Documentação

SysPAD

Sistema de Proteção de Dados Baseado em Técnicas de Encriptação e Anonimização

Fortaleza

23 de Julho de 2023

1. Introdução

O projeto do Sistema de Proteção de Dados Baseado em Técnicas de Encriptação e Anonimização (SysPAD) foi criado para a participação do programa FRIDA, em uma parceria da Universidade Estadual do Ceará (UECE) com o Registro de Endereçamento da Internet para a América Latina e o Caribe (LACNIC), na categoria Internet Aberta e Livre. No contexto da categoria citada, o objetivo do projeto que era trazer uma forma inovadora para proteger os dados sensíveis de usuários para diversos clientes através de técnicas de anonimização e computação em nuvem, foi alcançado, também trazendo um foco em técnicas de encriptação para uma maior cobertura de possíveis situações de vazamentos de dados armazenados em serviços de nuvem.

Dessa forma, o SysPAD têm avançado com novas funcionalidades, tecnologias e algoritmos de encriptação e anonimização de dados para cada vez mais cumprir seu papel de manter os dados do cliente seguros contra vazamentos de dados, tanto em um banco de dados armazenado localmente, quanto em um banco de dados armazenado em uma nuvem.

2. Descrição

Tendo em vista o objetivo do projeto, o SysPAD possui uma estrutura robusta e segura que mantém os dados do usuário seguros da seguinte maneira: ao cadastrar um banco e iniciar o processo da funcionalidade de proteção de dados, o usuário recebe uma cópia anonimizada do próprio banco, enquanto o banco original é armazenado no serviço de nuvem de forma encriptada. Além disso, a ferramenta conta com um subsistema chamado Agent, que funciona como um monitoramento das operações realizadas no banco de dados do cliente. Dessa forma, a preocupação com um dos principais problemas que empresas e organizações enfrentam atualmente é mitigada com as técnicas de segurança usadas no SysPAD: o vazamento de dados.

O sistema foi projetado para dar suporte aos bancos de dados MySQL e PostgreSQL - dois dos sistemas mais populares e utilizados atualmente - mas de forma adaptável, permitindo a possibilidade de ampliação dos tipos de banco de dados suportados, no futuro. Dessa forma, garantindo a permanência da relevância do sistema no cenário tecnológico.

Além disso, o aplicativo Web do sistema possui uma interface intuitiva e funcionalidades essenciais para a utilização descomplicada por qualquer tipo de usuário, adotando uma conduta no-code em que absolutamente todo o processo é feito sem que o usuário precise se preocupar em entender os detalhes técnicos do projeto.

3. Diagrama de Arquitetura

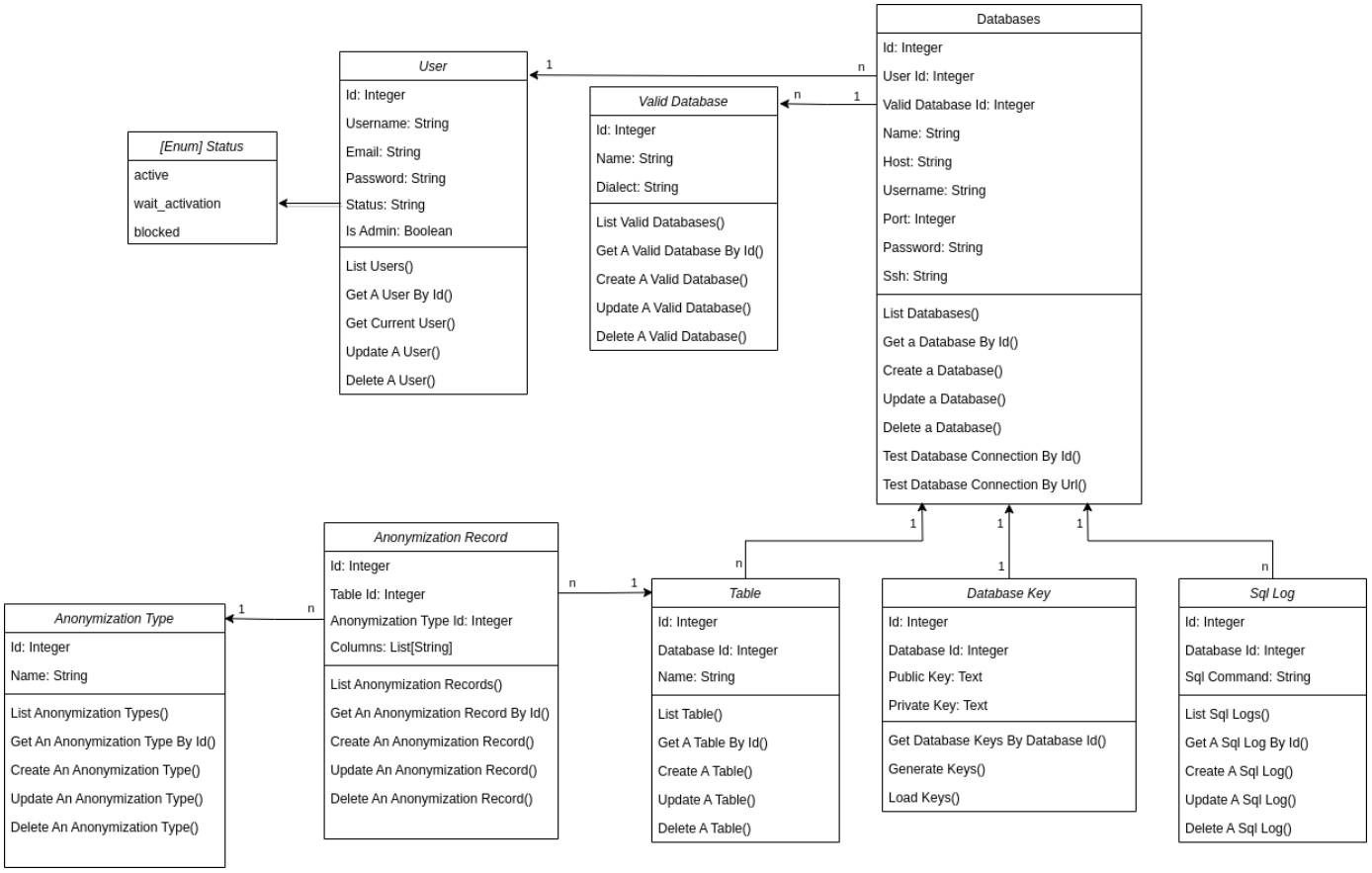

Figura 1: Diagrama de Classes

O projeto se encontra organizado conforme a estrutura visualizada na imagem acima. Nessa representação, é possível observar a presença dos usuários, cada um com seus respectivos bancos de dados que contém suas tabelas, chaves de encriptação e registros de logs. Além disso, foram estabelecidos os registros de anonimização (do inglês, anonymization record) como configuração dos dados sensiveis, visando garantir a proteção desses dados, seguindo diferentes tipos de anonimização, de acordo com as necessidades específicas.

4. Subsistemas

4.1. Interface do Aplicativo Web

A escolha pela construção de um aplicativo Web teve como objetivo simplificar a utilização da ferramenta pelo usuário final. Com isso em mente, a construção da interface do usuário visava simplicidade e intuitividade, para além de uma interface atraente. O público alvo que se foi imaginado para esse projeto são profissionais de TI que lidem com banco de dados e que queiram proteger da melhor forma possível os dados dos clientes de forma que a empresa fique completamente alinhada com a Lei Geral de Proteção de Dados. Com esse usuário em mente, criamos um fluxo simples e intuitivo que exige do usuário apenas alguns clicks para desfrutar de uma proteção de dados com técnicas que avançam no estado-da-arte atual.

4.1.1 Interface do Aplicativo Web

Todas as funcionalidades do SysPAD existem com o objetivo de ajudar o usuário na tarefa de proteger seus dados com técnicas de anonimização, encriptação e computação em nuvem. O objetivo foi criar um sistema completo, mas sem perder a simplicidade, para que até pessoas sem tanto entendimento sobre banco de dados e técnicas de anonimização, consigam utilizar a ferramenta de forma intuitiva.

Área do Cliente:

- Registro
  - O usuário preenche seu **nome de usuário, endereço de e-mail** e **senha**, e recebe um e-mail de confirmação para poder ativar sua conta e poder fazer o login no sistema, posteriormente. É necessário para acessar as ferramentas.
- Login
  - O usuário preenche seu **endereço de e-mail** e sua **senha** já registrados anteriormente para poder acessar a **Área do Cliente** e as demais funcionalidades da ferramenta. A conta deve ter sido ativada no momento do registro.
- Trocar a senha
  - O usuário preenche seu **endereço de e-mail** e recebe um e-mail autorizando a mudança de senha. Depois disso, ele pode trocar para uma outra senha de sua escolha
- Cadastro de Bancos de Dados

    - O usuário pode cadastrar um ou mais banco de dados para poder utilizar a ferramenta de proteção de dados sensíveis. Para isso, o usuário deve inserir **`host, usuário, senha, nome e tipo de database`** e pode testar a conexão do banco antes de adicionar.
- Visualização de Bancos de Dados
    - O usuário consegue **`visualizar`** todos os bancos de dados cadastrados, **`editar, testar conexão`** ou **`excluir`** cada um deles. Para aparecer na tabela de visualização, é necessário que o banco esteja cadastrado, mas não precisa estar com a conexão ativada.
- Proteção de Dados
    - Nessa área, o usuário deve escolher um **`banco de dados,`** e, em seguida, uma **`tabela.`** Com isso, é possível escolher um tipo de anonimização diferente para cada tabela e, após isso, o banco de dados do cliente é atualizado com as colunas anonimizadas e uma cópia da tabela original é enviada, encriptada, para a nuvem de armazenamento.

Área do Administrador:

- Registrar usuário
    - O administrador pode criar novos usuários, precisando fornecer **`nome de usuário, endereço de e-mail`** e **`senha,`** e a conta também precisa ser ativada.

- Visualizar usuários
    - O administrador consegue **`visualizar`** todos os usuários cadastrados, **`editar`** ou **`excluir`** cada um deles.

4.2 API de Proteção de Dados

A API (Application Programming Interface) do Syspad consiste em um modelo RESTFul (Representational State Transfer) que permite a interação de diferentes sistemas de forma eficiente por meio de um conjunto de diretrizes arquiteturais baseadas no protocolo HTTP, implementando métodos padronizados como GET, POST, PUT e DELETE.

4.2.1 Arquitetura e Organização

Essa API foi construída usando o padrão de arquitetura Model-View-Controller (MVC). Ele divide a aplicação em três camadas interconectadas: o Modelo (Model), a Visão (View) e o Controlador (Controller). Esse

padrão é conhecido por reduzir as dependências entre diferentes partes da aplicação, facilitando o trabalho com ela.

A camada Model contém todos os modelos de dados que serão armazenados no banco de dados. A camada View é responsável por exibir os dados ao usuário e simplificar a comunicação entre o usuário e o software. É onde a interface da aplicação é projetada e implementada. A camada Controller trata a entrada do usuário e interage tanto com a camada Model quanto com a camada View para controlar a aplicação.

No caso de Syspad, os diretórios do projeto são divididos em quatro pastas para refletir as diferentes camadas:

- Model
  - Contém todos os modelos para a construção das tabelas do banco de dados do sistema.

- Controller
  - Contém todas as rotas para cada solicitação e a documentação das mensagens retornadas por essas solicitações.

- Seeders
  - Contém os registros de testes para popular o banco de dados.

- Service
  - Contém todos os serviços, funções que serão chamadas durante a operação da API.

Em geral, o uso do padrão MVC na construção do Syspad permitiu um código mais organizado. Isso ajudou a reduzir a quantidade de código necessária para construir a aplicação e tornou mais fácil fazer mudanças, adicionar novos recursos ou refatorar.

4.2.2 Tecnologias Envolvidas

Durante o desenvolvimento do sistema, foram utilizadas algumas tecnologias e bibliotecas. A linguagem de programação principal utilizada para este projeto foi o Python, para construir as rotas e lidar com as solicitações HTTP foi utilizado o framework Flask. Para interagir com o banco de dados, a biblioteca SqlAlchemy foi usada para facilitar o gerenciamento e manipulação de dados. Para tornar mais fácil os testes e visualização dos endpoints da API, foi utilizado o Swagger como ferramenta de documentação. Juntas, essas tecnologias e bibliotecas contribuíram para criar a Frida.

4.2.3 Operações

User:

- Create new user
  - Cria um novo usuário e recebe como entrada um `username`, `email` e `senha`.

- List all registered users
  - Retorna uma lista com todos os usuários criados no sistema. É possível filtrar por `username`.

- Activate user
  - Ativa a conta do usuário a partir do `token` enviado para o email usado no cadastro do usuário.

- Check activation token
  - Verifica se o `token` passado é válido ou não antes de fazer a ativação de um usuário.

- Get current user
  - Retorna os dados do usuário que está logado no momento.

- Resend activation email
  - Reenvia o email de ativação ao usuário, recebe de entrada o `id do usuário` em questão.

- Get user by id
  - Retorna as informações de cadastro de um usuário a partir do `id do usuário`.

- Delete user
  - Apaga um usuário do banco de dados a partir do `id do usuário`.

Auth:

- Login
  - Recebe o `email` e `senha` do usuário cadastrado e faz o login do usuário no sistema.

Password:

- Change password when logged in
  - Altera a senha de um usuário atualmente logado no sistema. Recebe de entrada a `senha atual`, uma `nova senha` e uma `confirmação da nova senha`.

- Check password reset token
  - Verifica se o `token` passado é válido ou não antes de fazer a alteração da senha de um usuário .

- Forgot password
  - Envia um email de recuperação de senha com um token ao usuário que perdeu a senha original.

- Redefine password
  - Altera a senha do usuário que perdeu a senha, recebe de entrada a `senha nova`, uma `confirmação da senha` nova e o `token` enviado por email na solicitação.

Valid Database:

- Create new valid database
  - Cria um novo database válido, recebe como entrada o `nome` do database válido.

- List all registered valid databases
  - Retorna uma lista com todos os tipos de banco de dados válidos cadastrados no sistema. É possível filtrar por `nome`.

- Get valid database by id
  - Retorna as informações de um database válido a partir do `id do database válido`.

- Update valid database
  - Atualiza as informações de um database válido, recebe o `id do database válido` a ser atualizado e a `informação` a ser atualizada.

- Delete valid database
  - Apaga um database válido do banco de dados a partir do `id do database válido`.

Database:

- Create new database
  - Cria um novo database. Recebe de entrada o `id de um database válido`, o `nome` do database, o `host`, um `username`, uma `porta` e uma `senha`.

- List all registered databases of each user
  - Retorna uma lista com todos os databases criados por cada usuário e a lista de todos os databases criados por todos os usuários caso seja administrador. Sendo possível filtrar pelo nome do database.

- Test database connection by url
  - Testa a conexão de um database pela sua url, criada automaticamente após receber de entrada o `id de um database válido`, o `nome` do database, o `host`, um `username`, uma `porta` e uma `senha`.

- Get database by id
  - Retorna as informações de um database a partir da entrada do `id do database.`

- Update database
  - Atualiza informações de um database, a partir do `id do database`, recebe como entrada o `id de um database válido`, o `nome` do database, o `host`, um `username`, uma `porta` e uma `senha` para atualizar.

- Delete database
  - Deleta um database do sistema a partir do `id do database` a ser deletado.

- Create new table
  - Cria uma nova tabela em um determinado database já cadastrado. Recebe como entrada o `id de um database` e o `nome` da tabela.

- List all registered tables
  - Retorna uma lista com todas as tabelas existentes em um database, a partir do `id do database`. É possível filtrar as tabelas por nome.

- Get table by id
  - A partir do `id do database` e do `id da tabela`, retorna as informações da tabela: nome, status de encriptação, progresso de encriptação, status de anonimização e encriptação, progresso de anonimização, e progresso de

retirada de anonimização.

- Update table
  - Atualiza as informações de uma tabela, a partir do **`id do database`** e do **`id da tabela`**, recebendo um novo **`nome`** para ser atualizado.

- Delete table
  - Deleta uma tabela a partir do **id do database** e do **id da tabela**.

- Get column names for each table
  - Retorna o nome de todas as colunas e seu tipo de uma tabela a partir do **`id do database`** e do **`id da tabela`**.

- Get sensitive column names for each table
  - Retorna os nomes das colunas de conteúdo sensível a partir do **`id do database`** e do **`id da tabela`**.

- Test database connection
  - Verifica a conexão de um database a partir do **`id do database`**.

Anonymization Type:

- Create new anonymization type
  - Cria um novo tipo de anonimização, recebe como entrada o **`nome`** do tipo de anonimização.

- List all registered anonymization types
  - Retorna uma lista com todos os tipos de anonimização cadastrados no sistema. É possível filtrar por **`nome`**.

- Get anonymization type by id
  - Retorna as informações de um tipo de anonimização a partir do **`id do tipo de anonimização`**.

- Update anonymization type
  - Atualiza as informações de um tipo de anonimização, recebe o **`id do tipo de anonimização`** a ser atualizado e o novo **nome** a ser recebido.

- Delete anonymization type
  - Deleta um tipo de anonimização a partir do **`id do tipo de anonimização`**.

Anonymization Record:

- Create new anonymization record
  - Cria um novo registro de anonimização, recebe de entrada um **`id de database`**, um **`id de tabela`**, um **`id de tipo de anonimização`** e o **`nome das colunas`** a serem anonimizadas.

- List all registered anonymization records of each user
  - Retorna uma lista com todos os registro de anonimização criados por cada usuário, é possível filtrar pelo id do database.

- Update anonymization record
  - Atualiza as informações de um tipo de anonimização, recebe o **`id do registro de anonimização`** a ser atualizado e a **`informação`** a ser atualizada.

- Delete anonymization record
  - Deleta um registro de anonimização a partir do **`id do registro de anonimização`**.

Encryption:

- Encrypt database table
  - Criptografa uma tabela de um banco de dados. Recebe de entrada o **`id do database`** e o **`id da tabela`** a ser criptografada.

- Encrypt database rows
  - Criptografa linhas (um ou mais) de uma tabela de um banco de dados, recebe de entrada o **`id do database`**, o **`id da tabela`** e as **`linhas`** a serem criptografadas.

- Decrypt database row
  - Descriptografa uma linha de uma tabela, recebe de entrada o **`tipo de valor`** a ser pesquisado (valor da chave primária ou hash da linha) e o **`valor`**, além do **`id do database`**, o **`id da tabela`**.

- Get encryption progress
  - Retorna o progresso de 0 a 100 da criptografia de uma tabela, recebe de entrada o **`id do database`** e o **`id da tabela`**.

Anonymization

- Anonymize database table
  - Anonimiza uma tabela de um database, recebe de entrada o **`id do database`** e o **`id da tabela`** a ser anonimizado.

- Anonymize database rows
  - Anonimiza linhas (uma ou mais) de uma tabela de um database, recebe de entrada o **`id do database`**, o **`id da tabela`** e as **`linhas`** da tabela a serem anonimizadas.

- Get anonymization progress
  - Retorna o progresso de 0 a 100 da anonimização de uma tabela, recebe de entrada o **`id do database`** e o **`id da tabela`**.

- Remove table anonymization
  - Desfaz o processo de anonimização de uma tabela, recebe como entrada o **`id do database`** e o **`id da tabela`**.

- Get remove anonymization progress
  - Retorna o progresso de 0 a 100 da remoção da anonimização de uma tabela, recebe de entrada o **`id do database`** e o **`id da tabela`**.

Sql log:

- Create new sql log
  - Cria um novo registro de log de uma operação SQL, recebe como entrada um **`id de database`** e o **`comando SQL`** `executado no banco de dados`.

- List all registered sql logs
  - Retorna uma lista com as informações do banco de dados e a lista de comandos SQL, é possível filtrar por comando ou id do database.

- Get sql log by id
  - Retorna as informações do banco de dados e o comando SQL relacionado ao **`id do SQL Log`** passado como entrada.

- Update sql log
  - Atualiza um registro do SQL log, recebe como entrada um **`id do database`**, o novo **`comando SQL`** e o **`id do SQL log`**.

- Delete sql log
    - Deleta um registro do SQL log do banco de dados a partir do **`id do SQL log.`**

4.2.4 Possíveis Retornos

- **User**
    - cód: **201** mensagem: **user created**
    - cód: **400** mensagem: **Input payload validation failed**
    - cód: **401** mensagem: **token not found/token invalid**
    - cód: **403** mensagem: **required administrator privileges**
    - cód: **404** mensagem: **user not found**
    - cód: **409** mensagem: **username in use/ email in use**

- **Auth**
    - cód: **400** mensagem: **Input payload validation failed**
    - cód: **401** mensagem: **password incorrect information**
    - cód: **404** mensagem: **user not found**

- **Password**
    - cód: **200** mensagem: **token valid/password updated**
    - cód: **400** mensagem: **Input payload validation failed**
    - cód: **401** mensagem: **token expired**
    - cód: **404** mensagem: **user not found**
    - cód: **409** mensagem: **token invalid**

- **Valid database**
    - cód: **200** mensagem: **valid database updated/deleted**
    - cód: **201** mensagem: **valid database created**
    - cód: **400** mensagem: **Input payload validation failed**
    - cód: **401** mensagem: **token not found/token invalid**
    - cód: **403** mensagem: **required administrator privileges**
    - cód: **404** mensagem: **valid database not found**
    - cód: **409** mensagem: **valid database already exists/ valid database associated with database**

- **Database**
    - cód: **200** mensagem: **database updated/deleted/connected**
    - cód: **201** mensagem: **database created**
    - cód: **400** mensagem: **Input payload validation failed**
    - cód: **401** mensagem: **token not found/token invalid**
    - cód: **403** mensagem: **required administrator privileges**
    - cód: **404** mensagem: **database not found/table not found**
    - cód: **409** mensagem: **database not connected**
    - cód: **500** mensagem: **internal error getting table columns/names**

- **Anonymization type**
  - cód: **200** mensagem: **anonymization type updated/deleted**
  - cód: **201** mensagem: **anonymization type created**
  - cód: **400** mensagem: **Input payload validation failed**
  - cód: **401** mensagem: **token not found/token invalid**
  - cód: **403** mensagem: **required administrator privileges**
  - cód: **404** mensagem: **anonymization type not found**
  - cód: **409** mensagem: **anonymization type exists**

- **Anonymization record**
  - cód: **200** mensagem: **anonymization record updated/deleted**
  - cód: **201** mensagem: **anonymization record created**
  - cód: **400** mensagem: **Input payload validation failed**
  - cód: **404** mensagem: **anonymization type not found**
  - cód: **500** mensagem: **anonymization record not deleted**

- **Encryption**
  - cód: **200** mensagem: **table encrypted/database rows encrypted/row decrypted**
  - cód: **400** mensagem: **Input payload validation failed**
  - cód: **401** mensagem: **token not found/token invalid**
  - cód: **500** mensagem: **database rows not encrypted**

- **Anonymization**
  - cód: **200** mensagem: **table anonymized/database rows anonymized**
  - cód: **400** mensagem: **Input payload validation failed**
  - cód: **401** mensagem: **token not found/token invalid**
  - cód: **500** mensagem: **database rows not anonymized**

- **Sql log**
  - cód: **200** mensagem: **sql log updated/deleted**
  - cód: **201** mensagem: **sql log created**
  - cód: **400** mensagem: **Input payload validation failed**
  - cód: **404** mensagem: **sql log not found**

4.3 Agente Monitor de Banco de Dados

O Agente Monitor de Banco de Dados é uma aplicação desenvolvida em Flask, um framework de desenvolvimento web em Python, que tem como objetivo acompanhar e registrar as operações de inserção, atualização e exclusão de dados em um banco de dados específico.

Assim é possível assegurar que quaisquer novos dados inseridos ou atualizações feitas sejam imediatamente detectados e automaticamente protegidos por meio da encriptação em nuvem, juntamente com a anonimização dos dados no banco local.

4.3.1 Operações

- **Agent start**
    - Com o identificador do banco de dados do cliente atrelado a suas credenciais, o agente carregará todas as informações do banco de dados selecionado para o monitoramento e salva em um arquivo de configuração.

- **Agent database start**
    - Cria o banco de dados do agente com as mesmas tabelas do banco de dados do cliente. Entretanto, em cada tabela, serão criados apenas os campos **primary key** e **hash line**. Esses campos servirão para mapear e identificar os dados do cliente na cópia criptografada contida na nuvem quando seus dados estiverem anonimizados.

- **Agent verification start**
    - Inicia o processo de verificação para detectar inserções, atualizações e exclusões de dados, tratando-as adequadamente para garantir a proteção dos dados do cliente.

5. Glossário

- **Anonimização**
    - Técnica de proteção de dados que codifica, altera ou remove os dados de forma que estes passem a se tornar irreconhecíveis.

- **Encriptação**
    - É uma forma de decodificação de dados de forma que existe uma chave criptográfica com a qual tanto o remetente quanto o destinatário concordam. O destinatário usa a chave para descriptografar os dados, transformando-os de volta em texto original.

- **Chave de encriptação**
    - É uma sequência de caracteres usada em um algoritmo de criptografia para alterar os dados de forma que pareçam aleatórios.

- **Token**
    - O Token de segurança basicamente se trata de um código numérico exclusivo que tem como objetivo garantir a segurança da conta, e serve como um identificador do usuário, garantindo que é ele quem está realizando certas operações.