|  |
| --- |
| **Fatec São José do Rio Preto**  **Curso Superior de Tecnologia em Informática para Negócios** |
|  |
| **Projeto de Extensão Comunitária**  **(Curricularização)** |
|  |
| **Desenvolvimento de um MÍNIMO PRODUTO VIÁVEL (MPV) de aplicação Web para uma organização** |
|  |
|  |
|  |
|  |
| **Integrantes do Projeto:** |
| **Artur Nonato Beti Pedro** |
|  |
|  |
| **São José do Rio Preto/SP**  **2º Semestre de 2024** |

Sumário

[1. Características do Projeto de Extensão 1](#_Toc178083127)

[2. Informações da Equipe de Trabalho 3](#_Toc178083128)

[3. Etapa 1 – Programação, teste e demonstração em ambiente de testes de uma aplicação Web 4](#_Toc178083129)

[3.1 Interfaces gráficas desenvolvidas 4](#_Toc178083130)

[3.2 Classes, atributos e métodos (pacote *model* com *annotations*) 4](#_Toc178083131)

[3.3 Link do projeto no github (atualizar o READ.me) 4](#_Toc178083132)

[3.4 Vídeo demonstrativo (link do Google Drive ou YouTube) 4](#_Toc178083133)

[3.5 Diagrama do banco de dados 4](#_Toc178083134)

[3.6 Injeção de dados (Script do banco de dados) 4](#_Toc178083135)

[4. Etapa 2 – Implementação de recursos adicionais na base de dados para uma aplicação Web 5](#_Toc178083136)

[4.1 Criação das *stored procedure* para receber os dados do sistema e realizar a operação desejada de *insert, update ou delete* 5](#_Toc178083137)

[4.2 Criar as *views* e *functions* para realizar as consultas necessárias de forma que o sistema receba os dados e possa exibi-los aos usuários 5](#_Toc178083138)

[4.3 Automatizar algumas transações no banco de dados usando *triggers* 5](#_Toc178083139)

[4.4 Configurar o banco de dados para criar perfil de usuário e suas permissões para aumentar a segurança ao acesso aos dados, melhorando a segurança do sistema 5](#_Toc178083140)

[4.5 Automatizar *backup´s* periódicos para o banco de dados do sistema, garantindo que os dados estarão armazenados e seguros, mas possam ser recuperados, caso seja necessário 5](#_Toc178083140)

[5. Etapa 3 – Aplicação de métodos estatísticos para geração de dados e informações numa planilha de Excel 6](#_Toc178083141)

[5.1 Tabelas e Gráficos Estatísticos 6](#_Toc178083142)

[5.2 Distribuição de Frequências e Histogramas 6](#_Toc178083143)

[5.3 Medidas de Posição 6](#_Toc178083144)

[5.4 Medidas de Variabilidade ou Dispersão 6](#_Toc178083145)

[5.5 Análise de Regressão Linear 6](#_Toc178083142)

[5.6 Correlação Linear 6](#_Toc178083143)

[6. Etapa 4 – Estudo da qualidade total dos processos e demandas existentes numa organização 7](#_Toc178083146)

[6.1 Apresentar o negócio, a Missão, a Visão, os valores e os meios de comunicá-los para os colaboradores da empresa 7](#_Toc178083147)

[6.2 Citar um processo que é acompanhado e o(s) indicador(es) utilizado(s), descrevendo a unidade de medida 7](#_Toc178083148)

[6.3 Elaborar o fluxograma de um processo, considerando p modelo da Ficha de Diagnóstico/Levantamento de áreas oferecido 7](#_Toc178083149)

[**6**.4 Sugestão de melhoria no processo, considerando as ferramentas da qualidade 7](#_Toc178083150)

[7. Conclusão 8](#_Toc178083151)

# Características do Projeto de Extensão

|  |  |
| --- | --- |
| **Título** | Desenvolvimento de um MPV de aplicação Web para uma organização (4º Semestre). |
| **Temática** | ☐Programas ☒Projetos  ☐Cursos e Oficinas ☐Eventos ☐Prestação de Serviços |
| **Descrição** | O projeto poderá ser baseado nos resultados de projetos de extensão anteriores. Será realizado o desenvolvimento de um MPV (Mínimo Produto Viável) de uma aplicação Web destinada a organização estudada. |
| **Objetivos** | * Analisar documentações de Engenharia de Software existentes para o entendimento das melhores oportunidades para a criação da aplicação Web; * Agregar novas funções à base de dados persistente, por meio de *stored procedures*, *triggers*, entre outros recursos possíveis; * Implementar uma planilha eletrônica que permita a análise de dados e informações de negócios da empresa estudada a partir de técnicas e/ou indicadores estatísticos (medidas de tendência central, regressão, probabilidade, inferência entre outras); * Desenvolver um estudo de qualidade total e melhoria contínua da organização estudada a partir dos levantamentos dos processos de negócio e demandas existentes já detectadas previamente, apontando possíveis soluções e melhorias. |
| **Carga horária** | * Total: **106** horas/aula |
| **Público-alvo** | Empresas, ONGs, órgãos públicos e entidades sociais. |
| **Ações/Etapas de execução** | 1. Programar, testar e efetuar demonstração em ambiente de testes uma aplicação Web; 2. Implementar recursos adicionais na base de dados para uma aplicação Web; 3. Usar métodos estatísticos para geração de dados e informação numa planilha de Excel; 4. Efetuar um estudo de qualidade total para os processos e demandas existentes numa organização. |
| **Entregas** | * 01 Script de banco de dados, com as instruções SQL e/ou descritivas dos recursos de BD criados; * 01 Código-fonte da aplicação Web projetada em repositório online na internet (Github ou similar); * 01 Documento de planilha eletrônica funcional com tabela aplicando técnicas de Estatística e explicações; * 01 Relatório diagnóstico contendo as principais demandas, técnicas e soluções para a melhoria da qualidade dos processos corporativos considerados. |
| **Instrumentos e procedimentos de avaliação** | **Aluno** – trabalho em grupo, eficácia na realização das tarefas, entrega digital do resultado das tarefas em relatório padronizado de atividade de extensão.  **Projeto** – resultados obtidos, publicação dos resultados em repositório online, envio de resultados para a entidade beneficiada na comunidade ou execução de evento para apresentação dos resultados e integração com representantes da comunidade externa beneficiada. |
| **Componente(s)curricular(es) envolvidos** | * Administração de Banco de Dados (20 horas/aula); * Linguagens de Programação II (40 horas/aula); * Estatística (24 horas/aula). * Gestão da Qualidade e Ambiental (22 horas). |
| **Formas de evidência** | * Entrega de relatório de atividade de extensão com os resultados em cada disciplina. |

# Informações da Equipe de Trabalho

|  |
| --- |
| **Nome do Grupo:** SIGAS – Sistema de Gerenciamento Arte Sagrada |

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr.** | **Nome Completo do(s) Aluno(s)** | **Contatos** | |
| 01 | Artur Nonato Beti Pedro | Email: | anonatopedro@gmail.com |
| Whatsapp: | 17 98823-8239 |
| 02 |  | Email: |  |
| Whatsapp: |  |
| 03 |  | Email: |  |
| Whatsapp: |  |
| 04 |  | Email: |  |
| Whatsapp: |  |
| 05 |  | Email: |  |
| Whatsapp: |  |

# Etapa 1 – Programação, teste e demonstração em ambiente de testes de uma aplicação Web

|  |
| --- |
| **Objetivo**: Analisar documentações de Engenharia de Software existentes para o entendimento das melhores oportunidades para a criação da aplicação Web. |

## Interfaces gráficas desenvolvidas

## Classes, atributos e métodos (pacote *model* com *annotations*)

package br.com.sigas.entities;

import java.time.LocalDate;

import java.time.LocalDateTime;

import jakarta.persistence.\*;

@Entity

@Inheritance(strategy = InheritanceType.JOINED)

@DiscriminatorColumn(name = "tipo\_pessoa", discriminatorType = DiscriminatorType.STRING)

@Table(name = "pessoas")

public class Pessoas {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private long id\_pessoa;

    @Column(name = "tipo\_pessoa", insertable = false, updatable = false)

    private String tipo\_pessoa;

    @Column(name = "nome\_pessoa", nullable = false, length = 100)

    private String nome\_pessoa;

    @Column(name = "email\_pessoa", length = 60)

    private String email\_pessoa;

    @Column(name = "endereco\_pessoa", length = 200)

    private String endereco\_pessoa;

    @Column(name = "tel1\_pessoa", nullable = false, length = 13)

    private String tel1\_pessoa;

    @Column(name = "tel2\_pessoa", length = 13)

    private String tel2\_pessoa;

    @Column(name = "is\_active", nullable = false)

    private Boolean isActive = true;

    @Column(name = "data\_criacao", updatable = false)

    private LocalDateTime dataCriacao;

    @Column(name = "data\_modificacao")

    private LocalDateTime dataModificacao;

    @Column(name = "data\_nascimento", nullable = false)

    private LocalDate dataNascimento;

    @Column(name = "idade", insertable = false, updatable = false)

    private Integer idade; // A idade será gerada automaticamente pelo banco de dados

    @PrePersist

    protected void onCreate() {

        dataCriacao = LocalDateTime.now();

        dataModificacao = LocalDateTime.now();

    }

    @PreUpdate

    protected void onUpdate() {

        dataModificacao = LocalDateTime.now();

    }

    public Pessoas() {

    }

    public Pessoas(long id\_pessoa, String tipo\_pessoa, String nome\_pessoa, String email\_pessoa, String endereco\_pessoa,

            String tel1\_pessoa, String tel2\_pessoa, Boolean isActive, LocalDateTime dataCriacao,

            LocalDateTime dataModificacao, LocalDate dataNascimento, Integer idade) {

        this.id\_pessoa = id\_pessoa;

        this.tipo\_pessoa = tipo\_pessoa;

        this.nome\_pessoa = nome\_pessoa;

        this.email\_pessoa = email\_pessoa;

        this.endereco\_pessoa = endereco\_pessoa;

        this.tel1\_pessoa = tel1\_pessoa;

        this.tel2\_pessoa = tel2\_pessoa;

        this.isActive = isActive;

        this.dataCriacao = dataCriacao;

        this.dataModificacao = dataModificacao;

        this.dataNascimento = dataNascimento;

        this.idade = idade;

    }

    // Getters e Setters

    public long getId\_pessoa() {

        return id\_pessoa;

    }

    public void setId\_pessoa(long id\_pessoa) {

        this.id\_pessoa = id\_pessoa;

    }

    public String getTipo\_pessoa() {

        return tipo\_pessoa;

    }

    public void setTipo\_pessoa(String tipo\_pessoa) {

        this.tipo\_pessoa = tipo\_pessoa;

    }

    public String getNome\_pessoa() {

        return nome\_pessoa;

    }

    public void setNome\_pessoa(String nome\_pessoa) {

        this.nome\_pessoa = nome\_pessoa;

    }

    public String getEmail\_pessoa() {

        return email\_pessoa;

    }

    public void setEmail\_pessoa(String email\_pessoa) {

        this.email\_pessoa = email\_pessoa;

    }

    public String getEndereco\_pessoa() {

        return endereco\_pessoa;

    }

    public void setEndereco\_pessoa(String endereco\_pessoa) {

        this.endereco\_pessoa = endereco\_pessoa;

    }

    public String getTel1\_pessoa() {

        return tel1\_pessoa;

    }

    public void setTel1\_pessoa(String tel1\_pessoa) {

        this.tel1\_pessoa = tel1\_pessoa;

    }

    public String getTel2\_pessoa() {

        return tel2\_pessoa;

    }

    public void setTel2\_pessoa(String tel2\_pessoa) {

        this.tel2\_pessoa = tel2\_pessoa;

    }

    public Boolean getIsActive() {

        return isActive;

    }

    public void setIsActive(Boolean isActive) {

        this.isActive = isActive;

    }

    public LocalDateTime getDataCriacao() {

        return dataCriacao;

    }

    public void setDataCriacao(LocalDateTime dataCriacao) {

        this.dataCriacao = dataCriacao;

    }

    public LocalDateTime getDataModificacao() {

        return dataModificacao;

    }

    public void setDataModificacao(LocalDateTime dataModificacao) {

        this.dataModificacao = dataModificacao;

    }

    public LocalDate getDataNascimento() {

        return dataNascimento;

    }

    public void setDataNascimento(LocalDate dataNascimento) {

        this.dataNascimento = dataNascimento;

    }

    public Integer getIdade() {

        return idade;

    }

}

package br.com.sigas.entities;

import java.time.LocalDate;

import java.time.LocalDateTime;

import com.fasterxml.jackson.annotation.JsonBackReference;

import jakarta.persistence.\*;

@Entity

@DiscriminatorValue("F")

public class PessoasFisicas extends Pessoas {

    @Column(name = "cpf", nullable = false, length = 14, unique = true)

    private String cpf;

    @ManyToOne

    @JoinColumn(name = "id\_pessoa", referencedColumnName = "id\_pessoa", insertable = false, updatable = false)

    @JsonBackReference

    private Pessoas pessoa;

    public PessoasFisicas() {

    }

    public PessoasFisicas(long id\_pessoa, String tipo\_pessoa, String nome\_pessoa, String email\_pessoa,

            String endereco\_pessoa, String tel1\_pessoa, String tel2\_pessoa, Boolean isActive, LocalDateTime dataCriacao,

            LocalDateTime dataModificacao, LocalDate dataNascimento, Integer idade, String cpf, Pessoas pessoa) {

        super(id\_pessoa, tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, isActive,

                dataCriacao, dataModificacao, dataNascimento, idade);

        this.cpf = cpf;

        this.pessoa = pessoa;

    }

    public PessoasFisicas(String cpf, Pessoas pessoa) {

        this.cpf = cpf;

        this.pessoa = pessoa;

    }

    public String getCpf() {

        return cpf;

    }

    public void setCpf(String cpf) {

        this.cpf = cpf;

    }

    public Pessoas getPessoa() {

        return pessoa;

    }

    public void setPessoa(Pessoas pessoa) {

        this.pessoa = pessoa;

    }

}

package br.com.sigas.entities;

import java.time.LocalDate;

import java.time.LocalDateTime;

import com.fasterxml.jackson.annotation.JsonBackReference;

import jakarta.persistence.\*;

@Entity

@DiscriminatorValue("J")

@Table(name = "pessoas\_juridicas")

public class PessoasJuridicas extends Pessoas {

    @Column(name = "cnpj", nullable = false, length = 18, unique = true)

    private String cnpj;

    @Column(name = "razao\_social", nullable = false, length = 100)

    private String razao\_social;

    @ManyToOne

    @JoinColumn(name = "id\_pessoa", referencedColumnName = "id\_pessoa", insertable = false, updatable = false)

    @JsonBackReference

    private Pessoas pessoa;

    public PessoasJuridicas() {

    }

    public PessoasJuridicas(long id\_pessoa, String tipo\_pessoa, String nome\_pessoa, String email\_pessoa,

            String endereco\_pessoa, String tel1\_pessoa, String tel2\_pessoa, Boolean isActive, LocalDateTime dataCriacao,

            LocalDateTime dataModificacao, LocalDate dataNascimento, Integer idade, String cnpj, String razao\_social,

            Pessoas pessoa) {

        super(id\_pessoa, tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, isActive,

                dataCriacao, dataModificacao, dataNascimento, idade);

        this.cnpj = cnpj;

        this.razao\_social = razao\_social;

        this.pessoa = pessoa;

    }

    public PessoasJuridicas(String cnpj, String razao\_social, Pessoas pessoa) {

        this.cnpj = cnpj;

        this.razao\_social = razao\_social;

        this.pessoa = pessoa;

    }

    public String getCnpj() {

        return cnpj;

    }

    public void setCnpj(String cnpj) {

        this.cnpj = cnpj;

    }

    public String getRazao\_social() {

        return razao\_social;

    }

    public void setRazao\_social(String razao\_social) {

        this.razao\_social = razao\_social;

    }

    public Pessoas getPessoa() {

        return pessoa;

    }

    public void setPessoa(Pessoas pessoa) {

        this.pessoa = pessoa;

    }

}

package br.com.sigas.entities;

import java.time.LocalDateTime;

import java.util.List;

import jakarta.persistence.\*;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Entity

@Table(name = "categorias")

public class Categorias {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long id\_categoria;

    @Column(name = "nome\_categoria", nullable = false, length = 100)

    private String nome\_categoria;

    @Column(name = "isActive", nullable = false)

    private boolean isActive;

    @OneToMany(mappedBy = "categoria")

    @JsonIgnoreProperties("categoria")

    private List<Produtos> produtos;

    @Column(name = "data\_criacao", nullable = false, updatable = false)

    private LocalDateTime dataCriacao;

    @Column(name = "data\_modificacao")

    private LocalDateTime dataModificacao;

    @PrePersist

    protected void onCreate() {

        dataCriacao = LocalDateTime.now();

        dataModificacao = LocalDateTime.now();

    }

    @PreUpdate

    protected void onUpdate() {

        dataModificacao = LocalDateTime.now();

    }

    public Categorias() {

    }

    public Categorias(Long id\_categoria, String nome\_categoria, boolean isActive, List<Produtos> produtos,

            LocalDateTime dataCriacao, LocalDateTime dataModificacao) {

        this.id\_categoria = id\_categoria;

        this.nome\_categoria = nome\_categoria;

        this.isActive = isActive;

        this.produtos = produtos;

        this.dataCriacao = dataCriacao;

        this.dataModificacao = dataModificacao;

    }

    public Long getId\_categoria() {

        return id\_categoria;

    }

    public void setId\_categoria(Long id\_categoria) {

        this.id\_categoria = id\_categoria;

    }

    public String getNome\_categoria() {

        return nome\_categoria;

    }

    public void setNome\_categoria(String nome\_categoria) {

        this.nome\_categoria = nome\_categoria;

    }

    public boolean getIsActive() {

        return isActive;

    }

    public void setIsActive(boolean isActive) {

        this.isActive = isActive;

    }

    public List<Produtos> getProdutos() {

        return produtos;

    }

    public void setProdutos(List<Produtos> produtos) {

        this.produtos = produtos;

    }

    public LocalDateTime getDataCriacao() {

        return dataCriacao;

    }

    public void setDataCriacao(LocalDateTime dataCriacao) {

        this.dataCriacao = dataCriacao;

    }

    public LocalDateTime getDataModificacao() {

        return dataModificacao;

    }

    public void setDataModificacao(LocalDateTime dataModificacao) {

        this.dataModificacao = dataModificacao;

    }

}

package br.com.sigas.entities;

import java.math.BigDecimal;

import java.time.LocalDateTime;

import com.fasterxml.jackson.annotation.JsonBackReference;

import jakarta.persistence.\*;

@Entity

@Table(name = "Produtos")

public class Produtos {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long id\_produto;

    @Column(name = "nome\_produto", nullable = false, length = 50)

    private String nome\_produto;

    @Column(name = "descricao", nullable = false, length = 200)

    private String descricao;

    @Column(name = "unidade", nullable = false, length = 10)

    private String unidade;

    @Column(name = "preco\_unidade", nullable = false, precision = 10, scale = 2)

    private BigDecimal preco\_unidade;

    @Column(name = "qtd\_estoque", nullable = false)

    private Integer qtd\_estoque;

    @Column(name = "is\_active", nullable = false)

    private Boolean isActive = true;

    @Column(name = "data\_criacao", updatable = false)

    private LocalDateTime dataCriacao;

    @Column(name = "data\_modificacao")

    private LocalDateTime dataModificacao;

    @ManyToOne

    @JoinColumn(name = "id\_categoria", nullable = false)

    @JsonBackReference

    private Categorias categoria;

    @PrePersist

    protected void onCreate() {

        dataCriacao = LocalDateTime.now();

        dataModificacao = LocalDateTime.now();

    }

    @PreUpdate

    protected void onUpdate() {

        dataModificacao = LocalDateTime.now();

    }

    public Produtos() {

    }

    public Produtos(Long id\_produto, String nome\_produto, String descricao, String unidade, BigDecimal preco\_unidade,

            Integer qtd\_estoque, Boolean isActive, LocalDateTime dataCriacao, LocalDateTime dataModificacao,

            Categorias categoria) {

        this.id\_produto = id\_produto;

        this.nome\_produto = nome\_produto;

        this.descricao = descricao;

        this.unidade = unidade;

        this.preco\_unidade = preco\_unidade;

        this.qtd\_estoque = qtd\_estoque;

        this.isActive = isActive;

        this.dataCriacao = dataCriacao;

        this.dataModificacao = dataModificacao;

        this.categoria = categoria;

    }

    public Long getId\_produto() {

        return id\_produto;

    }

    public void setId\_produto(Long id\_produto) {

        this.id\_produto = id\_produto;

    }

    public String getNome\_produto() {

        return nome\_produto;

    }

    public void setNome\_produto(String nome\_produto) {

        this.nome\_produto = nome\_produto;

    }

    public String getDescricao() {

        return descricao;

    }

    public void setDescricao(String descricao) {

        this.descricao = descricao;

    }

    public String getUnidade() {

        return unidade;

    }

    public void setUnidade(String unidade) {

        this.unidade = unidade;

    }

    public BigDecimal getPreco\_unidade() {

        return preco\_unidade;

    }

    public void setPreco\_unidade(BigDecimal preco\_unidade) {

        this.preco\_unidade = preco\_unidade;

    }

    public Integer getQtd\_estoque() {

        return qtd\_estoque;

    }

    public void setQtd\_estoque(Integer qtd\_estoque) {

        this.qtd\_estoque = qtd\_estoque;

    }

    public Boolean getIsActive() {

        return isActive;

    }

    public void setIsActive(Boolean isActive) {

        this.isActive = isActive;

    }

    public LocalDateTime getDataCriacao() {

        return dataCriacao;

    }

    public void setDataCriacao(LocalDateTime dataCriacao) {

        this.dataCriacao = dataCriacao;

    }

    public LocalDateTime getDataModificacao() {

        return dataModificacao;

    }

    public void setDataModificacao(LocalDateTime dataModificacao) {

        this.dataModificacao = dataModificacao;

    }

    public Categorias getCategoria() {

        return categoria;

    }

    public void setCategoria(Categorias categoria) {

        this.categoria = categoria;

    }

}

package br.com.sigas.entities;

import java.time.LocalDate;

import java.util.List;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

import com.fasterxml.jackson.annotation.JsonManagedReference;

import jakarta.persistence.\*;

@Entity

@Table(name = "operacoes")

public class Operacoes {

    @Id

    @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "operacoes\_seq")

    @SequenceGenerator(name = "operacoes\_seq", sequenceName = "operacoes\_sequence", allocationSize = 1)

    private Integer id\_operacao;

    @ManyToOne

    @JoinColumn(name = "id\_pessoa", nullable = false)

    private Pessoas pessoa;

    @Column(name = "tipo\_operacao", nullable = false)

    private Character tipo\_operacao;

    @Column(name = "data\_operacao", nullable = false)

    private LocalDate data\_operacao;

    @OneToMany(mappedBy = "operacao", cascade = CascadeType.ALL, orphanRemoval = true)

    @JsonIgnoreProperties("operacao")

    @JsonManagedReference

    private List<ItensOperacao> itens\_operacao;

    public Operacoes() {

    }

    public Operacoes(Integer id\_operacao, Pessoas pessoa, Character tipo\_operacao, LocalDate data\_operacao,

            List<ItensOperacao> itens\_operacao) {

        this.id\_operacao = id\_operacao;

        this.pessoa = pessoa;

        this.tipo\_operacao = tipo\_operacao;

        this.data\_operacao = data\_operacao;

        this.itens\_operacao = itens\_operacao;

    }

    public Integer getId\_operacao() {

        return id\_operacao;

    }

    public void setId\_operacao(Integer id\_operacao) {

        this.id\_operacao = id\_operacao;

    }

    public Pessoas getPessoa() {

        return pessoa;

    }

    public void setPessoa(Pessoas pessoa) {

        this.pessoa = pessoa;

    }

    public Character getTipo\_operacao() {

        return tipo\_operacao;

    }

    public void setTipo\_operacao(Character tipo\_operacao) {

        this.tipo\_operacao = tipo\_operacao;

    }

    public LocalDate getData\_operacao() {

        return data\_operacao;

    }

    public void setData\_operacao(LocalDate data\_operacao) {

        this.data\_operacao = data\_operacao;

    }

    public List<ItensOperacao> getItens\_operacao() {

        return itens\_operacao;

    }

    public void setItens\_operacao(List<ItensOperacao> itens\_operacao) {

        this.itens\_operacao = itens\_operacao;

    }

}

package br.com.sigas.entities;

import java.math.BigDecimal;

import com.fasterxml.jackson.annotation.JsonBackReference;

import jakarta.persistence.\*;

@Entity

@Table(name = "itens\_operacao")

public class ItensOperacao {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long id\_item\_operacao;

    @ManyToOne

    @JoinColumn(name = "id\_operacao", nullable = false)

    @JsonBackReference

    private Operacoes operacao;

    @ManyToOne

    @JoinColumn(name = "id\_produto", nullable = false)

    private Produtos produto;

    @Column(name = "quantidade", nullable = false)

    private Integer quantidade;

    @Column(name = "preco\_unitario", nullable = false)

    private BigDecimal preco\_unitario;

    @Column(name = "valor\_total", precision = 10, scale = 2)

    private BigDecimal valor\_total;

    @PrePersist

    @PreUpdate

    public void calcularValorTotal() {

        if (quantidade != null && preco\_unitario != null) {

            this.valor\_total = preco\_unitario.multiply(new BigDecimal(quantidade));

        }

    }

    public ItensOperacao() {

    }

    public ItensOperacao(Long id\_item\_operacao, Operacoes operacao, Produtos produto, Integer quantidade,

            BigDecimal preco\_unitario, BigDecimal valor\_total) {

        this.id\_item\_operacao = id\_item\_operacao;

        this.operacao = operacao;

        this.produto = produto;

        this.quantidade = quantidade;

        this.preco\_unitario = preco\_unitario;

        this.valor\_total = valor\_total;

    }

    public Long getId\_item\_operacao() {

        return id\_item\_operacao;

    }

    public void setId\_item\_operacao(Long id\_item\_operacao) {

        this.id\_item\_operacao = id\_item\_operacao;

    }

    public Operacoes getOperacao() {

        return operacao;

    }

    public void setOperacao(Operacoes operacao) {

        this.operacao = operacao;

    }

    public Produtos getProduto() {

        return produto;

    }

    public void setProduto(Produtos produto) {

        this.produto = produto;

    }

    public Integer getQuantidade() {

        return quantidade;

    }

    public void setQuantidade(Integer quantidade) {

        this.quantidade = quantidade;

    }

    public BigDecimal getPreco\_unitario() {

        return preco\_unitario;

    }

    public void setPreco\_unitario(BigDecimal preco\_unitario) {

        this.preco\_unitario = preco\_unitario;

    }

    public BigDecimal getValor\_total() {

        return valor\_total;

    }

    public void setValor\_total(BigDecimal valor\_total) {

        this.valor\_total = valor\_total;

    }

}

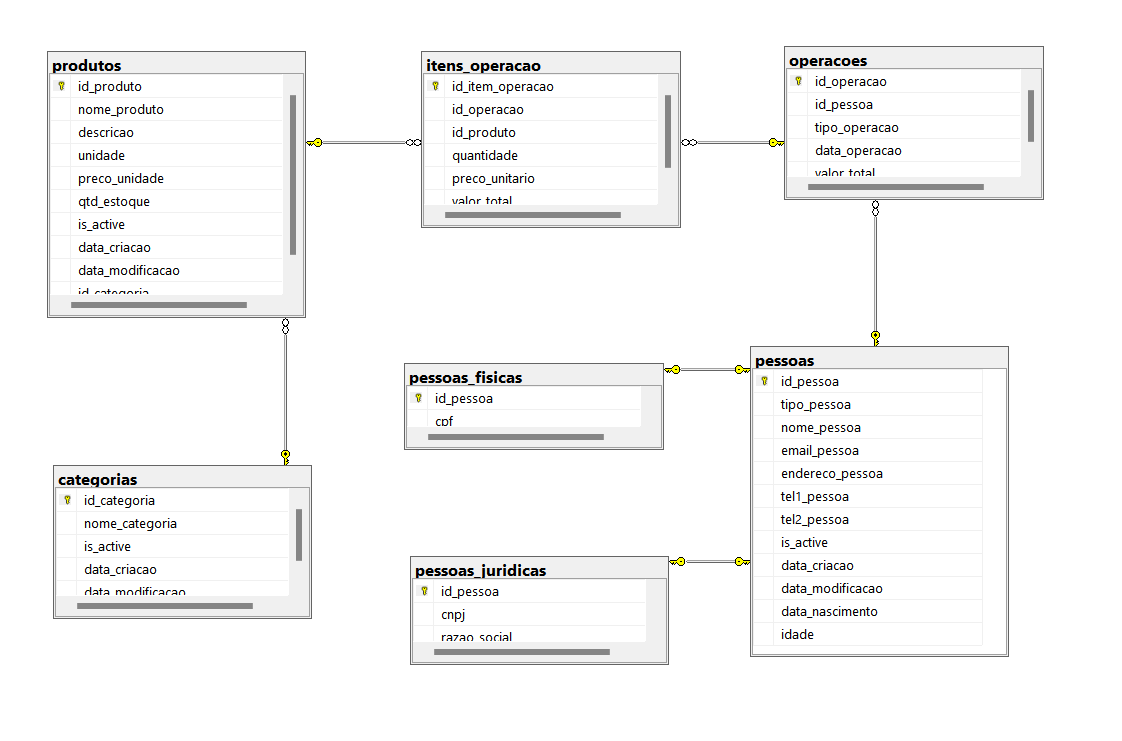
## Link do projeto no github (atualizar o READ.me)

<https://github.com/anonatopedro/Projeto-Extens-o-Curricularizado->

## Vídeo demonstrativo (link do Google Drive ou YouTube)

<https://drive.google.com/drive/folders/13wgcpG86CfUS5P_MLLHJwNf-8-NFMXJI?usp=sharing>

## Diagrama do banco de dados



## Injeção de dados (Script do banco de dados)

contendo no mínimo 10 inserts em cada tabela

INSERT INTO categorias (nome\_categoria, is\_active, data\_criacao)

VALUES

('Shampoo Artesanal', 1, GETDATE()),

('Condicionador', 1, GETDATE()),

('Sabonete Artesanal Líquido', 1, GETDATE()),

('Sabonete Artesanal em Barra', 1, GETDATE()),

('Rapé Artesanal', 1, GETDATE()),

('Santo Cruzeiro', 1, GETDATE()),

('Guia', 1, GETDATE()),

('Difusor de Ambiente', 1, GETDATE()),

('Creme Hidratante', 1, GETDATE()),

('Mistura de Ervas (Jurema)', 1, GETDATE()),

('Terço', 1, GETDATE()),

('Kuripe', 1, GETDATE()),

('Tepi', 1, GETDATE());

INSERT INTO produtos (nome\_produto, descricao, unidade, preco\_unidade, qtd\_estoque, is\_active, data\_criacao, id\_categoria)

VALUES

('Shampoo Artesanal Maracujá', 'Shampoo artesanal de maracujá', 'unidade', 25.50, 50, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Shampoo Artesanal')),

('Shampoo Artesanal Banana', 'Shampoo artesanal de banana', 'unidade', 26.00, 50, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Shampoo Artesanal')),

('Shampoo Artesanal Alecrim', 'Shampoo artesanal de alecrim', 'unidade', 24.90, 50, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Shampoo Artesanal'));

INSERT INTO produtos (nome\_produto, descricao, unidade, preco\_unidade, qtd\_estoque, is\_active, data\_criacao, id\_categoria)

VALUES

('Condicionador Manteiga de Karité', 'Condicionador com manteiga de karité', 'unidade', 30.00, 40, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Condicionador')),

('Condicionador Olibano', 'Condicionador com óleo essencial de olibano', 'unidade', 32.00, 40, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Condicionador')),

('Condicionador Samaúma', 'Condicionador de Samaúma', 'unidade', 28.50, 40, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Condicionador'));

INSERT INTO produtos (nome\_produto, descricao, unidade, preco\_unidade, qtd\_estoque, is\_active, data\_criacao, id\_categoria)

VALUES

('Sabonete Líquido Alecrim', 'Sabonete artesanal líquido de alecrim', 'unidade', 18.00, 60, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Sabonete Artesanal Líquido')),

('Sabonete Líquido Arruda com Sal Grosso', 'Sabonete artesanal líquido de arruda com sal grosso', 'unidade', 20.00, 60, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Sabonete Artesanal Líquido')),

('Sabonete Líquido Lavanda', 'Sabonete artesanal líquido de lavanda', 'unidade', 19.50, 60, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Sabonete Artesanal Líquido')),

('Sabonete Líquido Camomila', 'Sabonete artesanal líquido de camomila', 'unidade', 19.00, 60, 1, GETDATE(), (SELECT id\_categoria FROM categorias WHERE nome\_categoria = 'Sabonete Artesanal Líquido'));

INSERT INTO pessoas (tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, data\_nascimento)

VALUES

('F', 'Carlos Souza', 'carlos.souza@example.com', 'Av. Paulista, 101', '(12) 91000-1001', '(12) 92000-1001', '1981-02-12'),

('F', 'Fernanda Oliveira', 'fernanda.oliveira@example.com', 'Rua Bela Vista, 102', '(13) 91000-1002', '(13) 92000-1002', '1982-03-13'),

('F', 'João Almeida', 'joão.almeida@example.com', 'Rua Nova, 103', '(14) 91000-1003', '(14) 92000-1003', '1983-04-14'),

('F', 'Mariana Santos', 'mariana.santos@example.com', 'Av. Central, 104', '(15) 91000-1004', '(15) 92000-1004', '1984-05-15'),

('F', 'Rodrigo Batista', 'rodrigo.batista@example.com', 'Rua das Palmeiras, 105', '(16) 91000-1005', '(16) 92000-1005', '1985-06-16'),

('F', 'Tatiana Castro', 'tatiana.castro@example.com', 'Rua Aurora, 106', '(17) 91000-1006', '(17) 92000-1006', '1986-07-17'),

('F', 'Gabriel Ferreira', 'gabriel.ferreira@example.com', 'Av. do Sol, 107', '(18) 91000-1007', '(18) 92000-1007', '1987-08-18'),

('F', 'Luciana Pereira', 'luciana.pereira@example.com', 'Rua Verde, 108', '(19) 91000-1008', '(19) 92000-1008', '1988-09-19'),

('F', 'Bruno Lima', 'bruno.lima@example.com', 'Av. Azul, 109', '(20) 91000-1009', '(20) 92000-1009', '1989-01-11'),

('F', 'Juliana Alves', 'juliana.alves@example.com', 'Rua dos Pinhais, 110', '(11) 91000-1010', '(11) 92000-1010', '1980-02-12'),

('F', 'Pedro Henrique', 'pedro.henrique@example.com', 'Av. Independência, 111', '(12) 91000-1011', '(12) 92000-1011', '1981-03-13'),

('F', 'Bianca Monteiro', 'bianca.monteiro@example.com', 'Rua Nova Esperança, 112', '(13) 91000-1012', '(13) 92000-1012', '1982-04-14'),

('F', 'Ricardo Lopes', 'ricardo.lopes@example.com', 'Rua da Alegria, 113', '(14) 91000-1013', '(14) 92000-1013', '1983-05-15'),

('F', 'Patrícia Mendes', 'patrícia.mendes@example.com', 'Av. Brasil, 114', '(15) 91000-1014', '(15) 92000-1014', '1984-06-16');

INSERT INTO pessoas\_fisicas (id\_pessoa, cpf)

VALUES

(1, '001.002.003-07'),

(2, '002.004.006-14'),

(3, '003.006.009-21'),

(4, '004.008.012-28'),

(5, '005.010.015-35'),

(6, '006.012.018-42'),

(7, '007.014.021-49'),

(8, '008.016.024-56'),

(9, '009.018.027-63'),

(10, '010.020.030-70'),

(11, '011.022.033-77'),

(12, '012.024.036-84'),

(13, '013.026.039-91'),

(14, '014.028.042-98'),

(15, '015.030.045-05'),

(16, '016.032.048-12'),

(17, '017.034.051-19');

INSERT INTO pessoas (tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, data\_nascimento)

VALUES

('J', 'Alessandro Nunes', 'alessandro.nunes@grupoexcel.com', 'Av. dos Bandeirantes, 123', '(11) 91001-1001', '(11) 92001-1001', '1982-02-14'),

('J', 'Camila Ribeiro', 'camila.ribeiro@futuroverde.com', 'Rua das Árvores, 456', '(21) 91002-2002', '(21) 92002-2002', '1987-06-08'),

('J', 'Renato Lima', 'renato.lima@logexpress.com', 'Av. Industrial, 789', '(31) 91003-3003', '(31) 92003-3003', '1990-11-30'),

('J', 'Jéssica Souza', 'jessica.souza@mundonovo.com', 'Rua Nova Esperança, 101', '(41) 91004-4004', '(41) 92004-4004', '1985-09-15'),

('J', 'Pedro Henrique', 'pedro.henrique@megaeng.com', 'Av. das Indústrias, 222', '(51) 91005-5005', '(51) 92005-5005', '1992-01-01'),

('J', 'Letícia Alves', 'leticia.alves@vidamais.com', 'Rua Bem Estar, 333', '(61) 91006-6006', '(61) 92006-6006', '1983-08-22'),

('J', 'Vinícius Carvalho', 'vinicius.carvalho@techworld.com', 'Av. do Conhecimento, 444', '(71) 91007-7007', '(71) 92007-7007', '1980-12-12'),

('J', 'Patrícia Mendes', 'patricia.mendes@beautyshop.com', 'Rua da Beleza, 555', '(81) 91008-8008', '(81) 92008-8008', '1988-05-09'),

('J', 'Lucas Barros', 'lucas.barros@novaconstrucao.com', 'Av. Nova, 666', '(91) 91009-9009', '(91) 92009-9009', '1981-03-03'),

('J', 'Mariana Oliveira', 'mariana.oliveira@ecoplan.com', 'Rua Sustentável, 777', '(31) 91010-0010', '(31) 92010-0010', '1989-07-19');

INSERT INTO pessoas\_juridicas (id\_pessoa, cnpj, razao\_social)

VALUES

(151, '01.123.456/0001-11', 'Grupo Excel Contabilidade LTDA'),

(152, '01.234.567/0002-22', 'Futuro Verde Sustentabilidade ME'),

(153, '01.345.678/0003-33', 'LogExpress Transporte e Logística LTDA'),

(154, '01.456.789/0004-44', 'Mundo Novo Comércio de Produtos LTDA'),

(155, '01.567.890/0005-55', 'MegaEng Engenharia e Projetos SA'),

(156, '01.678.901/0006-66', 'Vida Mais Plano de Saúde LTDA'),

(157, '01.789.012/0007-77', 'TechWorld Soluções em TI EIRELI'),

(158, '01.890.123/0008-88', 'BeautyShop Cosméticos LTDA'),

(159, '01.901.234/0009-99', 'Nova Construção Engenharia LTDA'),

(160, '01.012.345/0010-10', 'EcoPlan Consultoria Ambiental SA');

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (1, 'V', '2024-11-29', 0); -- id\_pessoa = 1

DECLARE @id\_operacao\_1 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES

(@id\_operacao\_1, 1, 2, 25.50), -- Shampoo Artesanal Maracujá

(@id\_operacao\_1, 3, 1, 18.00), -- Sabonete Líquido Alecrim

(@id\_operacao\_1, 5, 1, 15.00); -- Rapé Samaúma

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (2, 'V', '2024-11-28', 0); -- id\_pessoa = 2

DECLARE @id\_operacao\_2 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES

(@id\_operacao\_2, 2, 1, 26.00), -- Shampoo Artesanal Banana

(@id\_operacao\_2, 4, 2, 20.00), -- Sabonete Líquido Arruda com Sal Grosso

(@id\_operacao\_2, 6, 3, 16.00); -- Rapé Sansara

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (3, 'V', '2024-11-27', 0); -- id\_pessoa = 3

DECLARE @id\_operacao\_3 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES

(@id\_operacao\_3, 7, 2, 17.00), -- Rapé 3 Ervas

(@id\_operacao\_3, 8, 1, 14.50), -- Rapé Cumaru

(@id\_operacao\_3, 9, 1, 16.50); -- Rapé Mulateiro

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (4, 'V', '2024-11-09', 0);

DECLARE @id\_operacao\_4 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_4, 3, 5, 14.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_4, 4, 2, 16.00);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_4, 2, 4, 13.00);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (5, 'V', '2024-11-30', 0);

DECLARE @id\_operacao\_5 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_5, 2, 3, 13.00);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_5, 4, 4, 16.00);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_5, 5, 3, 17.50);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (6, 'V', '2024-11-09', 0);

DECLARE @id\_operacao\_6 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_6, 1, 3, 11.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_6, 5, 2, 17.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_6, 2, 5, 13.00);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (7, 'V', '2024-11-29', 0);

DECLARE @id\_operacao\_7 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_7, 8, 4, 22.00);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_7, 5, 2, 17.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_7, 4, 3, 16.00);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (8, 'V', '2024-11-11', 0);

DECLARE @id\_operacao\_8 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_8, 6, 4, 19.00);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_8, 9, 2, 23.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_8, 5, 1, 17.50);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (9, 'V', '2024-11-18', 0);

DECLARE @id\_operacao\_9 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_9, 3, 5, 14.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_9, 7, 2, 20.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_9, 6, 5, 19.00);

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao, valor\_total)

VALUES (10, 'V', '2024-11-21', 0);

DECLARE @id\_operacao\_10 INT = SCOPE\_IDENTITY();

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_10, 7, 5, 20.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_10, 1, 4, 11.50);

INSERT INTO itens\_operacao (id\_operacao, id\_produto, quantidade, preco\_unitario)

VALUES (@id\_operacao\_10, 4, 4, 16.00);

# Etapa 2 – Implementação de recursos adicionais na base de dados para uma aplicação Web

|  |
| --- |
| **Objetivo**: Agregar novas funções à base de dados persistente, por meio de *stored procedures*, *triggers*, entre outros recursos possíveis. |

## Criação das *stored procedure* para receber os dados do sistema e realizar a operação desejada de *insert, update ou delete*.

CREATE PROCEDURE inserir\_categoria

@nome\_categoria NVARCHAR(100)

AS

BEGIN

INSERT INTO categorias (nome\_categoria, is\_active, data\_criacao)

VALUES (@nome\_categoria, 1, GETDATE());

END;

CREATE PROCEDURE atualizar\_categoria

@id\_categoria INT,

@nome\_categoria NVARCHAR(100)

AS

BEGIN

UPDATE categorias

SET nome\_categoria = @nome\_categoria,

data\_modificacao = GETDATE()

WHERE id\_categoria = @id\_categoria;

END;

CREATE PROCEDURE deletar\_categoria

@id\_categoria INT

AS

BEGIN

UPDATE categorias

SET is\_active = 0,

data\_modificacao = GETDATE()

WHERE id\_categoria = @id\_categoria;

END;

CREATE PROCEDURE inserir\_produto

@nome\_produto NVARCHAR(50),

@descricao NVARCHAR(MAX),

@unidade NVARCHAR(10),

@preco\_unidade DECIMAL(10, 2),

@qtd\_estoque INT,

@id\_categoria INT

AS

BEGIN

INSERT INTO produtos (nome\_produto, descricao, unidade, preco\_unidade, qtd\_estoque, id\_categoria, is\_active, data\_criacao)

VALUES (@nome\_produto, @descricao, @unidade, @preco\_unidade, @qtd\_estoque, @id\_categoria, 1, GETDATE());

END;

CREATE PROCEDURE atualizar\_produto

@id\_produto INT,

@nome\_produto NVARCHAR(50),

@descricao NVARCHAR(MAX),

@unidade NVARCHAR(10),

@preco\_unidade DECIMAL(10, 2),

@qtd\_estoque INT

AS

BEGIN

UPDATE produtos

SET nome\_produto = @nome\_produto,

descricao = @descricao,

unidade = @unidade,

preco\_unidade = @preco\_unidade,

qtd\_estoque = @qtd\_estoque,

data\_modificacao = GETDATE()

WHERE id\_produto = @id\_produto;

END;

CREATE PROCEDURE deletar\_produto

@id\_produto INT

AS

BEGIN

UPDATE produtos

SET is\_active = 0,

data\_modificacao = GETDATE()

WHERE id\_produto = @id\_produto;

END;

CREATE PROCEDURE inserir\_operacao

@id\_pessoa INT,

@tipo\_operacao CHAR(1),

@data\_operacao DATE

AS

BEGIN

INSERT INTO operacoes (id\_pessoa, tipo\_operacao, data\_operacao)

VALUES (@id\_pessoa, @tipo\_operacao, @data\_operacao);

END;

CREATE PROCEDURE atualizar\_operacao

@id\_operacao INT,

@id\_pessoa INT,

@tipo\_operacao CHAR(1),

@data\_operacao DATE

AS

BEGIN

UPDATE operacoes

SET id\_pessoa = @id\_pessoa,

tipo\_operacao = @tipo\_operacao,

data\_operacao = @data\_operacao

WHERE id\_operacao = @id\_operacao;

END;

CREATE PROCEDURE deletar\_operacao

@id\_operacao INT

AS

BEGIN

DELETE FROM operacoes

WHERE id\_operacao = @id\_operacao;

END;

CREATE PROCEDURE inserir\_pessoa\_fisica

@tipo\_pessoa CHAR(1),

@nome\_pessoa NVARCHAR(100),

@email\_pessoa NVARCHAR(60),

@endereco\_pessoa NVARCHAR(200),

@tel1\_pessoa NVARCHAR(13),

@tel2\_pessoa NVARCHAR(13),

@data\_nascimento DATE,

@cpf NVARCHAR(14)

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

DECLARE @id\_pessoa INT;

INSERT INTO pessoas (

tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, data\_nascimento, is\_active, data\_criacao

)

VALUES (

@tipo\_pessoa, @nome\_pessoa, @email\_pessoa, @endereco\_pessoa, @tel1\_pessoa, @tel2\_pessoa, @data\_nascimento, 1, GETDATE()

);

SET @id\_pessoa = SCOPE\_IDENTITY();

INSERT INTO pessoas\_fisicas (

id\_pessoa, cpf

)

VALUES (

@id\_pessoa, @cpf

);

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

CREATE PROCEDURE atualizar\_pessoa\_fisica

@id\_pessoa INT,

@cpf NVARCHAR(14),

@nome NVARCHAR(100),

@email NVARCHAR(60),

@endereco NVARCHAR(200),

@tel1 NVARCHAR(13),

@tel2 NVARCHAR(13),

@data\_nascimento DATE

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

UPDATE pessoas\_fisicas

SET cpf = @cpf

WHERE id\_pessoa = @id\_pessoa;

UPDATE pessoas

SET

nome\_pessoa = @nome, email\_pessoa = @email, endereco\_pessoa = @endereco, tel1\_pessoa = @tel1, tel2\_pessoa = @tel2,

data\_nascimento = @data\_nascimento, data\_modificacao = GETDATE()

WHERE id\_pessoa = @id\_pessoa;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

CREATE PROCEDURE deletar\_pessoa\_fisica

@id\_pessoa INT

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

UPDATE pessoas

SET is\_active = 0,

data\_modificacao = GETDATE()

WHERE id\_pessoa = @id\_pessoa;

IF NOT EXISTS (SELECT 1 FROM pessoas\_fisicas WHERE id\_pessoa = @id\_pessoa)

BEGIN

THROW 50000, 'Pessoa Física não encontrada.', 1;

END

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

CREATE PROCEDURE inserir\_pessoa\_juridica

@tipo\_pessoa CHAR(1),

@nome\_pessoa NVARCHAR(100),

@email\_pessoa NVARCHAR(60),

@endereco\_pessoa NVARCHAR(200),

@tel1\_pessoa NVARCHAR(13),

@tel2\_pessoa NVARCHAR(13),

@data\_nascimento DATE,

@cnpj NVARCHAR(18),

@razao\_social NVARCHAR(100)

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

DECLARE @id\_pessoa INT;

-- Insere na tabela "pessoas"

INSERT INTO pessoas (

tipo\_pessoa, nome\_pessoa, email\_pessoa, endereco\_pessoa, tel1\_pessoa, tel2\_pessoa, data\_nascimento,is\_active, data\_criacao

)

VALUES (

@tipo\_pessoa, @nome\_pessoa, @email\_pessoa, @endereco\_pessoa, @tel1\_pessoa, @tel2\_pessoa, @data\_nascimento, 1, GETDATE()

);

SET @id\_pessoa = SCOPE\_IDENTITY();

-- Insere na tabela "pessoas\_juridicas"

INSERT INTO pessoas\_juridicas (

id\_pessoa, cnpj, razao\_social

)

VALUES (

@id\_pessoa, @cnpj, @razao\_social

);

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

CREATE PROCEDURE atualizar\_pessoa\_juridica

@id\_pessoa INT,

@nome\_pessoa NVARCHAR(100),

@email\_pessoa NVARCHAR(60),

@endereco\_pessoa NVARCHAR(200),

@tel1\_pessoa NVARCHAR(13),

@tel2\_pessoa NVARCHAR(13),

@data\_nascimento DATE,

@cnpj NVARCHAR(18),

@razao\_social NVARCHAR(100)

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

-- Atualiza a tabela "pessoas"

UPDATE pessoas

SET

nome\_pessoa = @nome\_pessoa,

email\_pessoa = @email\_pessoa,

endereco\_pessoa = @endereco\_pessoa,

tel1\_pessoa = @tel1\_pessoa,

tel2\_pessoa = @tel2\_pessoa,

data\_nascimento = @data\_nascimento,

data\_modificacao = GETDATE()

WHERE id\_pessoa = @id\_pessoa;

-- Atualiza a tabela "pessoas\_juridicas"

UPDATE pessoas\_juridicas

SET

cnpj = @cnpj,

razao\_social = @razao\_social

WHERE id\_pessoa = @id\_pessoa;

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

select \* from pessoas

CREATE PROCEDURE deletar\_pessoa\_juridica

@id\_pessoa INT

AS

BEGIN

BEGIN TRANSACTION;

BEGIN TRY

-- Atualiza a tabela "pessoas" para marcar como inativa

UPDATE pessoas

SET is\_active = 0,

data\_modificacao = GETDATE()

WHERE id\_pessoa = @id\_pessoa;

-- Verifica se a pessoa jurídica existe

IF NOT EXISTS (SELECT 1 FROM pessoas\_juridicas WHERE id\_pessoa = @id\_pessoa)

BEGIN

THROW 50000, 'Pessoa Jurídica não encontrada.', 1;

END

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

THROW;

END CATCH

END;

## Criar as *views* e *functions* para realizar as consultas necessárias de forma que o sistema receba os dados e possa exibi-los aos usuários.

CREATE VIEW vw\_operacoes\_com\_valor\_total AS

SELECT

o.id\_operacao,

o.id\_pessoa,

o.tipo\_operacao,

o.data\_operacao,

SUM(io.valor\_total) AS valor\_total

FROM

operacoes o

LEFT JOIN

itens\_operacao io ON o.id\_operacao = io.id\_operacao

GROUP BY

o.id\_operacao, o.id\_pessoa, o.tipo\_operacao, o.data\_operacao;

CREATE VIEW vw\_estoque\_atual AS

SELECT

P.id\_produto,

P.nome\_produto,

P.qtd\_estoque,

P.unidade,

P.preco\_unidade

FROM Produtos P

WHERE P.qtd\_estoque > 0;

select \* from v

CREATE VIEW vw\_produtos\_estoque\_baixo AS

SELECT

p.id\_produto,

p.nome\_produto,

p.qtd\_estoque,

p.unidade,

p.preco\_unidade

FROM produtos p

WHERE p.qtd\_estoque < 10;

CREATE VIEW vw\_operacoes\_periodo AS

SELECT

o.id\_operacao,

o.tipo\_operacao,

o.data\_operacao,

p.nome\_pessoa AS pessoa\_nome,

p.tipo\_pessoa

FROM operacoes o

JOIN pessoas p ON o.id\_pessoa = p.id\_pessoa;

CREATE VIEW vw\_resumo\_financeiro AS

SELECT

o.tipo\_operacao,

SUM(io.valor\_total) AS total\_valor,

COUNT(o.id\_operacao) AS total\_operacoes,

o.data\_operacao

FROM operacoes o

JOIN itens\_operacao io ON o.id\_operacao = io.id\_operacao

GROUP BY o.tipo\_operacao, o.data\_operacao;

CREATE VIEW vw\_pessoas\_fisicas AS

SELECT

pf.id\_pessoa,

p.nome\_pessoa,

p.email\_pessoa,

pf.cpf,

p.tel1\_pessoa,

p.tel2\_pessoa,

p.endereco\_pessoa

FROM pessoas p

JOIN pessoas\_fisicas pf ON p.id\_pessoa = pf.id\_pessoa;

CREATE VIEW vw\_pessoas\_juridicas AS

SELECT

pj.id\_pessoa,

p.nome\_pessoa,

p.email\_pessoa,

pj.cnpj,

pj.razao\_social,

p.tel1\_pessoa,

p.tel2\_pessoa,

p.endereco\_pessoa

FROM pessoas p

JOIN pessoas\_juridicas pj ON p.id\_pessoa = pj.id\_pessoa;

CREATE VIEW vw\_valor\_gasto\_por\_categoria AS

SELECT

c.id\_categoria,

c.nome\_categoria,

SUM(io.quantidade \* io.preco\_unitario) AS valor\_total\_gasto

FROM

itens\_operacao io

INNER JOIN

produtos p ON io.id\_produto = p.id\_produto

INNER JOIN

categorias c ON p.id\_categoria = c.id\_categoria

INNER JOIN

operacoes o ON io.id\_operacao = o.id\_operacao

WHERE

o.tipo\_operacao = 'V' -- Apenas operações de compra

GROUP BY

c.id\_categoria, c.nome\_categoria;

CREATE VIEW vw\_quantidade\_produtos\_por\_categoria AS

SELECT

c.id\_categoria,

c.nome\_categoria,

COUNT(p.id\_produto) AS quantidade\_produtos

FROM

categorias c

LEFT JOIN

produtos p ON c.id\_categoria = p.id\_categoria

GROUP BY

c.id\_categoria, c.nome\_categoria;

CREATE FUNCTION calcular\_total\_vendas(@inicio DATE, @fim DATE)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @total DECIMAL(10, 2);

SELECT @total = SUM(io.valor\_total)

FROM operacoes o

JOIN itens\_operacao io ON o.id\_operacao = io.id\_operacao

WHERE o.tipo\_operacao = 'V' AND o.data\_operacao BETWEEN @inicio AND @fim;

RETURN @total;

END;

CREATE FUNCTION calcular\_total\_compras(@inicio DATE, @fim DATE)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @total DECIMAL(10, 2);

SELECT @total = SUM(io.valor\_total)

FROM operacoes o

JOIN itens\_operacao io ON o.id\_operacao = io.id\_operacao

WHERE o.tipo\_operacao = 'C' AND o.data\_operacao BETWEEN @inicio AND @fim;

RETURN @total;

END;

CREATE FUNCTION produtos\_mais\_vendidos(@inicio DATE, @fim DATE)

RETURNS TABLE

AS

RETURN (

SELECT

p.id\_produto,

p.nome\_produto,

SUM(io.quantidade) AS total\_vendido

FROM itens\_operacao io

JOIN produtos p ON io.id\_produto = p.id\_produto

JOIN operacoes o ON io.id\_operacao = o.id\_operacao

WHERE o.tipo\_operacao = 'V' AND o.data\_operacao BETWEEN @inicio AND @fim

GROUP BY p.id\_produto, p.nome\_produto

);

## Automatizar algumas transações no banco de dados usando *triggers*.

CREATE TRIGGER trg\_atualizar\_estoque

ON itens\_operacao

AFTER INSERT, UPDATE

AS

BEGIN

DECLARE @id\_produto BIGINT, @quantidade INT, @tipo\_operacao CHAR(1);

SELECT

@id\_produto = i.id\_produto,

@quantidade = i.quantidade,

@tipo\_operacao = o.tipo\_operacao

FROM inserted i

INNER JOIN operacoes o ON i.id\_operacao = o.id\_operacao;

IF @tipo\_operacao = 'C' -- Compra

BEGIN

UPDATE produtos

SET qtd\_estoque = qtd\_estoque + @quantidade

WHERE id\_produto = @id\_produto;

END

ELSE IF @tipo\_operacao = 'V' -- Venda

BEGIN

UPDATE produtos

SET qtd\_estoque = qtd\_estoque - @quantidade

WHERE id\_produto = @id\_produto;

END

END;

CREATE TRIGGER trg\_atualizar\_valor\_total

ON itens\_operacao

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

UPDATE o

SET o.valor\_total = (

SELECT ISNULL(SUM(io.quantidade \* io.preco\_unitario), 0)

FROM itens\_operacao io

WHERE io.id\_operacao = o.id\_operacao

)

FROM operacoes o

WHERE o.id\_operacao IN (

SELECT DISTINCT id\_operacao FROM inserted

UNION

SELECT DISTINCT id\_operacao FROM deleted

);

END;

CREATE TRIGGER trg\_garantir\_estoque\_positivo

ON itens\_operacao

AFTER INSERT, UPDATE

AS

BEGIN

DECLARE @id\_produto BIGINT, @quantidade INT, @tipo\_operacao CHAR(1);

SELECT

@id\_produto = i.id\_produto,

@quantidade = i.quantidade,

@tipo\_operacao = o.tipo\_operacao

FROM inserted i

INNER JOIN operacoes o ON i.id\_operacao = o.id\_operacao;

IF @tipo\_operacao = 'V' -- Venda

BEGIN

IF EXISTS (

SELECT 1

FROM produtos

WHERE id\_produto = @id\_produto AND qtd\_estoque < @quantidade

)

BEGIN

RAISERROR ('Estoque insuficiente para a venda.', 16, 1);

ROLLBACK TRANSACTION;

END

END

END;

CREATE TRIGGER trg\_atualizar\_data\_modificacao\_categorias

ON categorias

AFTER UPDATE

AS

BEGIN

UPDATE categorias

SET data\_modificacao = GETDATE()

WHERE id\_categoria IN (SELECT id\_categoria FROM inserted);

END;

CREATE TRIGGER trg\_atualizar\_data\_modificacao\_produtos

ON produtos

AFTER UPDATE

AS

BEGIN

UPDATE produtos

SET data\_modificacao = GETDATE()

WHERE id\_produto IN (SELECT id\_produto FROM inserted);

END;

CREATE TRIGGER trg\_atualizar\_data\_modificacao\_pessoas

ON pessoas

AFTER UPDATE

AS

BEGIN

UPDATE pessoas

SET data\_modificacao = GETDATE()

WHERE id\_pessoa IN (SELECT id\_pessoa FROM inserted);

END;

CREATE TRIGGER trg\_log\_alteracoes\_categorias

ON categorias

AFTER UPDATE

AS

BEGIN

DECLARE @id\_categoria INT, @nome\_categoria\_old NVARCHAR(100), @nome\_categoria\_new NVARCHAR(100);

SELECT @id\_categoria = i.id\_categoria,

@nome\_categoria\_old = d.nome\_categoria,

@nome\_categoria\_new = i.nome\_categoria

FROM inserted i

INNER JOIN deleted d ON i.id\_categoria = d.id\_categoria;

IF @nome\_categoria\_old <> @nome\_categoria\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('categorias', @id\_categoria, 'nome\_categoria', @nome\_categoria\_old, @nome\_categoria\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_produtos

ON produtos

AFTER UPDATE

AS

BEGIN

DECLARE @id\_produto INT, @nome\_produto\_old NVARCHAR(50), @nome\_produto\_new NVARCHAR(50);

SELECT @id\_produto = i.id\_produto,

@nome\_produto\_old = d.nome\_produto,

@nome\_produto\_new = i.nome\_produto

FROM inserted i

INNER JOIN deleted d ON i.id\_produto = d.id\_produto;

IF @nome\_produto\_old <> @nome\_produto\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('produtos', @id\_produto, 'nome\_produto', @nome\_produto\_old, @nome\_produto\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_pessoas

ON pessoas

AFTER UPDATE

AS

BEGIN

DECLARE @id\_pessoa INT, @nome\_pessoa\_old NVARCHAR(100), @nome\_pessoa\_new NVARCHAR(100);

SELECT @id\_pessoa = i.id\_pessoa,

@nome\_pessoa\_old = d.nome\_pessoa,

@nome\_pessoa\_new = i.nome\_pessoa

FROM inserted i

INNER JOIN deleted d ON i.id\_pessoa = d.id\_pessoa;

IF @nome\_pessoa\_old <> @nome\_pessoa\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('pessoas', @id\_pessoa, 'nome\_pessoa', @nome\_pessoa\_old, @nome\_pessoa\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_pessoas\_fisicas

ON pessoas\_fisicas

AFTER UPDATE

AS

BEGIN

DECLARE @id\_pessoa INT, @cpf\_old NVARCHAR(14), @cpf\_new NVARCHAR(14);

SELECT @id\_pessoa = i.id\_pessoa,

@cpf\_old = d.cpf,

@cpf\_new = i.cpf

FROM inserted i

INNER JOIN deleted d ON i.id\_pessoa = d.id\_pessoa;

IF @cpf\_old <> @cpf\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('pessoas\_fisicas', @id\_pessoa, 'cpf', @cpf\_old, @cpf\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_pessoas\_juridicas

ON pessoas\_juridicas

AFTER UPDATE

AS

BEGIN

DECLARE @id\_pessoa INT, @cnpj\_old NVARCHAR(18), @cnpj\_new NVARCHAR(18);

SELECT @id\_pessoa = i.id\_pessoa,

@cnpj\_old = d.cnpj,

@cnpj\_new = i.cnpj

FROM inserted i

INNER JOIN deleted d ON i.id\_pessoa = d.id\_pessoa;

IF @cnpj\_old <> @cnpj\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('pessoas\_juridicas', @id\_pessoa, 'cnpj', @cnpj\_old, @cnpj\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_operacoes

ON operacoes

AFTER UPDATE

AS

BEGIN

DECLARE @id\_operacao INT, @tipo\_operacao\_old CHAR(1), @tipo\_operacao\_new CHAR(1);

SELECT @id\_operacao = i.id\_operacao,

@tipo\_operacao\_old = d.tipo\_operacao,

@tipo\_operacao\_new = i.tipo\_operacao

FROM inserted i

INNER JOIN deleted d ON i.id\_operacao = d.id\_operacao;

IF @tipo\_operacao\_old <> @tipo\_operacao\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('operacoes', @id\_operacao, 'tipo\_operacao', @tipo\_operacao\_old, @tipo\_operacao\_new);

END

END;

CREATE TRIGGER trg\_log\_alteracoes\_itens\_operacao

ON itens\_operacao

AFTER UPDATE

AS

BEGIN

DECLARE @id\_item\_operacao INT, @quantidade\_old INT, @quantidade\_new INT;

SELECT @id\_item\_operacao = i.id\_item\_operacao,

@quantidade\_old = d.quantidade,

@quantidade\_new = i.quantidade

FROM inserted i

INNER JOIN deleted d ON i.id\_item\_operacao = d.id\_item\_operacao;

IF @quantidade\_old <> @quantidade\_new

BEGIN

INSERT INTO log\_alteracoes (tabela\_alterada, id\_registro, campo\_alterado, valor\_antigo, valor\_novo)

VALUES ('itens\_operacao', @id\_item\_operacao, 'quantidade', CAST(@quantidade\_old AS NVARCHAR(MAX)), CAST(@quantidade\_new AS NVARCHAR(MAX)));

END

END;

## Configurar o banco de dados para criar perfil de usuário e suas permissões para aumentar a segurança ao acesso aos dados, melhorando a segurança do sistema.

CREATE LOGIN Admin WITH PASSWORD = 'Admin2024';

CREATE LOGIN Operacional1 WITH PASSWORD = 'Operacional2024';

CREATE USER Admin FOR LOGIN Admin;

CREATE USER Operacional1 FOR LOGIN Operacional1;

CREATE ROLE Role\_Administrador;

CREATE ROLE Role\_Operacional;

GRANT SELECT, INSERT, UPDATE, DELETE ON pessoas TO Role\_Administrador;

GRANT SELECT, INSERT, UPDATE, DELETE ON produtos TO Role\_Administrador;

GRANT SELECT, INSERT, UPDATE, DELETE ON categorias TO Role\_Administrador;

GRANT SELECT, INSERT, UPDATE, DELETE ON operacoes TO Role\_Administrador;

GRANT SELECT, INSERT, UPDATE, DELETE ON itens\_operacao TO Role\_Administrador;

GRANT EXECUTE ON SCHEMA::dbo TO Role\_Administrador;

-- Para teste de Usuário (Operador1)

-- Deletar (inativar) um registro em pessoas

UPDATE pessoas SET is\_active = 0 WHERE id\_pessoa = 1;

Update pessoas set is\_active = 1 where id\_pessoa = 1;

select \* from pessoas

## Automatizar *backup´s* periódicos para o banco de dados do sistema, garantindo que os dados estarão armazenados e seguros, mas possam ser recuperados, caso seja necessário.

# Etapa 3 – Aplicação de métodos estatísticos para geração de dados e informações numa planilha de Excel

|  |
| --- |
| **Objetivo**: Implementar uma planilha eletrônica que permita a análise de dados e informações de negócios da empresa estudada a partir de técnicas e/ou indicadores estatísticos (medidas de tendência central, regressão, probabilidade, inferência entre outras). |

5.1. Tabelas e Gráficos: ferramentas que ajudam a visualizar a distribuição e a relação entre dados. Dentre as principais representações temos:

* Gráficos de linha: usados para mostrar tendências ao longo do tempo.
* Gráficos de colunas/barras: utilizados para comparar valores entre diferentes categorias
* Gráficos de setores (ou pizza): representam a composição de um todo, representados em partes proporcionais.

5.2. Distribuição de frequências e a representação gráfica de uma distribuição de frequências (histogramas)

5.3. Medidas de Posição: fornecem medidas que ajudam na caracterização e comportamento dos elementos de uma série ou um conjunto de observações. Tais medidas incluem a Média Aritmética Simples, a Mediana e a Moda, que indicam a localização central dos dados.

5.4. Medidas de Variabilidade ou Dispersão: parâmetros estatísticos que quantificam a variação dos valores de um conjunto de dados. A finalidade dessas medidas é avaliar o grau de concentração ou afastamento entre os valores, ou seja, o quanto eles estão dispersos em relação à Média Aritmética Simples. Dentre as principais medidas estão a amplitude total, o desvio-padrão, a variância e o coeficiente de variação de Pearson.

5.5. Análise de Regressão (regressão linear): técnica que estima as relações entre variáveis dependentes e independentes para prever resultados

5.6. Gráficos de Dispersão: Utilizados para exibir a relação entre duas variáveis, plotando pontos em um gráfico. Correlação (R-quadrado): medida que indica a força e a direção da relação (linear) entre duas variáveis.

# Etapa 4 – Estudo da qualidade total dos processos e demandas existentes numa organização

|  |
| --- |
| **Objetivo**: Desenvolver um estudo de qualidade total e melhoria contínua da organização estudada a partir dos levantamentos dos processos de negócio e demandas existentes já detectadas previamente, apontando possíveis soluções e melhorias. |

## Apresentar o negócio, a Missão, a Visão, os valores e os meios de comunicá-los para os colaboradores da empresa.

**Negócio:**  
A Arte Sagrada, representada pelo projeto SIGAS, é uma loja de produtos naturais e artigos religiosos, especializada em oferecer higiene natural e conexão com o Sagrado através de terços e rapés medicinais indígenas. Seu foco é atender clientes e fornecedores com eficiência, garantindo uma gestão ágil de estoques e operações.

**Missão:**  
Proporcionar acesso a produtos naturais e artefatos religiosos de qualidade, promovendo bem-estar, conexão, sustentabilidade e respeito à natureza e às tradições indígenas.

**Visão:**  
Ser reconhecida como referência nacional na comercialização de produtos naturais e religiosos, fortalecendo práticas de despertar da consciência, higiene sustentável e expandindo seu impacto social e ambiental.

**Valores:**

1. Sustentabilidade e respeito ao meio ambiente.
2. Ética e transparência nos negócios.
3. Valorização da cultura indígena e religiosa.
4. Compromisso com a qualidade e a satisfação do cliente.
5. Promoção do bem-estar e da espiritualidade.

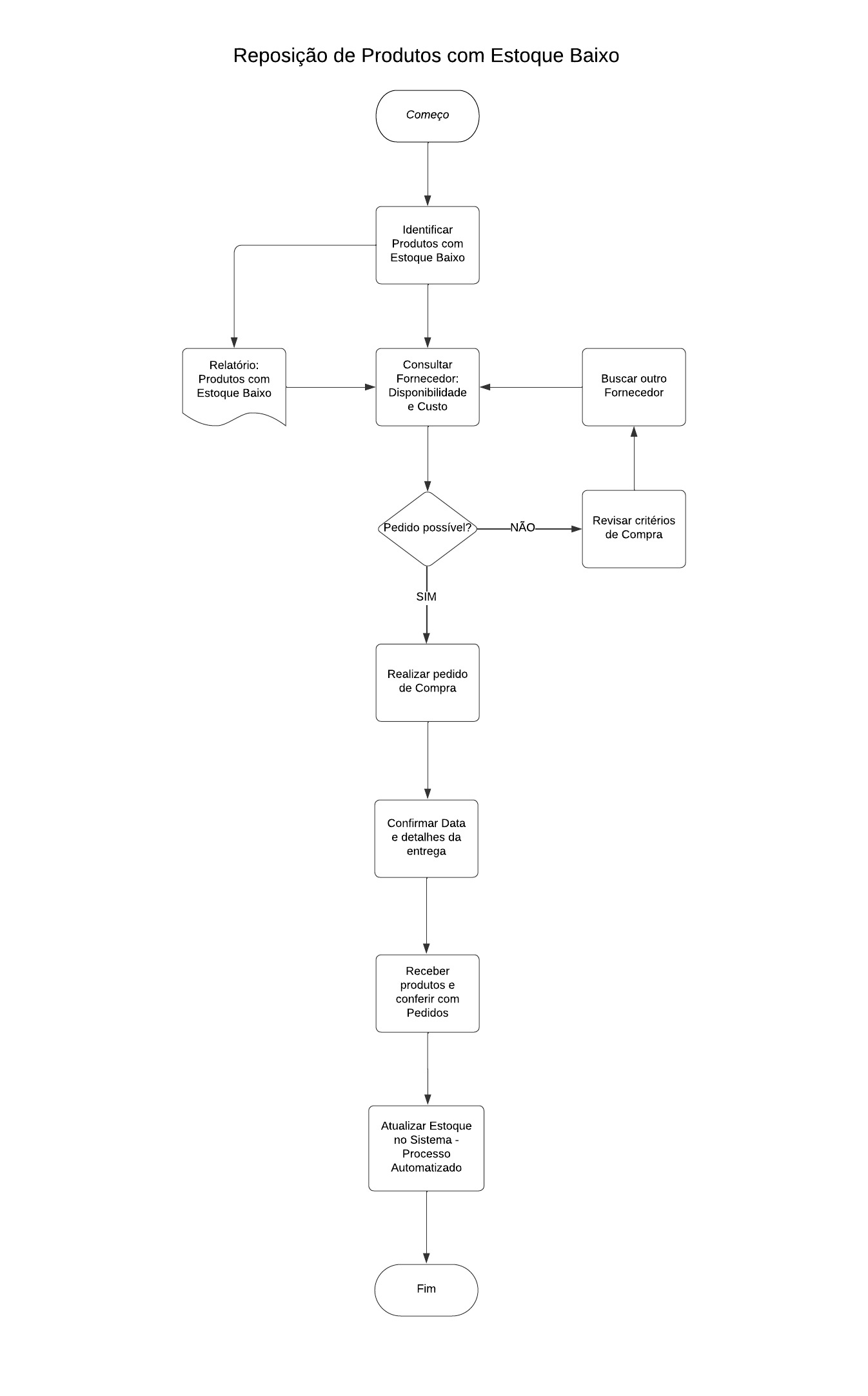
**Meios de comunicação dos valores:**

* **Reuniões regulares:** Encontros semanais para discutir o desempenho e reforçar os valores da empresa.
* **Manuais e treinamentos:** Materiais que explicam a missão, visão e valores, além de capacitar os colaboradores.
* **Quadros de avisos e comunicados internos:** Divulgar os valores e boas práticas em locais visíveis na empresa.
* **Campanhas motivacionais:** Atividades e incentivos que integrem os colaboradores em torno dos objetivos organizacionais.

## Citar um processo que é acompanhado e o(s) indicador(es) utilizado(s), descrevendo a unidade de medida.

|  |  |  |
| --- | --- | --- |
| **PROCESSO** | **INDICADOR** | **UNIDADE DE MEDIDA** |
| Gerenciamento de Estoques | |  | | --- | | Produtos com estoque baixo | | |  | | --- | | Produtos com estoque baixo | |
|  | Tempo de reposição de produtos | Dias úteis |
| Operações de venda | Taxa de vendas finalizadas | Percentual (%) |
|  | Satisfação do cliente | Índice de satisfação (1-5) |

## Elaborar o fluxograma de um processo, considerando p modelo da Ficha de Diagnóstico/Levantamento de áreas oferecido.



**6.4** Sugestão de melhoria no processo, considerando as ferramentas da qualidade – justificar a melhoria através da ferramenta utilizada para propor a mudança.

**Problema identificado:**

A atualização manual do estoque após o recebimento é lenta e propensa a erros.

**Ferramenta da qualidade aplicada:**

**Diagrama de Ishikawa (Causa e Efeito)** foi utilizado para identificar os fatores que contribuem para atrasos na atualização de estoques, como falhas humanas, falta de integração entre sistemas e demora na conferência.

**Solução proposta:**

Automatizar a atualização de estoque através de um sistema integrado que:

Use códigos de barras para registrar a entrada de produtos.

Atualize automaticamente os dados no sistema SIGAS.

Emita alertas para discrepâncias detectadas.

**Justificativa:**  
Essa solução reduzirá o tempo gasto na atualização, minimizará erros humanos e melhorará a precisão das informações do estoque, contribuindo para maior eficiência operacional e satisfação do cliente.

# Conclusão

O projeto SIGAS (Sistema de Gerenciamento Arte Sagrada) proporcionou uma experiência prática e enriquecedora, que contribuiu para o aprendizado técnico e o desenvolvimento de uma solução eficiente para a gestão de informações. Durante a execução, enfrentei desafios relacionados à estruturação do banco de dados, integração de funcionalidades e ajustes nos processos, o que exigiu criatividade e aprimoramento de habilidades técnicas.

O SIGAS mostrou-se uma solução funcional, permitindo o gerenciamento de pessoas, produtos e operações de forma organizada. Funcionalidades geração de relatórios e automação de processos agregam confiabilidade e praticidade ao sistema. Essas características tornam o SIGAS um exemplo real de como ferramentas tecnológicas podem atender a demandas específicas.

Apesar das dificuldades, como erros técnicos e restrições de tempo, o projeto trouxe importantes contribuições acadêmicas e práticas. Ele reforçou a relevância de um planejamento adequado e boa organização no desenvolvimento de sistemas.

Por fim, o SIGAS vai além de um produto técnico, representando uma experiência formativa essencial para futuros desafios profissionais e acadêmicos. Ele serve como base para aprimoramentos e inspiração, demonstrando a aplicabilidade de soluções tecnológicas para resolver problemas de gestão e otimizar processos.