GRUPO 04

INTEGRANTES

Beatriz Santina
Caroline Ribeiro
Izabelly Gutierres
Luan Oliveira
Lucas Manhães

PROJETO INTEGRADOR
ESCOPO DO PROJETO

Opportune - CRM

São Paulo
March de 2025

GRUPO 04

INTEGRANTES

Beatriz Santina
Caroline Ribeiro
Izabelly Gutierres
Luan Oliveira
Lucas Manhães

PROJETO INTEGRADOR
ESCOPO DO PROJETO

Opportune - CRM

Relatório solicitado pela Generation Brasil para compor o projeto final. O relatório refere-se ao escopo do projeto integrador.

São Paulo
March de 2025

1. Visão Geral do Projeto

Desenvolver um sistema de CRM (Customer Relationship Management) chamado Opportune, que permita o gerenciamento de clientes, oportunidades de vendas e usuários do sistema. O sistema terá funcionalidades de CRUD (Create, Read, Update, Delete) para cada entidade, além de relacionamentos entre as tabelas para garantir a integridade e organização dos dados. Além do método para lógica de atualização de status de oportunidade de false para True

2. Modelo de Negócio

Descrição:

- **Tipo de Plataforma:** Sistema de CRM para gerenciamento de clientes e oportunidades de vendas.
- **Público-alvo:** Empresas e profissionais que desejam gerenciar seus relacionamentos com clientes, oportunidades de negócios e vendas de forma eficiente.
- **Proposta de Valor:** Oferecer uma solução digital acessível e eficiente para o gerenciamento de clientes e oportunidades de vendas, permitindo o acompanhamento do ciclo de vendas e a melhoria do relacionamento com os clientes.

3. Entidades e Atributos

tb_cliente (Usuários do Sistema):

- ID do usuário - Long Int
- Nome - String
- Email - String
- Senha - String
- Foto - String
- Data - Date

São Paulo
March de 2025

- Relacionamento: OneToMany com tb_oportunidade (um usuário pode cadastrar várias oportunidades).

tb_plano (Oportunidades de Vendas/Negócios):

- ID - Long Int
- Nome - String
- Descrição - String
- Status - String
- Valor - Double
- Data - Date
- Relacionamento: ManyToOne com tb_usuario (uma oportunidade pertence a um usuário).
- Relacionamento: ManyToOne com tb_cliente (uma oportunidade está relacionada a um cliente).

tb_empresa (Informações de Clientes):

- ID - Long Int
- Nome da empresa - String
- CNPJ/CPF - String
- Telefone - String
- Endereço - String
- Senha - String
- Email - String
- Relacionamento: OneToMany com tb_oportunidade (um cliente pode estar associado a várias oportunidades).

4. Principais Funcionalidades (CRUD):

tb_cliente:

Listar Todos: Lista todos os clientes.

- **Endpoint**: GET

Listar Todos por ID: Lista todos os clientes por ID.

- **Endpoint**: GET

Criar/Adicionar: Cadastro de novos clientes no sistema.

- **Endpoint**: POST
- **Dados de Entrada**: {nome, email, senha, foto, data}

Listar por nome: Consulta de informações de clientes.

- **Endpoint**: GET
- **Parâmetros de Consulta**: {nome, email, senha, foto, data}

Atualização: Atualizar dados do cliente.

- **Endpoint**: PUT
- **Dados de Entrada**: {nome, email, senha, foto, data}

Deletar: Excluir usuário.

- **Endpoint**: DELETE

tb_plano (Oportunidades de Vendas/Negócios):

Listar Todas: Lista todas as oportunidades.
- **Endpoint:** GET

Listar por ID: Lista uma oportunidade específica por ID.
- **Endpoint:** GET

Listar por nome: Lista as oportunidades com o nome buscado.

- **Endpoint**: GET
- **Parâmetros de Consulta**: {nome, descricao, status, valor, data, usuario_id, cliente_id}

Criar/Adicionar: Cadastro de novas oportunidades.
- **Endpoint:** POST
- **Dados de Entrada:** {descricao, status, valor, data, usuario_id, cliente_id}

Atualizar: Atualizar dados de uma oportunidade.
- **Endpoint:** PUT
- **Dados de Entrada:** {descricao, status, valor, data, usuario_id, cliente_id}

Deletar: Excluir uma oportunidade.
- **Endpoint:** DELETE

Lógica de Atualização: Status de Oportunidade de false para True

- **Endpoint:** PUT

tb_empresa (Informações de Empresas):

Listar Todos: Lista todos as empresas.

- **Endpoint:** GET

Listar por ID: Lista uma empresa específica por ID.

- **Endpoint:** GET

Listar por nome: Consulta de informações por nome da empresa.

- **Endpoint**: GET
- **Parâmetros de Consulta**: {nome_cliente, cnpj_cpf, telefone, endereco, email}

Criar/Adicionar: Cadastro de novas empresas.

- **Endpoint:** POST
- **Dados de Entrada:** {nome_cliente, cnpj_cpf, telefone, endereco, email}

Atualizar: Atualizar dados de uma nova empresa.

- **Endpoint:** PUT
- **Dados de Entrada:** {nome_cliente, cnpj_cpf, telefone, endereco, email}

Deletar: Excluir uma empresa.

- **Endpoint:** DELETE

5. Segurança da Aplicação com Spring Security

O código implementa a segurança de uma aplicação usando o framework Spring Security, integrando autenticação JWT (JSON Web Token).

5.1 Configuração de Segurança

A configuração de segurança é feita através de uma classe anotada com @Configuration e @EnableWebSecurity. Essa classe define os beans necessários para a segurança, incluindo o serviço de detalhes do usuário (UserDetailsService), codificador de senhas (PasswordEncoder), provedor de autenticação (AuthenticationProvider) e o gerenciador de autenticação (AuthenticationManager).

5.2 Filtro de Autenticação JWT

Um filtro personalizado é implementado para interceptar requisições HTTP e verificar a presença de um token JWT no cabeçalho de autorização. Se o token estiver presente e for válido, o usuário será autenticado. Esse filtro é adicionado à cadeia de filtros de segurança antes do filtro padrão de autenticação por nome de usuário e senha.

5.3 Geração e Validação de Tokens JWT

Um serviço JWT é responsável por gerar e validar tokens JWT. O token é criado assinando-o com uma chave secreta e tem um tempo de expiração definido. Esse serviço também fornece métodos para extrair reivindicações (claims) do token, como o nome de usuário e a data de expiração.

5.4 Detalhes do Usuário e Serviço de Detalhes do Usuário

Uma classe de implementação de UserDetails representa os detalhes do usuário, incluindo nome de usuário, senha e autoridades (permissões). O serviço de detalhes do usuário (UserDetailsServiceImpl) carrega os detalhes do usuário a partir de um repositório, geralmente consultando uma tabela de usuários no banco de dados.

5.5 Segurança HTTP e Políticas de Autorização

A configuração de segurança HTTP define as políticas de autorização para diferentes endpoints da aplicação. Alguns endpoints são permitidos para todos, como os endpoints de login e cadastro, enquanto outros requerem autenticação. A política de gerenciamento de sessões é configurada como "stateless" (sem estado) para reforçar a segurança em APIs RESTful.

5.6 Codificação de Senhas

Para garantir a segurança das senhas armazenadas, um codificador de senhas como BCryptPasswordEncoder é usado para criptografar as senhas dos usuários antes de armazená-las no banco de dados.

Tecnologias Utilizadas

Backend

- **Java:** Linguagem de programação robusta e amplamente utilizada para desenvolvimento backend.
- **Spring Framework:** Framework para desenvolvimento de aplicações Java, fornecendo suporte para injeção de dependência, segurança, e muito mais.
- **Spring Boot:** Extensão do Spring Framework que simplifica a configuração e o desenvolvimento de aplicações standalone e produção-ready.
- **Spring Security:** Mecanismo de segurança para aplicações Java baseadas no Spring Framework, fornecendo autenticação, autorização e proteção contra ataques.
- **Spring Data JPA:** Extensão do Spring Data que simplifica o acesso a bancos de dados relacionais usando JPA (Java Persistence API).

Banco de Dados

- **MySQL:** Sistema de gerenciamento de banco de dados relacional, ideal para armazenar dados estruturados.

Ferramentas de Desenvolvimento e Teste

- **Spring Tool Suite (STS)**: IDE baseada no Eclipse, otimizada para desenvolvimento com Spring Framework.
- **Insomnia:** Ferramenta de teste de APIs que facilita a criação, envio e gerenciamento de requisições HTTP.
- **Hibernate:** Framework de mapeamento objeto-relacional (ORM) para Java, utilizado para facilitar a interação com o banco de dados**.**
- **BCryptPasswordEncoder:** Utilizado para criptografar senhas de forma segura.
- **Maven:** Gerenciador de dependências e ferramenta de automação de build para projetos Java.