System4Team

Moisés João Ferreira

Yohan Siedschlag

Mark Stolfi

Igor da Silva

Introdução

System4Team é um sistema de gestão empresarial feito pelos alunos: Moisés, Yohan, Igor e Mark para a disciplina de Desenvolvimento de Sistemas. É um sistema robusto desenvolvido para otimizar o gerenciamento de projetos empresariais. Com uma interface intuitiva e funcionalidades abrangentes, o System4Team facilita o planejamento, a execução e o monitoramento financeiro, facilitando o gerenciamento e aumentando a eficiência operacional, ideal para gestores de empresas.

O sistema foi feito com as Linguagens de marcação e estilização HTML e CSS, e com as linguagens de programação Python, com frameworks Django e Flask, Javascript e SQL.

Funcionalidades Principais

- Planejamento de Projetos: Ferramentas para definição de metas, cronogramas, e alocação de recursos de forma eficiente.
- Gestão de Tarefas: Atribuição de tarefas a membros da equipe, com prazos e prioridades claramente definidos, ajudando a manter o foco nas entregas.
- Monitoramento e Relatórios: Painéis de controle que permitem o acompanhamento do progresso dos projetos, além de relatórios detalhados sobre desempenho, tempo e recursos utilizados.

Estudo de Caso

Problema	Solução	Resultado
Qual o problema que você teve que resolver?	**O que você fez para solucionar o problema?**	**Quais resultados obtidos? Números?**
A dificuldade de gerir uma empresa	Organiza em uma estrutura para melhor gestão	Melhor funcionamento de gerenciamento do negócio

Estratégia

Quais foram as estratégias usadas para realizar a solução do problema?
Organização de aspectos empresariais, como clientes, vendas, etc.
Agilidade de Consulta de informações da empresa como lucro, produtos e funcionários
Acesso aos funcionários da empresa, para monitoramento de seu trabalho e produtividade
Conexão com diversos tipos de dispositivos(responsividade)
Fácil uso do sistema
Atulizações recorrentes

Conclusão

Conclua o estudo com os pontos fortes desse aprendizado.
Chegamos á conclusão de que dessa forma foi possível facilitar a gestão de uma empresa, tendo como principal fator a organização e agilidade nos processos mais complexos e trabalhosos

MATRIZ S.W.O.T

System4Team

S Forças

- Negócio em potencial
- Equipe Proativa
- Agilidade no desenvolvimento
- Equipe Organizada
- Qualidade do Produto
- Trabalho Otimizado

W Fraquezas

- Equipe Júnior
- Capacidade Limitada
- Conhecimento Limitado
- Dependência de ferramentas IA
- Escala Limitada

O Oportunidades

- Novos Conhecimentos e tecnologias
- Integração com Redes Sociais
- Expansão Regional
- Desenvolvimento de APPs

T Ameaças

- Riscos de Segurança
- Concorrência Forte
- Erros técnicos
- Regulamentações do mercado

Regras de Negócio

O Acesso do Usuário, Conforme o Cargo na Empresa

RN A

Item	Descrição
Descrição	"O acesso ao sistema será restrito pelo cargo do usuário, conforme registrado no banco de dados. Ao realizar o login, o sistema verificará as credenciais. Se inválidas, exibirá: 'Credenciais inválidas. Contacte o seu superior.' Se válidas, concederá acesso às funcionalidades permitidas pelo cargo e suas permissões."
Exemplo:	“Marcos irá acessar o sistema porém as credenciais estão inválidas, após contatar a sua superiora, ele recebeu suas credenciais de acesso, conseguindo assim acessar o sistema, como o cargo de Marcos é de Estagiário, as funcionalidades disponíveis serão de Estagiário”

Pseudo-Código	FUNC login(usuario, senha):
	  credenciaisDB = buscarUsuarioNoBanco(usuario)
	  SE credenciaisDB E compararSenha(senha, credenciaisDB.senhaHash):
	  cargo = consultarBanco("SELECT cargo FROM usuarios WHERE usuario = ?", usuario).cargo
	  permissoes = consultarBanco("SELECT funcionalidades FROM permissoes WHERE cargo = ?", cargo)
	  RETORNAR "Login bem-sucedido", permissoes
	  RETORNAR "Credenciais inválidas. Contacte o seu superior."
	
	FUNC compararSenha(senha, senhaHash):
	  RETORNAR hash(senha) == senhaHash
	
	// Fluxo Principal
	ENTRADA usuario, senha
	resposta, permissoes = login(usuario, senha)
	
	SE resposta == "Login bem-sucedido":
	  EXIBIR "Bem-vindo! Suas permissões:", permissoes
	SENÃO:
	  EXIBIR resposta

Responsável	Igor

As Funcionalidades de Acordo com o Cargo do usuário

RN A-1

Item	Descrição
Descrição	"Ao realizar o login, o sistema valida as credenciais válidas e sistema verifica o cargo do usuário no banco de dados. As permissões e funcionalidades do sistema são concedidas conforme o cargo: - Usuários com cargos de maior hierarquia, como **CTO** (Raquel), terão acesso total ao sistema. - Usuários com cargos mais restritos, como **Estagiário** (Marcos), terão acesso limitado às funcionalidades específicas atribuídas ao cargo Garantindo que cada usuário tenha acesso apenas às funcionalidades necessárias para seu papel na organização, promovendo segurança e organização no uso do sistema.
Exemplo:	“Raquel acessou o sistema e como o cargo é de CTO da empresa, ela terá total acesso às funcionalidades do sistema. Já Marcos é um estagiário, então as funcionalidades serão de estagiário”
Pseudo-Código	FUNC verificarAcesso(usuario): cargo = consultarBanco("SELECT cargo FROM usuarios WHERE usuario = ?", usuario).cargo SE cargo == "CTO": RETORNAR "Acesso total: Todas as funcionalidades disponíveis." SENÃO SE cargo == "Estagiário": RETORNAR "Acesso limitado: Funcionalidades específicas de estagiário." SENÃO: RETORNAR "Cargo desconhecido: Sem acesso definido." // Fluxo Principal ENTRADA usuario funcionalidade = verificarAcesso(usuario) EXIBIR funcionalidade
Regra relacionada	RN A - O Acesso do Usuário, Conforme o Cargo na Empresa
Responsável	Igor

Definições das funcionalidades dos cargos

RN A-2

Item	Descrição
Descrição	Cada cargo terá um conjunto específico de funções ou permissões. As permissões podem ser baseadas em tarefas como **acessar relatórios ao painel**, **gerenciar clientes**, etc.
Exemplo:	“**CEO**: Acesso total e todas as permissões; **Gerente**: Acesso à relatórios e ao painel, permissão para gerenciar sua equipe e visualizar dados específicos; **CTO:** Acesso a gestão de equipe, e relatórios; **Estagiário:** Acesso restrito a funcionalidades básicas; **Operador:** Acesso a funcionalidades operacionais, como executar processos.”

Pseudo-Código	FUNC login(usuario, senha):
	  SE verificarCredenciais(usuario, senha):
	  cargo = buscarCargoNoBanco(usuario)
	  permissoes = buscarPermissoesPorCargo(cargo)
	  RETORNAR "Login bem-sucedido", permissoes
	  SENÃO:
	  RETORNAR "Credenciais inválidas. Contacte o seu superior."
	
	FUNC verificarCredenciais(usuario, senha):
	  credenciaisDB = buscarUsuarioNoBanco(usuario)
	  RETORNAR credenciaisDB E compararSenha(senha, credenciaisDB.senhaHash)
	
	FUNC buscarCargoNoBanco(usuario):
	  RETORNAR consultarBanco("SELECT cargo FROM usuarios WHERE usuario = ?", usuario).cargo
	
	FUNC buscarPermissoesPorCargo(cargo):
	  RETORNAR consultarBanco("SELECT funcionalidades FROM permissoes WHERE cargo = ?", cargo)
	
	FUNC compararSenha(senha, senhaHash):
	  RETORNAR hash(senha) == senhaHash
	
	ENTRADA usuario
	permissoesUsuario = determinarPermissoes(usuario)
	
	EXIBIR "As permissões para o usuário são:", permissoesUsuario

Regra relacionada	RN A-1 - As Funcionalidades de Acordo com o Cargo do usuário
Responsável	Igor

Gestão de Relatórios

RN B

Item	Descrição
Descrição	Apenas Gerentes de Relatório ou Administradores podem criar ou editar Relatórios. Ao criar um Relatório, deve ser possível definir metas, cronograma, orçamento e membros da equipe.
Exemplo:	"Gerente de Relatório ou Administrador pode criar ou editar Relatórios, atribuindo membros da equipe, cronograma e orçamento."
Pseudo-Código	FUNC criarOuEditarRelatório(Relatório, usuario): SE usuario.cargo = "Gerente de Relatório" OU usuario.cargo = "Administrador": SALVAR Relatório.noBancoDeDados() RETORNAR "Relatório criado/atualizado com sucesso." SENÃO: RETORNAR "Apenas Gerentes de Relatório ou Administradores podem criar ou editar Relatórios."
Responsável	Igor

Controle de Status do Relatório

RN B-1

Item	Descrição
Descrição	Os Relatórios terão status definidos (ex: Em Andamento, Concluído, Suspenso, Cancelado). O status pode ser alterado conforme o progresso do Relatório, e a mudança de status deve ser registrada para fins de auditoria.
Exemplo:	"O status de cada Relatório pode ser alterado conforme o progresso, registrando a mudança para fins de auditoria."
Pseudo-Código	FUNC alterarStatusRelatório(Relatório, novoStatus): SALVAR novoStatus.noBancoDeDados(Relatório) RETORNAR "Status do Relatório alterado com sucesso."
Regra Relacionada	RN B - Gestão de Relatórios
Responsável	Igor

Acompanhamento de Andamento de Vendas

RN B-2

Item	Descrição
Descrição	O sistema deve permitir que os Gerentes de Projeto e Membros de Equipe possam atualizar o status de tarefas e enviar relatórios de progresso. Dependendo do status das tarefas, o sistema gerará alertas para as próximas ações ou para possíveis atrasos.
Exemplo:	Gerentes e Membros da Equipe podem atualizar o progresso das tarefas e gerar alertas caso haja atrasos."
Pseudo-Código	FUNC atualizarProgressoTarefa(tarefa, progresso, usuario): SE usuario.cargo = "Gerente de Projeto" OU usuario.cargo = "Membro da Equipe": SALVAR progresso.noBancoDeDados(tarefa) NOTIFICAR membros sobre a atualização RETORNAR "Progresso atualizado com sucesso."
Regra relacionada	RN B - Gestão de Relatórios
Responsável	Igor

Atribuição de Vendas e Prazos

RN B-3

Item	Descrição
Descrição	O sistema gera relatórios de desempenho por projeto, incluindo tempo gasto, recursos utilizados e tarefas concluídas. Relatórios podem ser exportados em formatos como PDF ou Excel para fácil compartilhamento.
Exemplo:	"Relatórios de desempenho detalhados para cada projeto."
Pseudo-Código	FUNC gerarRelatorioDesempenho(projeto): RELATORIO = CONSULTAR bancoDeDados("SELECT * FROM relatorios WHERE projeto_id = ?", projeto.id) RETORNAR RELATORIO
Regra relacionada	RN B - Gestão de Relatórios
Responsável	Igor

Gestão de Clientes

RN C

Item	Descrição
Descrição	O sistema deve permitir a gestão completa dos clientes, incluindo cadastro, atualização, exclusão e consulta de informações. Relatórios detalhados devem ser gerados por cliente, contemplando o histórico de interações, projetos vinculados e status das demandas. Relatórios podem ser exportados em formatos como PDF ou Excel para facilitar o compartilhamento e análise.
Exemplo:	Relatórios detalhados de interações e status de demandas para cada cliente."
Pseudo-Código	FFUNC cadastrarCliente(nome, email, telefone): INSERIR bancoDeDados("INSERT INTO clientes (nome, email, telefone) VALUES (?, ?, ?)", nome, email, telefone) RETORNAR "Cliente cadastrado com sucesso." FUNC gerarRelatorioCliente(id_cliente): RELATORIO = CONSULTAR bancoDeDados("SELECT * FROM interacoes WHERE cliente_id = ?", id_cliente) EXPORTAR relatorioParaPDF(RELATORIO) RETORNAR "Relatório gerado com sucesso."
Responsável	Igor

Gestão de Produtos Vendidos

RN C - 1

Item	Descrição
Descrição	O sistema deve registrar e gerenciar os produtos vendidos para cada cliente, incluindo detalhes como produto, quantidade, data da venda e valor total. Deve ser possível consultar o histórico de vendas de cada cliente e gerar relatórios consolidados por período ou produto.
Exemplo:	"Histórico detalhado de produtos vendidos para o cliente, com filtros por período e produto., e todos os relatórios"

Pseudo-Código	FUNC registrarVenda(cliente_id, produto_id, quantidade, data, valor_total):
	  INSERIR bancoDeDados("INSERT INTO vendas (cliente_id, produto_id, quantidade, data, valor_total) VALUES (?, ?, ?, ?, ?)", cliente_id, produto_id, quantidade, data, valor_total)
	  RETORNAR "Venda registrada com sucesso."
	
	FUNC gerarRelatorioVendas(cliente_id):
	  RELATORIO = CONSULTAR bancoDeDados("SELECT * FROM vendas WHERE cliente_id = ?", cliente_id)
	  EXPORTAR relatorioParaExcel(RELATORIO)
	  RETORNAR "Relatório de vendas gerado com sucesso."

Responsável	Igor

Requisitos

Requisitos funcionais:

1. A empresa contratante tem total controle das funções que serão implementadas no sistema;
2. O sistema deve permitir o usuário(empresa) a gerir seus funcionários;
3. O sistema deve conter informações sobre vendas, gastos, contratações e fornecimento;
4. O sistema deve acessar cada funcionário tendo controle de pagamentos;
5. O sistema deve conter informações, projetos e tarefas;
6. O sistema deve ter o controle de Frequência e Ponto Eletrônico;
7. O sistema deve apresentar relatórios constantes;
8. O sistema deve gerir arquivos e documentos da empresa contratante.

Requisitos não funcionais:

1. Atualizações frequentes;
2. Proteção e integridade de dados;
3. Compatibilidade;
4. Visual limpo.

Descrição de Cenário

Descrição de Cenário para um Sistema de Gestão Empresarial

Contexto Geral:
O sistema tem como objetivo gerenciar as operações internas de empresas, com foco em painel, vendas, clientes e relatórios gerenciais. Ele deve proporcionar eficiência, segurança e usabilidade, permitindo que gestores e funcionários otimizem processos e tomem decisões baseadas em dados apresentados e organizados pelo sistema.

Elementos Principais do Cenário

1. **Empresas Usuárias do Sistema:**
   O sistema será utilizado por empresas que precisam controlar suas operações de vendas e produtos, como lojas de varejo, distribuidoras ou prestadores de serviços. Cada empresa terá seu próprio cadastro e poderá configurar o sistema conforme suas necessidades.

2. **Usuários do Sistema:**
    a. **Administradores:** Gerenciam a empresa no sistema, incluindo cadastro de produtos, funcionários, vendas e análise de relatórios.
    b. **Funcionários:** Acessam o sistema para registrar vendas, consultar produtos e realizar atividades operacionais.
3. **Produtos:**
    a. Cada produto possui informações como código, descrição, preço de custo, preço de venda, quantidade em estoque e categoria.
    b. O sistema deve rastrear a movimentação de estoque e permitir o registro de compras e vendas.
4. **Vendas:**
    a. Registro das transações, incluindo data, hora, produtos vendidos, quantidade, valor total e vendedor responsável.
    b. Registro do id da empresa e da venda
    c. O sistema gera um comprovante ou nota fiscal para o cliente.
5. **Relatórios:**
    a. Organiza e Expõe em forma de tabela as vendas, com todas suas informações
6. **Sistema de Login:**
    a. Sistema de autenticação para login.
    b. Controle de permissões: nem todos os usuários podem realizar ações administrativas.

Fluxo de Atividades

1. **Cadastro Inicial:**
   O administrador cadastra a empresa, os produtos iniciais e os usuários que irão operar o sistema.
2. **Gerenciamento de Estoque:**
    - Atualização manual ou automática do estoque com base em compras e vendas.
3. **Realização de Vendas:**
    - Funcionário acessa o sistema, seleciona produtos e registra a venda.
    - O sistema gera um comprovante ou nota fiscal para o cliente.
4. **Relatórios Gerenciais:**
    - Relatórios de vendas diárias, semanais e mensais.
    - Produtos mais vendidos e menos vendidos.
    - Desempenho individual dos funcionários.

Desafios a Resolver

- **Escalabilidade:** O sistema deve suportar múltiplas empresas simultaneamente.
- **Segurança:** Proteger dados confidenciais, como preços de custo e dados financeiros.
- **Usabilidade:** Interface simples e funcional para usuários com diferentes níveis de conhecimento técnico.
- **Integrações:** Possibilidade de integrar com ERPs ou sistemas externos para aumentar funcionalidades.

Modelagem do Sistema

Diagrama de Caso de Uso

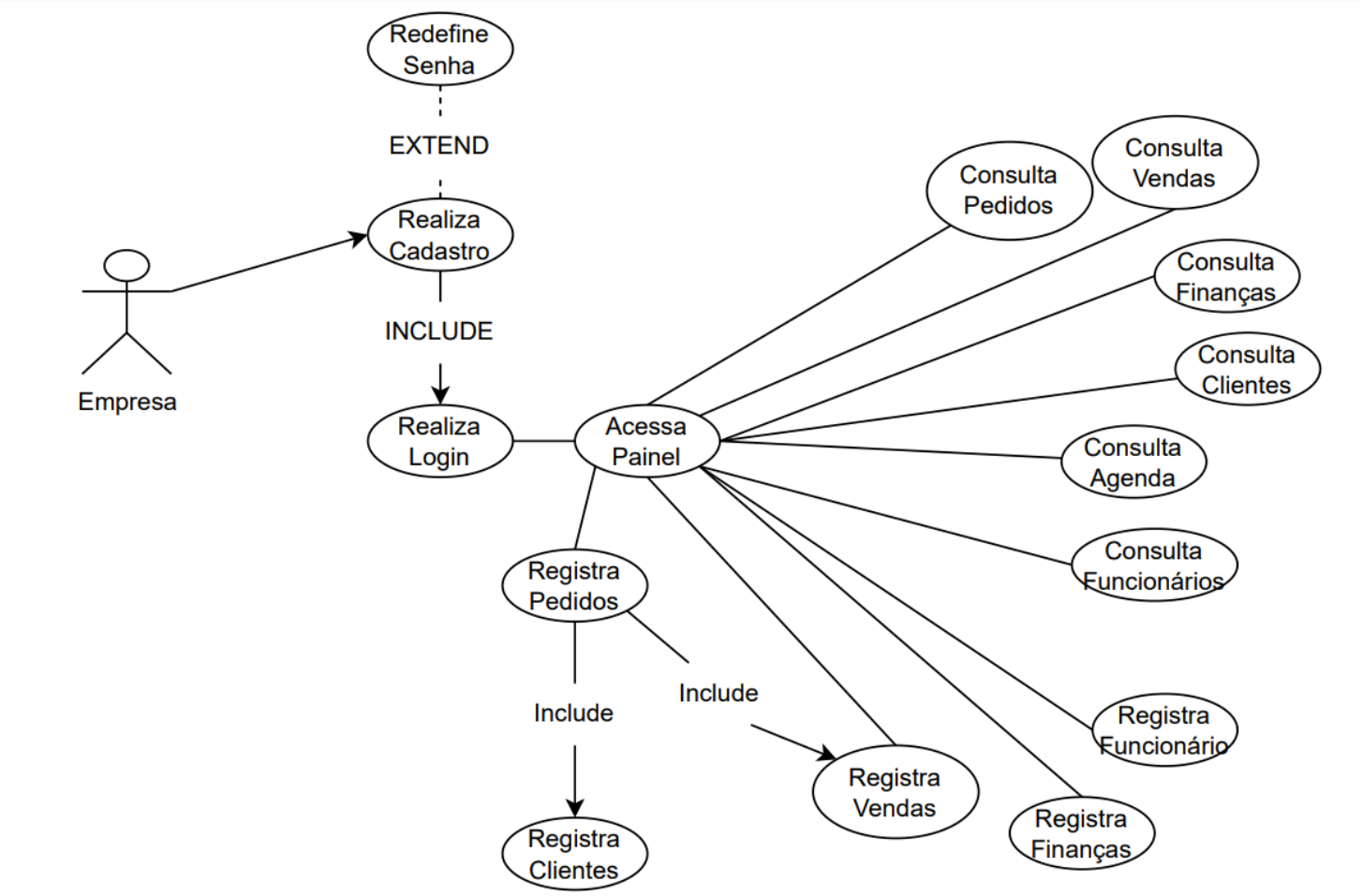
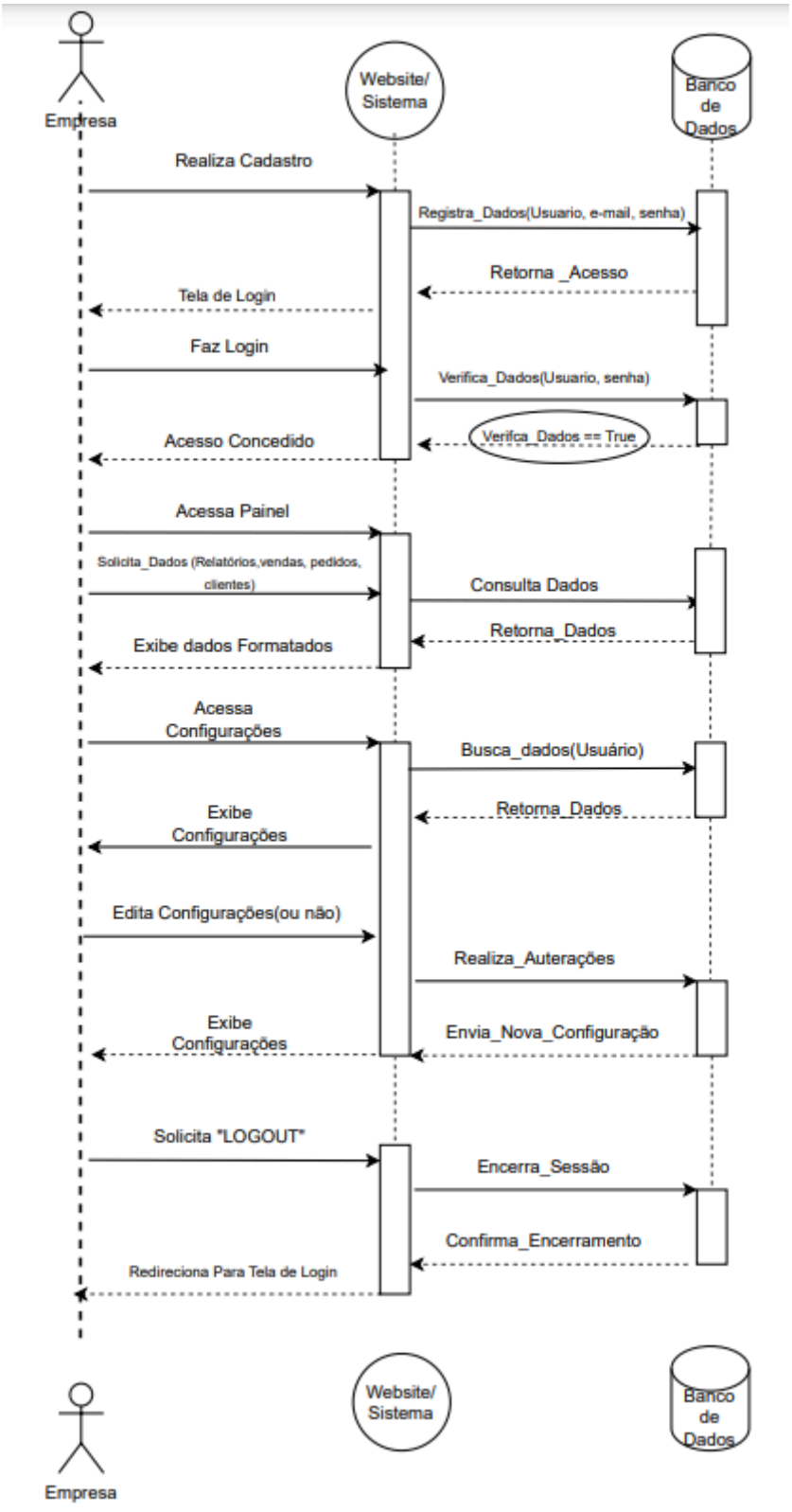


Diagrama de Sequência



Modelagem de Dados

Diagrama Entidade-Relacionamento (Conceitual)

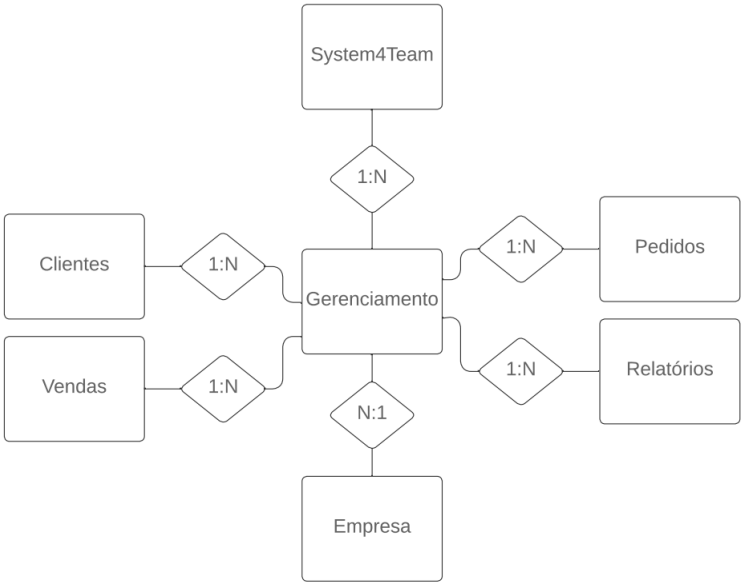
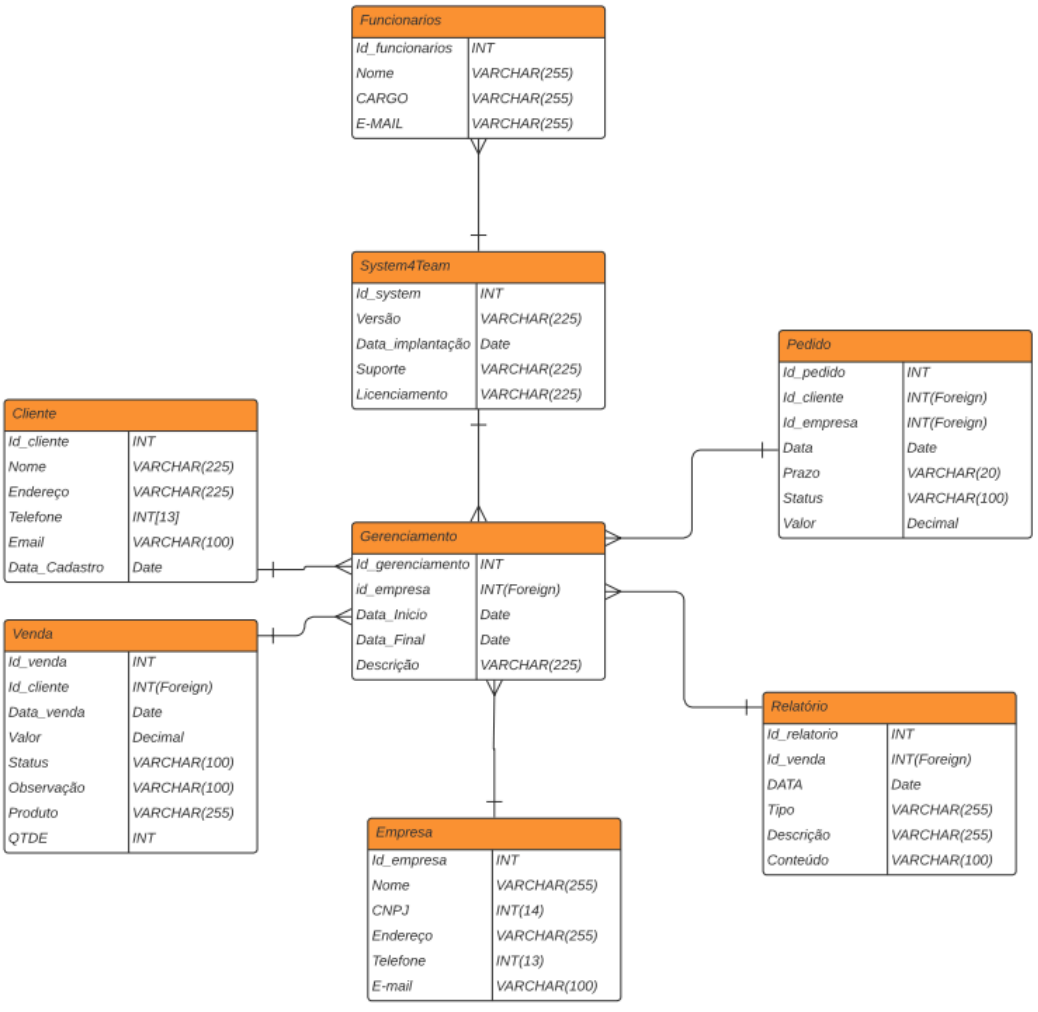


Diagrama Entidade-Relacionamento (Físico)



“

Este documento apresentou uma análise detalhada do sistema proposto, abordando desde o levantamento de requisitos até a modelagem de dados e diagramas.A implementação do sistema trará benefícios significativos, como maior agilidade e produtividade, além da organização. O sistema tem muito a evoluir, como integração com outras plataformas e o desenvolvimento de um APP.

Estamos contentes no nosso trabalho e com essa documentação nosso sistema será implementado com êxito.

”

- Equipe System4team