a gestão de documentos, compartilhamento de informações e criação de intranets corporativas, facilitando a comunicação, organização e colaboração entre equipes dentro das organizações.
- Teams: é uma plataforma de comunicação e colaboração que reúne chats, reuniões, chamadas, compartilhamento de arquivos e integração com ferramentas corporativas em um só lugar, facilitando o trabalho em equipe e a troca de informações de forma segura.
- Azure Devops Services: é uma plataforma SaaS que reúne ferramentas integradas para todo o ciclo de vida do desenvolvimento de software.

  Azure Boards — Planejamento e rastreamento de trabalho (Kanban, Scrum, backlog, sprints), work itens:

  Epic — Grandes iniciativas de negócio

  Feature — Funcionalidades dentro de um Epic

  User Story / PBI — Requisitos do usuário

  Task — Tarefas técnicas vinculadas a uma Story

  Bug — Defeitos encontrados no software

6.3.8. Refinamento Contínuo

O *PO*, em conjunto com o time de desenvolvimento, estabelece uma rotina de refinamentos contínuos do *backlog,* para garantir que as histórias estejam bem definidas e claras para todos. O objetivo é listar em tarefas menores e estimar (pode ser no refinamento ou na planning) os itens do *backlog* para que tudo esteja aderente ao *DoR* (*Definition of Ready* = Definição de Pronto), para serem selecionados nas próximas s*prints*, promovendo um fluxo de trabalho eficiente e alinhado às prioridades do projeto.

6.3.9. Organizando o Trabalho e Definindo Próximos Passos

O ciclo da *sprint* começa com uma reunião pré planning onde é envolvido o *PO* e lideranças realizando a quebras das Features, User Stories e validando se há impedimentos, a partir do resultado dessa agenda inicia-se o rito de *Planning*, uma reunião que envolve o time de desenvolvimento, *PO*, Liderança Técnica e Product Delivery Manager. Durante essa reunião, este grupo define o objetivo da s*print*;

Seleciona, e organiza os itens do *backlog* do produto que serão trabalhados formando o *backlog* da *Sprint*. O time garante que o escopo definido é realista e viável de ser concluído, dentro do tempo previsto para a *sprint*.

6.3.10. Construindo e Testando a Solução

O time de desenvolvimento executa as atividades necessárias para concluir os itens do *Sprint Backlog*, garantindo a entrega das funcionalidades com qualidade. Entre essas atividades estão:

- **Codificação**: Desenvolvimento e testes do código para implementar as funcionalidades planejadas. Ao final da codificação de uma história, se gera o *Merge Request* (*MR*). O *Merge Request* (*MR*) é o processo de solicitar a mesclagem de uma *branch* (versão do código para trabalho em novas funcionalidades ou correções) ao código principal.

- **Scan de Vulnerabilidades (código fonte)**: Processo automatizado de cibersegurança que identifica, detecta e avalia falhas de segurança da informação que visa centralizar achados de segurança vindos de várias ferramentas, como scanners de SAST, DAST, SCA e pentests.
- **Testes**: Execução de testes unitários, de integração e de aceitação para garantir que as funcionalidades atendam aos critérios das histórias de usuário. São utilizados dados fictícios nos testes para garantir segurança, flexibilidade e controle; além de testar diversos cenários, sem comprometer dados reais dos clientes.
- **Refatoração**: Aperfeiçoamento contínuo do código existente, visando melhorar sua qualidade, manutenção e escalabilidade. Será executado em algumas situações, se necessário.
- **Integração Contínua**: Integração regular do código ao repositório central, com execução automática de testes para validar a entrega. Será executado em algumas situações, se necessário.
- **Documentação**: Criação ou atualização de documentação mínima e suficiente para o entendimento, desenvolvimento e manutenção das funcionalidades. Esta documentação fica disponível para consulta do time, ao longo e pós-implantação do projeto, no *Wiki do Devops e no Gitlab*.
- **Revisão de Código (*Code Review*)**: Avaliação do código realizada por outros membros da equipe, para garantir conformidade com as boas práticas de desenvolvimento, identificar possíveis erros e promover um código limpo e bem estruturado.

Importante: antes de informar a história como concluída, cada membro do time que atuou na história deve verificar se ela está atendendo a todos os itens que fazem parte do *DoD* (*Definition of Done* = Definição de Concluído).

Ao longo da execução da *sprint*, o time se reúne diariamente para sincronizar as atividades do dia e se apoiar na remoção de impedimentos. As atividades contidas nas histórias são atualizadas quanto a status, comentários e tempo gasto para atualização, documentação e acompanhamento da velocidade de entrega do time.

6.3.11. Validação do Analista de Qualidade e Homologação do Cliente

A validação do QA pode acontecer em 2 momentos:

- Ao longo da sprint, à medida que o time efetua as entregas de cada **história;**
- No evento de final de sprint, a *Review* (Revisão).

A validação do *QA* consiste na inspeção das entregas, se está aderente ao que consta na História que fazia parte do backlog da Sprint.

Havendo ocorrências de:

- Erros: o próprio *QA* registra como bug dentro da respectiva história, para o time atuar na correção, na própria *sprint* se houver tempo hábil ou em outra que será planejada no futuro.
- Alterações de escopo: eles também são registrados no Azure Devops, como novas histórias que serão planejadas futuramente em outra sprint.

Não havendo a ocorrência de erros, o *QA* segue com a homologação junto ao cliente e outras partes interessadas:

- Por envio de link do ambiente de homologação, para avaliação de forma assíncrona ou síncrona (assistida) do cliente/*stakeholder*.

Durante a *Review* realizada momentos antes da retrospectiva, o time vai realizar a demonstração do que foi produzido e o cliente/*stakeholder* pode ser convidado a participar, a convite do *PO*. Na *Review* o time também discute o *backlog* do produto, o *PO* pode revisá-lo e pré organizar como será a reunião de planejamento da próxima sprint.

6.3.12. Implantação

Assim que o cliente/stakeholder aprovar a implantação, o *PO* comunica o time técnico que inicia as atividades de geração dos artefatos de entrega, que são: a criação do release notes (*changelog*, *tag* de versionamento e release final) junto ao *backup* do banco de dados.

A cada publicação em produção é realizado um versionamento do código com o seguinte padrão de documentação: Cada versão é composta por 3 dígitos no padrão X.Y.Z (ex.: 1.0.0), onde:

- X é incrementado quando há uma grande reestruturação do sistema
- Y é incrementado a cada nova funcionalidade nos módulos já existentes
- Z é incrementado a cada publicação com correção de erro ou melhoria em funcionalidades existentes.

Os registros do processo são realizados nas ferramentas JIRA, *GitLab e Azure Devops* que possuem acessos controlados para a equipe de desenvolvimento.

Caso ocorra alguma falha na versão publicada, é realizado um processo de *rollback*, que consiste na restauração do sistema a partir da última versão estável (tag do código fonte e *backup* do banco de dados).

Os ambientes de homologação, produção e o banco de dados da Kaffa são controlados pela área de Tecnologia da Informação da Atmis Gestão e possuem os processos e controles de segurança da informação implementados para evitar incidentes no ambiente, conforme M-TI-01 – Manual de Segurança da Informação. Além disso existe a gestão de ambientes de homologação, produção e banco de dados controlados por clientes da Kaffa.

Estão em desenvolvimento os ambientes de homologação e produção para futura disponibilização do produto EGW na nuvem. Atualmente, esses ambientes estão sob administração direta da Kaffa.

6.3.13. Pós-Implantação

Após a geração de release (ambiente de desenvolvimento e homologação), o time realiza *smoke test, tratando-se* de ambiente de produção é realizado esse tipo de teste após a aprovação da release de homologação disponibilizada e testada para validar se o sistema está funcionando corretamente. Em seguida, o *changelog* é atualizado, documentando todas as alterações implementadas. Durante o monitoramento pós-*deploy*, a equipe acompanha métricas e *logs* para identificar falhas ou possíveis degradações de performance.

Por fim, o *Analista de Negócios* comunica a conclusão da entrega aos clientes/*stakeholders* e coleta *feedback* sobre a nova versão.

6.4. GESTÃO DE INCIDENTES E SUSTENTAÇÃO

O processo de atendimento ao cliente em casos de identificação de ocorrências é realizado por meio de abertura de chamado nas plataformas de gestão de serviços fornecidas pelos próprios clientes. As ferramentas utilizadas incluem soluções como JIRA, ServiceNow e Freshworks, conforme a plataforma adotada por cada cliente contratante. O solicitante deve relatar o problema e, quando aplicável, anexar evidências (prints, relatórios ou vídeos). O fluxo de atendimento segue os princípios do framework ITIL (Information Technology Infrastructure Library), garantindo rastreabilidade, controle e qualidade no tratamento de cada ocorrência, conforme descritos nos fluxogramas P-DSC-02 - Atendimento AMS EGW KAFFA, P-DSC-03 - Atendimento AMS GSE e P-DSC-04 - Atendimento AMS PGR

O fluxo de atendimento é estruturado em níveis de suporte, conforme descrito a seguir:

N1 – Triagem e Primeiro Atendimento: O chamado é recebido pelo N1, responsável por realizar a triagem da ocorrência. Nessa etapa, o atendente classifica o chamado, coleta as informações necessárias, define a prioridade com base no impacto ao negócio e encaminha para o nível adequado de atendimento.

N2 – Suporte Kaffa: O chamado triado é direcionado ao time de suporte da Kaffa (N2), responsável pela análise técnica e resolução das ocorrências dentro do escopo de sustentação. O time investiga o problema analisando logs, dados e evidências disponíveis. Quando necessário, envolve outros papéis como Liderança Técnica (TL), Desenvolvedores e Analista de Negócio, dependendo da complexidade. Após o diagnóstico, é aplicada uma solução paliativa para reduzir o impacto imediato ao cliente

N3 – Escalação para Problemas e Demandas: A partir da análise do N2, o chamado pode originar dois tipos de escalação para o N3.

Problema (PRB): Quando um chamado é identificado como recorrente ou de causa raiz desconhecida, é registrado um Problema (PRB) e encaminhado ao N3 para investigação aprofundada e tratativa estrutural.

Demanda: Quando a ocorrência evidencia a necessidade de evolução ou melhoria do sistema, pode ser aberta uma demanda a partir do chamado, que também é encaminhada ao N3 para análise, priorização e desenvolvimento conforme o ciclo de desenvolvimento seguro descrito nesta política.

6.4.1. Análise e Solução

O time de sustentação investiga o problema analisando logs e dados. Se necessário, envolve outros papéis como *TL* e *PO* dependendo da complexidade.

Após o diagnóstico, define se a solução será paliativa (para reduzir o impacto imediato) ou definitiva (para evitar que o problema volte a acontecer).

6.4.2. Solucionando e Implantando

Com a análise concluída, o time de sustentação desenvolve a correção com base nas práticas de desenvolvimento dessa política e realiza os testes. Após os testes, implanta a solução em produção, válida se o problema foi resolvido e registra no chamado o que foi feito. Em seguida, encerra o chamado.

6.5. DIRETRIZES APLICADAS PARA O PROCESSO DE DESENVOLVIMENTO SEGURO

A Kaffa adota diretrizes de segurança da informação, para todo o processo que impacta o desenvolvimento seguro, desde a infraestrutura até o processo de implantação. As diretrizes relacionadas à infraestrutura são de responsabilidade da equipe de TI da Atmis Gestão, enquanto as regras aplicáveis às aplicações e sistemas desenvolvidos são de responsabilidade da Kaffa, conforme M-TI-01 – Manual do SGSI e nos tópicos abaixo.

6.5.1. Armazenamento de senhas

As senhas dos usuários devem ser armazenadas de forma segura, seguindo as diretrizes estabelecidas para proteção dos dados. As bases de dados correspondentes devem garantir a confidencialidade das senhas, e a política de acesso segue as diretrizes definidas no documento PO-TI-05 – Política de Gestão de Acessos.

- **Armazenamento para Dados Fechados**

  Diretriz aplicada: Acesso de leitura/escrita restrito por senha.

- **Permissões de Acesso a Dados em Banco**

  Diretriz aplicada: Cada aplicação tem seu usuário de serviço com permissão de acesso (leitura e escrita) a base. O acesso feito pela equipe de

desenvolvimento às bases de dados são executados utilizando o usuário nomeado.

- **Gerenciamento e Distribuição de Senhas para Acesso a Dados**

  Diretriz aplicada: As senhas mestre são geradas seguindo as variações de tipos de caracteres: letras maiúsculas, letras minúsculas, dígitos e caracteres especiais. As senhas são criptografadas de acordo com o sistema operacional. Os sistemas só conseguem acessar as bases via servidores de produção / homologação. As senhas mestras são armazenadas em um repositório de dados da TI.

6.5.2. Controle de Usuários: Acessos e Permissões

A gestão da identidade dos usuários e dos níveis de acesso segue padrões de segurança que asseguram a integridade e confidencialidade das informações.

Os acessos são concedidos por meio de usuários e senhas nominais, com permissões estabelecidas de acordo com o perfil de atuação de cada colaborador, seguindo as diretrizes estabelecida pela TI da Atmis Gestão e pela Kaffa, conforme PO-TI-05 - Política De Gestão de Acessos.

- **Identidade do Usuário e Nível de Acesso**

  Diretriz aplicada: Usuário e senha nominais, são dadas ciência das permissões e níveis de acesso utilizando de grupos do Active Directory que gerencia a autenticação de usuários na rede interna, uso de VPN, grupos e pastas no sharepoint, e serve de autenticação no Azure Devops.

- **Autenticação de Usuários**

  Diretriz aplicada: Não são armazenadas senhas em texto sem utilização de um algoritmo de hash seguro. Utiliza-se autenticação via AD para autenticar usuários internos.

- **Autenticação em Sistemas Web**

Diretriz aplicada: Todas as autenticações de sistema Web da corporação são utilizadas as autenticações integradas ao sistema AD, seguindo as normas do PR-TI-04 - Gestão de Acessos.

- **Usuários Genéricos**

  Diretriz aplicada: não é permitido a utilização de usuários genéricos em nenhuma etapa do desenvolvimento, testes, homologação ou produção. Todos os usuários são individuais e devidamente identificados, assegurando a rastreabilidade e a responsabilidade pelas ações realizadas no sistema.

- **Comunicação de Desligamento e Revogação de acessos**

Diretriz aplicada: É responsabilidade dos líderes de equipes e gestores de projetos assegurar que, em caso de desligamento ou mudança de função de colaboradores que atuem em atividades de desenvolvimento, manutenção de sistemas ou que possuam acesso a ambientes, repositórios, bases de dados ou informações de clientes, realizem a comunicação formal e imediata aos clientes e demais partes interessadas, a fim de possibilitar a revogação dos acessos sob sua gestão, bem como prevenir comunicações indevidas ou não autorizadas em nome da organização.

6.5.3. Comunicação Segura

São estabelecidas diretrizes para as comunicações entre sistemas e módulos, assegurando a integridade, confidencialidade e autenticidade das informações transmitidas. Todas as interações seguem mecanismos que previnem a duplicação de mensagens e garantem que apenas sistemas autorizados possam trocar dados de forma segura.

A comunicação segura é estabelecida conforme as diretrizes definidas na PO-TI-03 – Política de Criptografia, que determina o uso de protocolos e métodos de criptografia.

6.5.4. Ataques à Sistemas e suas Defesas

A prevenção de ataques é realizada por meio da implementação de medidas de segurança contra DDoS, SQL Injection, HTML Injection, JavaScript Injection, ataques XSS, quebra de autenticação e falhas no gerenciamento de sessão. Essas proteções garantem a integridade dos sistemas e a mitigação de vulnerabilidades exploráveis por agentes mal-intencionados.

6.5.5. Auditoria, Rastreamento e Logs

O rastreamento das operações realizadas pelos usuários nos sistemas é garantido por meio da utilização de ferramentas específicas. O GitLab é adotado para o gerenciamento de códigos, enquanto a ferramenta SIEM é utilizada para monitorar e registrar as atividades do Active Directory e do ambiente de gerenciamento da Tenant.

6.5.6. Prevenção, Reação e Mitigação de Falhas de Segurança

A segurança dos sistemas é mantida por meio de estratégias que englobam prevenção, resposta rápida e mitigação de falhas. São adotadas práticas proativas para identificar e corrigir vulnerabilidades antes que sejam exploradas, além de mecanismos para detecção e resposta imediata a incidentes de segurança.

- **Diretivas de backup**

  Diretriz aplicada: Conforme a política PO-TI-04. Política de Backup e Disaster Recovery.

- **Políticas de testes**

  Diretriz aplicada: São realizados testes manuais e automáticos antes de liberações de versão de software, para reduzir o risco de falhas no ambiente de produção que causem impactos no negócio, conforme descrito no item 6.3.10 dessa política.

- **Ocorrências de falhas de segurança**

  Diretriz aplicada: Ocorrências de falhas de segurança são identificadas proativamente pela gestão de vulnerabilidades técnicas de sistemas ou reativamente através de bugs ou incidentes.

  As ocorrências consideradas incidentes são tratadas pela gestão de incidentes, conforme PR-TI-03 – Gestão de Incidentes.

6.5.7. Anonimização, Mascaramento e Proteção de Dados

A proteção de dados pessoais e informações sigilosas é assegurada por meio da adoção de acordos de confidencialidade assinados por todos os colaboradores e fornecedores da Kaffa, visando garantir o sigilo e a não divulgação indevida de informações sensíveis, também é realizada a aplicação de controles de segurança da informação que serão detalhados a seguir. Os dados pessoais de clientes tratados pela Kaffa, como nome completo e endereço, permanecem armazenados em banco de dados de propriedade do cliente, com acesso controlado e sujeito à gestão de acesso definida pelo próprio cliente, mediante credenciais de acesso, uso de VPN e mecanismos de validação temporária por hashes.

Quando há necessidade de acesso ao banco de dados do cliente para execução de atividades específicas, a Kaffa realiza previamente a análise dos dados estritamente necessários à finalidade pretendida e, na sequência, aplica a anonimização dos dados pessoais e confidenciais. Quando o uso dos dados for indispensável, é realizado o mascaramento das informações por meio de randomização dos valores contidos nas colunas que contenham dados pessoais ou confidenciais. Todo dado pessoal acessado pela Kaffa é mapeado e comunicado ao encarregado de privacidade (DPO) do Grupo Imagem, para adequação do tratamento conforme os requisitos da LGPD.

O mascaramento também é aplicado para restringir a visualização de dados confidenciais em ambientes de desenvolvimento, testes e homologação, permitindo que apenas usuários autorizados tenham acesso às informações completas. Além disso, todas as senhas e dados sigilosos são armazenados e transmitidos utilizando algoritmos de criptografia e hash seguros.

No contexto de segurança de aplicações mobile, todos os dispositivos possuem bloqueio de acesso por senha, sendo exigido o uso de VPN e credenciais de acesso com usuário e senha para acesso a aplicação. Após o acesso à aplicação, é possível consultar informações do banco de dados do aplicativo, o qual possui criptografia. Também é possível utilizar a aplicação em modo offline, no qual a senha do usuário é armazenada de forma criptografada por hash.

- **Tamanho de chave para cifradores**

Diretriz aplicada: SHA 256

- **Modo de cifrador de bloco**

Diretriz aplicada: *Advanced Encryption Standard (AES)*

- **Requisitos cifradores**

Diretriz aplicada: Somente chave sigilosa, melhor ataque força-bruta (*pentest*).

- **Requisitos função de hash criptográfico**

Diretriz aplicada: Usar *salt* sempre que possível.

6.5.8. Senhas

A política de senhas dos colaboradores da Kaffa segue as diretrizes estabelecidas no PO-TI-05 – Política de Gestão de Acessos, garantindo a segurança e a integridade dos acessos aos sistemas. As senhas devem ter no mínimo 8 caracteres, e para usuários internos (AD), é exigida a variação de quatro tipos de

caracteres (letras maiúsculas, minúsculas, números e caracteres especiais). A troca periódica ocorre a cada 90 dias para usuários internos (AD), e todas as senhas são armazenadas criptografadas. Além disso, o número máximo de tentativas de login para usuários internos é de 5 tentativas.

Serviço de autenticação no domínio corporativo o serviço de identidade que centraliza o acesso a Azure DevOps, Azure Portal, Sharepoint e outros serviços, permitindo login via credenciais corporativas (SSO), tokens (PAT) para CLI/automações e Services. A autenticação em duas etapas (MFA) adiciona uma camada extra de segurança exigindo um segundo fator (Microsoft Authenticator, chave FIDO2 ou biometria). O administrador controla tudo via Conditional Access (políticas de acesso condicional), gerenciamento de grupos e permissões, exigência de MFA, restrição por dispositivo/localização, e monitoramento por audit logs — podendo revogar acessos a qualquer momento.

6.5.9. Ciclo de vida do software

As solicitações de desenvolvimento de *softwares* devem vir de três lugares:

1. Solicitações da diretoria, visando uma estratégia de negócio;
2. Solicitações referente a um projeto, ou seja, para atender a um cliente/*stakeholder*;
3. Solicitações internas de melhorias e/ou correções de *bugs*.

A Diretoria Executiva junto com o Project Manager e com o Product Delivery Manager define as prioridades de desenvolvimento macro, ou seja, grandes desenvolvimentos, principalmente referente a estratégia da empresa. Geralmente com prazos mais longos. O Product Owner com a equipe de desenvolvimento definem as demais prioridades referentes a solicitações internas e/ou de projetos.

São definidas as tarefas para serem executadas em uma *sprint* de duas semanas. Portanto, no início do s*print*, é feita uma reunião de pré planejamento e posteriormente uma reunião de planejamento definindo as tarefas a serem executadas. Durante a *sprint*, são realizadas reuniões rápidas para checar o andamento das tarefas, bem como incluir novas solicitações com definição de prioridade ou readequar as tarefas planejadas. Após o término da *sprint*, as atividades que passaram pela validação são disponibilizadas para os usuários.

Todos esses controles e gerenciamentos são realizados pelo Product Owner, Analista de Negócios, Liderança Técnica, Gerente de Projeto e Gerente de Produto.

- Documentação e codificação

  Diretriz aplicada: O sistema/aplicação contém documentação digital, visível para a equipe de desenvolvimento, armazenada no repositório de dados do próprio sistema, dentro da plataforma *GitLab; e Azure Devops*. A documentação contém o objeto do sistema, sua arquitetura e os requisitos para o desenvolvimento.

6.5.10. Gestão de Vulnerabilidades e Achados de Segurança

A equipe de desenvolvimento executa um processo automatizado de varredura no código-fonte, centraliza os resultados de scanners e testes, organiza as falhas identificadas, evita duplicidades e acompanha o ciclo de correção. Além disso, a ferramenta utilizada (DefectDojo) permite gerar relatórios, atribuir responsáveis e integrar-se a outras soluções de segurança e automação

6.5.11. Ciclo de Validação e Testes

O desenvolvedor obtém uma cópia do código fonte em sua máquina para realizar o desenvolvimento. Após realizar a implementação de uma nova rotina ou de

uma correção de um erro, o desenvolvedor executa os *Testes Unitários* para verificar se a implementação foi realizada com sucesso. Depois, ele executa os *Testes Automáticos de Unidade e de Integração*, que estão localizados juntos ao sistema, esses testes têm o objetivo de validar as rotinas mais importantes do sistema, a fim de determinar se essa alteração ocasionou algum erro em outra parte do sistema. Após essas validações, o código fonte é enviado para o repositório central por um processo chamado *Merge Request*. Esse *merge request* é validado por outro desenvolvedor que segue com a execução do *Code Review* (revisão de códigos). Nesse processo, o outro desenvolvedor analisa o código que foi criado/alterado para identificar erros tanto na codificação quanto nas regras de negócio.

Após validação do analista de qualidade e da equipe de produtos ou projetos, o sistema é publicado em ambiente de produção

6.5.12. Ambiente de desenvolvimento

- **Armazenamento do código fonte**

  Diretriz aplicada: O armazenamento e gerenciamento dos códigos fontes dos sistemas é feito pela ferramenta *GitLab e Azure Devops.*

- **Acesso ao código fonte.**

  Diretriz aplicada.: O acesso ao código fonte é gerenciado pela ferramenta *GitLab e Azure Devops*, onde cada desenvolvedor tem acesso apenas ao código fonte da aplicação que está trabalhando.

- **Segregação dos ambientes (Teste, Homologação e Produção).**

Diretriz aplicada: O *deploy* das aplicações nos ambientes é gerenciado pela ferramenta *GitLab e Azure Devops*, apenas os desenvolvedores autorizados têm acesso ao ambiente de produção.

O deploy das aplicações é feito por meio de ferramentas internas que utilizam os artefatos gerados no GitLab e encaminham para o ambiente de acordo com configurações da ferramenta.

O deploy das aplicações do PGR é realizado por meio do Azure Pipelines, utilizando os artefatos gerados a partir do Azure Repos. Cada pipeline é configurado para direcionar automaticamente os artefatos aos ambientes previamente definidos.

6.5.13. Retenção de Dados

A retenção de dados é realizada em conformidade com a legislação aplicável e as melhores práticas de segurança da informação. Os prazos e critérios para armazenamento, eliminação ou anonimização de dados variam de acordo com a natureza da informação e os requisitos regulatórios. A gestão da retenção considera a necessidade de proteção, integridade e disponibilidade dos dados, garantindo que apenas informações essenciais sejam mantidas pelo tempo necessário.

7. ANEXOS

Não se aplica.

8. REFERÊNCIAS

- NBR ISO/IEC 27001:2022
- NBR ISSO/IEC 27002:2022
- *Scrum Guide* (Novembro, 2020 -Ken Schwaber e Jeff Sutherland),
- M-TI-01 - Manual do Sistema de Gestão de Segurança da Informação
- PO-TI-03 - Política de Criptografia

- PO-TI-04 - Política de *Backup* e *Disaster Recovery*
- PO-TI-05 - Política de Gestão de Acessos
- PR-TI-03 - Gestão de Incidentes
- P-DSC-02 - Atendimento AMS EGW KAFFA
- P-DSC-03 - Atendimento AMS GSE
- P-DSC-04 - Atendimento AMS PGR
- P-DP-01 - Desenvolvimento do Produto e Implementação de Melhorias EGW;
- P-DP-02-Comitê de Demandas de Melhorias EGW;
- P-DP-03-Correção de Bugs EGW;
- P-DP-02 - Desenvolvimento do Produto PGR e
- P-DI-01- Desenvolvimento – Implantação
- P-DI-02 – Gerenciamento de Projetos
- PO-GOV-30 Política De Retenção Exclusão e Mascaramento de Dados

9. SIGLAS

- **GUID (*Globally Unique Identifier*):** é um número de 128 *bits* usado para identificar informações em sistemas de computação.
- ***AD* (*Active Directory*):** É um serviço de diretório (que contém informações essenciais sobre o ambiente, incluindo os usuários, computadores, recursos de rede, impressoras, etc. existentes e quem tem permissão para fazer o quê) e um conjunto de serviços (que certificam de que cada pessoa é quem afirma ser "autenticação", geralmente verificando a *ID* do usuário e a senha inserida, e permitem que acessem apenas os recursos que estão autorizados a usar).
- ***DEV*:** refere-se ao ambiente de Desenvolvimento.
- **PRD:** refere-se ao ambiente de Produção.
- **HOM:** refere-se ao ambiente de Homologação.

- ***AES (Advanced Encryption Standard)*:** padrão de criptografia avançada é uma especificação para a criptografia de dados eletrônicos estabelecida pelo Instituto Nacional de Padrões e Tecnologia dos EUA.
- ***SHA-256 (Secure Hash Algorithms*):** função *hash* criptográfica projetada pela Agência de Segurança Nacional dos Estados Unidos. É um Padrão Federal de Processamento de Informação dos Estados Unidos publicado pelo Instituto Nacional de Padrões e Tecnologia (*NIST*).
- **VPN (Virtual Private Network):** Rede privada virtual para acesso seguro a recursos internos.
- **SSH (Secure Shell):** Protocolo de comunicação segura por meio de chaves criptográficas.
- **CI/CD (Continuous Integration / Continuous Delivery):** Práticas de integração e entrega contínua de software.
- **MR (Merge Request):** Solicitação de mesclagem de código ao repositório principal.