**UNIVERSIDADE SÃO JUDAS TADEU**

**Natan Fernandes Araujo Ibiapina - 821225920**

**Gabriela Alves Rodrigues - 82311687**

**Graziele Cristina Matsuzaki - 824127569**

**Henryk Bagdanovicius Roza - 823135401**

**MODELOS, MÉTODOS E TÉCNICAS DE ENGENHARIA DE SOFTWARE**

**PROJETO A3 - GESTÃO DE FUNCIONÁRIOS**

**São Paulo**

**2025**

**SUMÁRIO**

[1. INTRODUÇÃO 3](#_Toc200120454)

[2. CONTEXTUALIZAÇÃO DO PROJETO 3](#_Toc200120455)

[2.1. O CLIENTE 3](#_Toc200120456)

[2.1.1. NECESSIDADES DA EMPRESA 3](#_Toc200120457)

[2.1.2. ELICITAÇÃO DE REQUISITOS 4](#_Toc200120458)

[3. REQUISITOS FUNCIONAIS 5](#_Toc200120459)

[4. REQUISITOS NÃO FUNCIONAIS 6](#_Toc200120460)

[5. PRIMEIRA VERSÃO DO BACKLOG DE TAREFAS 8](#_Toc200120461)

[6. ESPECIFICAÇÃO DE CASOS DE USO 10](#_Toc200120462)

[6.2. DIAGRAMA DE CASO DE USO 16](#_Toc200120463)

[7. GESTÃO E PROCESSO DE DESENVOLVIMENTO 17](#_Toc200120464)

[7.1. SPRINTS 18](#_Toc200120465)

[8. MATRIZ DE RASTREABILIDADE 21](#_Toc200120466)

[9. DESIGN E ARQUITETURA DO SISTEMA 23](#_Toc200120467)

[9.1. ARQUITETURA DO SISTEMA 23](#_Toc200120468)

[9.2. CAMADAS DO SISTEMA 23](#_Toc200120469)

[9.3. EXEMPLO DE UM FLUXO DE REQUISIÇÃO PADRÃO MVC 24](#_Toc200120470)

[10. PRINCIPAIS TECNOLOGIAS UTILIZADAS 25](#_Toc200120471)

[11. DEVOPS E DEPLOY 26](#_Toc200120472)

[11.1 REPOSITÓRIO E ESTRUTURA DE PASTAS 26](#_Toc200120473)

[11.2 IMPLEMENTAÇÃO CI/CD 27](#_Toc200120474)

[11.2.1 FUNCIONALIDADES DA PIPELINE CI/CD 27](#_Toc200120475)

[12. PROTOTIPAGEM 28](#_Toc200120476)

[13. CONCLUSÃO 29](#_Toc200120477)

# 1. INTRODUÇÃO

Este documento tem como objetivo apresentar a documentação do Projeto A3 da UC Modelos, Métodos e Técnicas de Engenharia de Software, sendo uma especificação técnica de um sistema de gestão de colaboradores de empresas**.** A proposta é simular o desenvolvimento de um sistema web, que será criado como solução para a necessidade de uma corporação hipotética, abordando práticas de engenharia de software, metodologias ágeis como SCRUM, requisitos funcionais e não funcionais, caso de uso e considerando aspectos como qualidade, desempenho, usabilidade, entre outros.

# 2. CONTEXTUALIZAÇÃO DO PROJETO

O projeto simula o desenvolvimento de um sistema de gestão de colaboradores, com funcionalidades como cadastro, edição, exclusão, consulta dos funcionários da empresa. O sistema também é responsável pelo controle frequência dos funcionários, com a funcionalidade de registro de entrada, saída, almoço e retorno, além da visualização de calendários de dias úteis e feriados.

A aplicação é um software a ser construído na arquitetura MVC utilizando .NET (Framework C#), Entity Framework para a consistência de dados e SQL Server como banco de dados.

## 2.1. O CLIENTE

A TecnoLogix Soluções Ltda. é uma empresa brasileira de médio porte, atuante no setor de Tecnologia da Informação, com sede em São Paulo e algumas de suas equipes remotas atuando pelo país inteiro. Conta com um quadro de aproximadamente 150 colaboradores.

### 2.1.1. NECESSIDADES DA EMPRESA

Através de análises, reuniões e técnicas de elicitação de requisitos foram entendidas e identificadas as principais necessidades e expectativas do cliente com o software a ser desenvolvido. Foram levantadas as seguintes necessidades:

* Gestão de dados dos funcionários: A corporação necessita de um sistema que permita aos administradores cadastrarem e consultar informações detalhadas dos colaboradores, como dados pessoais e administrativos, visando maior controle interno.
* Controle de frequência e jornada de trabalho: A empresa tem uma jornada de trabalho flexível, com equipes remotas em todo país. Para manter a organização, é necessário
* Consulta de horários, folgas e férias: Os colaboradores devem poder acessar com facilidade seus registros de frequência, escalas de trabalho, períodos de folga e férias, promovendo autonomia e redução da carga sobre o RH.
* Funcionalidades para gestores: Os gestores precisam de ferramentas para consultar os dados dos funcionários sob sua supervisão, aprovar ajustes de frequência, acompanhar relatórios de jornada e desempenho, facilitando a tomada de decisões e o planejamento de equipes.

### 2.1.2. ELICITAÇÃO DE REQUISITOS

Para o processo de elicitação de requisitos, foram aplicadas práticas baseadas em técnicas clássicas de elicitação:

* **Análise de Cenário**
  + Foi criado um cenário de estrutura organizacional para a empresa TecnoLogix. Esse cenário serviu como base para simular as dores, necessidades e principais desafios enfrentados na gestão da corporação.
* **Entrevista Simulada**
  + Em simulação, foram feitas entrevistas e reuniões que, através de perguntas e repostas, foi possível extrair os principais requisitos funcionais e não funcionais relevantes para o sistema a ser desenvolvido.

Dessa forma, foi possível definir um conjunto de requisitos funcionais, como: cadastro, edição, exclusão e consulta de colaboradores, envio e visualização de comunicados e registro de frequência. Também foi identificado uma base de requisitos não funcionais, cobrindo principalmente aspectos de segurança, desempenho, usabilidade, disponibilidade, confiabilidade e compatibilidade.

# 3. REQUISITOS FUNCIONAIS

|  |  |  |
| --- | --- | --- |
| ID | Requisito Funcional | Descrição |
| RF001 | Cadastro de colaboradores | O sistema deve permitir que gestores cadastrem novos colaboradores, preenchendo dados obrigatórios como CPF, Nome, cargo, e-mail. |
| RF002 | Edição dos dados dos colaboradores | O sistema deve permitir que gestores editem os dados de colaboradores que estejam previamente cadastrados no sistema. |
| RF003 | Exclusão ou Inativação de colaboradores | O sistema deve permitir que os gestores inativem ou excluam permanentemente um colaborador do sistema. |
| RF004 | Consulta de colaboradores | O sistema deve permitir que os usuários visualizem informações básicas dos colaboradores cadastrados no sistema. |
| RF005 | Login de Usuários | O sistema deve permitir que usuários acessem o sistema através da autenticação de suas credenciais cadastradas no sistema. |
| RF006 | Controle de acessos e permissões | O sistema deve restringir suas funcionalidades de acordo com o nível de permissão do perfil do usuário (Gestor ou funcionário). |
| RF007 | Registro de frequência | O sistema deve permitir que colaboradores registrem sua frequência, sendo horário de entrada e saída. |
| RF008 | Consulta de calendário | O sistema deve permitir que colaboradores consulte um calendário com seus horários e dias úteis no mês, além de feriados previstos e período de férias. |
| RF009 | Envio de comunicados | O sistema deve permitir que gestores enviem comunicados para os colaboradores da empresa. |
| RF010 | Visualização de comunicados | O sistema deve permitir que colaboradores visualizem os comunicados enviados pela gestão. |

# 4. REQUISITOS NÃO FUNCIONAIS

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requisito não funcional | Tipo | Descrição |
| RNF001 | Navegadores WEB | Compatibilidade | O sistema deve poder ser acessado através dos principais navegadores WEB modernos. |
| RNF002 | Suporte a dispositivos móveis | Compatibilidade | O sistema deve poder ser acessado por dispositivos móveis via navegador; |
| RNF003 | Suporte a sistemas operacionais | Compatibilidade | O sistema deve ser acessível através dos principais sistemas operacionais (Windows, Linux, Android, Ios) via navegador. |
| RNF004 | Autenticação obrigatória | Segurança | Todos os usuários devem se autenticar com suas credenciais antes de acessar o sistema. |
| RNF005 | Dados protegidos | Segurança | Os dados dos usuários devem ser protegidos e acessados somente com autorização. |
| RNF006 | Senhas Criptografadas | Segurança | As senhas devem ser armazenadas no banco de dados utilizando criptografias. |
| RNF007 | Fácil utilização | Usabilidade | O sistema deve ser intuitivo e de fácil navegação até mesmo para usuários iniciantes. |
| RNF008 | Interface Acessível | Usabilidade | A interface deve seguir boas práticas de acessibilidade. |
| RNF009 | Interface responsiva | Usabilidade | A interface do sistema deve ser responsiva, e se adequar a diferentes dispositivos móveis. |
| RNF010 | Mensagens intuitivas | Usabilidade | O sistema deve apresentar mensagens claras de validação, erro, sucesso, confirmação, cancelamento entre outros. |

|  |  |  |  |
| --- | --- | --- | --- |
| RNF011 | Disponibilidade mínima | Disponibilidade | O sistema deve estar disponível 99% do tempo durante dias úteis e fora de períodos de manutenção. |
| RNF012 | Recuperação do sistema | Disponibilidade | Em caso de falhas críticas, o sistema deve ser recuperado em até 3 horas. |
| RNF013 | Backup diário | Confiabilidade | O sistema deve realizar backup dos dados pelo menos uma vez ao dia. |
| RNF014 | Análise de Logs | Confiabilidade | A aplicação deve registrar falhas em arquivos de log para análise posterior. |
| RNF015 | Código limpo e bem documentado | Manutenibilidade | O código do sistema deve ser modular, limpo e comentado. |
| RNF016 | Documentação atualizada | Manutenibilidade | O sistema deve possuir uma documentação clara, completa e atualizada. |
| RNF017 | Versionamento | Manutenibilidade | O código e a documentação devem ser ter controle de versionamento através do GitHub. |
| RNF018 | Padrão Arquitetural | Manutenibilidade | O projeto/código deve seguir o padrão arquitetural MVC. |
| RNF019 | Escalabilidade | Desempenho | O sistema deve suportar ao menos 100 usuários simultâneos sem perda significativa de desempenho. |
| RNF020 | Performance de carregamento | Desempenho | As páginas do sistema e pequenas requisições ao banco de dados devem ser carregados em até no máximo 023 segundos. |

# 5. PRIMEIRA VERSÃO DO BACKLOG DE TAREFAS

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Funcionalidade** | **Descrição Resumida** | **Prioridade** |
| 1 | Cadastro de colaboradores | Permite o cadastro de novos colaboradores no sistema. | Alta |
| 2 | Edição de colaboradores | Permite a alteração dos dados dos colaboradores cadastrados. | Alta |
| 3 | Inativação/Exclusão de colaboradores | Gerencia o ciclo de vida dos colaboradores, incluindo a inativação ou exclusão. | Alta |
| 4 | Consulta de colaboradores | Permite visualizar a lista de colaboradores cadastrados. | Alta |
| 5 | Login de usuários | Permite que usuários façam login no sistema de forma segura. | Alta |
| 6 | Controle de acessos e permissões | Gerencia os perfis de acesso dos usuários no sistema. | Alta |
| 7 | Registro de frequência | Permite o registro de horários de entrada, saída e pausas dos colaboradores. | Média |
| 8 | Consulta de calendário | Apresenta um calendário com feriados e períodos de férias. | Média |
| 9 | Envio e visualização de comunicados | Permite à gestão enviar comunicados internos para os colaboradores. | Média |
| 10 | Visualização de comunicados | Permite aos colaboradores visualizar comunicados enviados. | Média |
| 11 | Navegadores WEB | Compatibilidade do sistema com os principais navegadores modernos. | Alta |
| 12 | Suporte a dispositivos móveis | Garantia de funcionamento do sistema em dispositivos móveis. | Alta |
| 13 | Suporte a sistemas operacionais | Compatibilidade do sistema com diferentes sistemas operacionais. | Alta |
| 14 | Autenticação obrigatória | Exige autenticação para acessar as funcionalidades do sistema. | Alta |
| 15 | Dados protegidos | Assegura a integridade e a proteção das informações dos usuários. | Alta |
| 16 | Senhas criptografadas | Armazena as senhas dos usuários de forma criptografada. | Alta |
| 17 | Fácil utilização | Interface desenvolvida com foco na experiência do usuário. | Média |
| 18 | Interface acessível | Sistema com acessibilidade para diferentes públicos. | Média |
| 19 | Interface responsiva | Adaptação do layout para diferentes tamanhos de tela. | Alta |
| 20 | Mensagens intuitivas | Mensagens claras e informativas para facilitar o uso. | Alta |

# 6. ESPECIFICAÇÃO DE CASOS DE USO

#### **6.1.1. Caso de uso 1: Cadastrar colaborador**

Ator: Gestor.

Pré-Condição: O ator deve estar autenticado em um perfil com permissão de cadastro.

Pós condição: O novo colaborador deve ser cadastrado no banco de dados do sistema com suas devidas permissões.

Fluxo principal:

1. O ator seleciona “Cadastrar Colaborador”.
2. O sistema exibe um formulário para preenchimento de dados do formulário.
3. O ator preenche os dados obrigatórios.
4. O ator confirma a operação.
5. O sistema valida os dados.
6. O sistema cadastra os dados do usuário no banco de dados do sistema.

Fluxos alternativos:

3a. Se o usuário não preencheu os dados obrigatórios:

1. O sistema solicita o preenchimento dos dados faltantes.

5a. Se os dados foram preenchidos de forma incorreta (erro de validação):

1. O sistema impede o cadastro no banco de dados e informa o preenchimento incorreto dos dados.

5b. Se existirem dados únicos já cadastrados no sistema (como CPF duplicado)

1. O sistema impede o cadastro no banco de dados e informa a existência de dados duplicados.

#### **6.1.2. Caso de uso 2: Editar dados de Colaborador**

Ator: Gestor.

Pré-Condições: O ator deve estar autenticado em um perfil com permissão para edição e o colaborador alvo da edição deve estar cadastrado no banco de dados.

Pós-Condição: Os dados do colaborador são atualizados no banco de dados.

Fluxo principal:

1. O ator acessa a lista de colaboradores.
2. O ator seleciona o colaborador desejado.
3. O sistema exibe os dados atuais.
4. O ator edita os campos desejados.
5. O ator confirma a operação.
6. O sistema valida os dados.
7. O sistema pede confirmação para a edição dos dados.
8. O ator confirma.
9. O sistema atualiza o cadastro.

Fluxos alternativos:

6a. Se os dados foram preenchidos de forma incorreta (erro de validação):

1. O sistema impede o cadastro no banco de dados e informa o preenchimento incorreto dos dados.

6b. Se existirem dados únicos já cadastrados no sistema (como CPF)

1. O sistema impede o cadastro no banco de dados e informa a existência de dados duplicados.

8a. Se o ator cancela a operação:

1. O sistema retorna à tela anterior sem realizar as alterações.

#### **6.1.3. Caso de uso 3: Excluir Funcionário**

Ator: Gestor.

Pré-Condições: O Ator Administrador deve estar autenticado em um perfil com permissão para edição e o colaborador alvo deve estar cadastrado no banco de dados.

Pós-Condição: O colaborador é inativado ou removido permanentemente do sistema.

Fluxo principal:

1. O ator acessa a lista de colaboradores.
2. O ator seleciona o colaborador a ser excluído.
3. O sistema apresenta duas opções: Inativar funcionário ou excluir permanentemente.
4. O ator escolhe a opção desejada.
5. O sistema pede confirmação.
6. O ator confirma.

O sistema exclui permanentemente ou inativa o funcionário.

Fluxos alternativos:

5a. Se o ator cancela a operação:

1. O sistema retorna à tela anterior sem realizar a exclusão/inativação.

#### **6.1.4. Caso de uso 4: Enviar comunicados aos colaboradores**

Ator: Gestor.

Pré-Condições: O Ator gestor deve estar autenticado em um perfil com permissão para envio de comunicados.

Pós-Condição: O comunicado enviado aos destinatários.

Fluxo principal:

1. O ator acessa a área de comunicados.
2. O ator seleciona “novo comunicado”
3. O sistema exibe um formulário de criação do comunicado.
4. O ator preenche título, mensagem e destinatários.
5. O ator confirma o envio.
6. O sistema envia o comunicado aos destinatários

Fluxos alternativos:

4a. Se um campo estiver em branco

1. O sistema impede o envio do comunicado e informa os campos em branco.

#### **6.1.5. Caso de uso 5: Consultar Funcionário**

Ator: Todos os usuários.

Pré-Condições: O usuário deve estar autenticado.

Pós-Condição: Os dados do funcionário são exibidos.

Fluxo principal:

1. O ator acessa a lista de funcionários.
2. O ator seleciona um funcionário.
3. O sistema exibe os resultados com dados básicos do funcionário selecionado.

#### **6.1.6. Caso de uso 6: Login**

Ator: Todos os usuários.

Pré-Condições: O usuário deve já estar cadastrado no sistema.

Pós-Condição: O usuário é autenticado e acessa o sistema com suas devidas permissões.

Fluxo principal:

1. O ator acessa a tela de login.
2. O ator insere e-mail e senha.
3. O sistema valida as credenciais.
4. O sistema redireciona o usuário para o sistema com funcionalidades de acordo com suas permissões.

Fluxos alternativos:

3a. Se o e-mail ou senha estão incorretos:

1. O sistema informa que há dados preenchidos incorretamente e permanece na página de login.

#### **6.1.7. Caso de uso 7: Registrar frequência**

Ator: colaborador.

Pré-condição: O colaborador está autenticado e tem cadastro ativo.

Pós-condição: O registro de frequência é salvo no sistema.

Fluxo principal:

1. O ator acessa a área de frequência.
2. O ator identifica o tipo de frequência a registrar (entrada, saída, almoço e retorno)
3. O ator registra o tipo de frequência.
4. O sistema registra o horário atual com o tipo de frequência correspondente.

#### **6.1.8. Caso de uso 8: Consultar dias úteis do mês**

Ator: colaborador.

Pré-condição: O colaborador está autenticado e tem cadastro ativo.

Pós-condição: O sistema exibe um calendário de horários e dia úteis no mês.

Fluxo principal:

1. O funcionário acessa a área de Calendário de Dias Úteis.
2. Seleciona o mês desejado.
3. O sistema exibe um calendário destacando:

* Horário de entrada, saída e almoço
* Dias úteis.
* Finais de semana.
* Feriados nacionais/regionais.
* Período de férias do funcionário

#### **6.1.9. Caso de uso 9: Visualizar comunicados da gestão**

Ator: colaborador.

Pré-condição: O colaborador está autenticado e tem cadastro ativo.

Pós-condição: O colaborador visualiza o comunicado enviado pela gestão.

Fluxo principal:

1. O funcionário acessa a área de comunicados.
2. O sistema exibe a lista de comunicados enviados.
3. O funcionário seleciona uma das notificações.
4. O sistema exibe o comunicado e a hora em que foi recebido.

Fluxos alternativos:

3a. Se não houver novos comunicados enviados:

1. O sistema informa que não há novos comunicados.

## 

## 6.2. DIAGRAMA DE CASO DE USO

Diagrama

O conteúdo gerado por IA pode estar incorreto.

# 

# 7. GESTÃO E PROCESSO DE DESENVOLVIMENTO

O projeto está sendo conduzido com base no modelo iterativo e incremental, o que permite desenvolver a aplicação de forma progressiva, realizando entregas parciais ao longo do tempo. Essa abordagem facilita a adaptação do projeto conforme surgem novas necessidades ou ajustes identificados durante o desenvolvimento.

Para a gestão das atividades, está sendo utilizada a metodologia ágil Scrum, com sprints curtas (semanais), definição de backlog, e realização de reuniões de planejamento e acompanhamento diárias. Essa organização tem contribuído para uma melhor divisão de tarefas, acompanhamento constante do andamento e colaboração entre os membros da equipe.

* Modelo iterativo e incremental como base para o desenvolvimento da aplicação.
* Aplicação prática da abordagem ágil Scrum, com uso de sprints, backlog e reuniões periódicas de alinhamento.

Principais referências utilizadas para embasar o processo adotado:

* *Scrum Guide* (2020) – Ken Schwaber e Jeff Sutherland
* *Engenharia de Software* – Ian Sommerville
* *Engenharia de Software* – Roger S. Pressman

**Scrum Master:** Natan Fernandes  
**Product Owner:** Gabriela Alves  
**Desenvolvedores:** Graziele Cristina Matsuzaki, Henryk Bagdanovicius Roza

O projeto foi dividido em cinco sprints, cada uma com escopo claramente definido e entregas incrementais, de acordo com os princípios da metodologia Scrum. A seguir, descrevem-se os objetivos, requisitos abordados e resultados obtidos em cada ciclo de desenvolvimento.

# 7.1. SPRINTS

**Sprint 1 – Fundamentos e Autenticação**

Esta sprint teve como objetivo estabelecer a base do sistema com foco na segurança de acesso e estruturação dos perfis de usuário. Foram implementadas as funcionalidades de autenticação e controle de permissões, bem como os ajustes de compatibilidade e responsividade da interface.

**Requisitos contemplados:**

* RF005 – Login de usuários
* RF006 – Controle de acessos e permissões
* RNF004 – Autenticação obrigatória
* RNF006 – Senhas criptografadas
* RNF001 – Navegadores WEB
* RNF009 – Interface responsiva

**Principais entregas:**

* Módulo de login funcional com criptografia de senhas;
* Controle de acesso conforme o perfil do usuário;
* Compatibilidade testada com navegadores modernos;
* Interface adaptada para diferentes tamanhos de tela.

**Sprint 2 – Gestão de Colaboradore**s

O foco da segunda sprint foi a implementação das funcionalidades essenciais relacionadas à gestão de colaboradores, incluindo cadastro, edição e consulta. Também foram introduzidas melhorias relacionadas à usabilidade, validações de dados e segurança das informações manipuladas.

**Requisitos contemplados:**

* RF001 – Cadastro de colaboradores
* RF002 – Edição de dados de colaboradores
* RF004 – Consulta de colaboradores
* RNF007 – Fácil utilização
* RNF010 – Mensagens intuitivas
* RNF005 – Dados protegidos

**Principais entregas:**

* Formulário completo de cadastro com validações;
* Tela de edição com campos dinâmicos e regras de negócios;
* Listagem de colaboradores com filtros e visualização organizada;
* Mensagens claras de sucesso, erro e confirmação;
* Proteção e integridade dos dados implementada no backend.

**Sprint 3 – Controle e Ciclo de Vida**

A terceira sprint foi dedicada à implementação do gerenciamento do ciclo de vida dos colaboradores, incluindo a possibilidade de inativação ou exclusão de registros. Além disso, foram realizados testes de compatibilidade com diferentes dispositivos e sistemas operacionais, visando acessibilidade ampla.

**Requisitos contemplados:**

* RF003 – Inativação/Exclusão de colaboradores
* RNF002 – Suporte a dispositivos móveis
* RNF003 – Suporte a sistemas operacionais
* RNF008 – Interface acessível

**Principais entregas:**

* Funcionalidade de inativação com confirmação do usuário;
* Remoção lógica ou física com registro de auditoria;
* Testes de interface em Android, iOS, Windows e Linux;
* Interface revisada para atender critérios de acessibilidade.

**Sprint 4 – Frequência e Calendário**

A quarta sprint abordou funcionalidades operacionais de rotina dos colaboradores. Foi desenvolvido o módulo de controle de frequência (entrada, saída e pausas) e o calendário com visualização de feriados, dias úteis e períodos de férias. A responsividade foi validada em cenários reais.

**Requisitos contemplados:**

* RF007 – Registro de frequência
* RF008 – Consulta de calendário
* RNF007 – Fácil utilização (complementar)
* RNF009 – Interface responsiva (validação em cenário prático)

**Principais entregas:**

* Registro de horários com controle por dia e usuário;
* Integração de calendário com dados do colaborador;
* Testes de usabilidade com feedback de usuários-teste;
* Confirmação de responsividade em tablets e smartphones.

**Sprint 5 – Comunicação Interna**

A última sprint teve como foco o desenvolvimento da comunicação institucional entre gestão e colaboradores. Foram implementadas as funcionalidades de envio e leitura de comunicados, com reforço nas mensagens intuitivas e proteção final dos dados.

**Requisitos contemplados:**

* RF009 – Envio de comunicados
* RF010 – Visualização de comunicados
* RNF010 – Mensagens intuitivas (validação completa)
* RNF005 – Dados protegidos (verificação final)

**Principais entregas:**

* Editor de comunicados com campos de título, descrição e data;
* Tela de leitura com marcação de visualização;
* Verificação final das mensagens exibidas ao usuário;
* Proteção de dados reforçada para evitar acessos indevidos.

# 8. MATRIZ DE RASTREABILIDADE

Para assegurar a entrega de um produto de alta qualidade e totalmente alinhado aos objetivos do projeto, foi elaborada a Matriz de Rastreabilidade. Esta ferramenta serve como um mapa de referência que conecta a origem de cada requisito com sua implementação e validação final, garantindo que nenhuma funcionalidade solicitada seja omitida ou mal interpretada durante o ciclo de desenvolvimento.

Neste projeto, a matriz cria uma ligação bidirecional entre os Requisitos Funcionais (RF) e Não Funcionais (RNF) levantados, os Casos de Uso que detalham a interação do usuário com o sistema e as Sprints em que foram desenvolvidos, conforme a metodologia Scrum adotada.

A utilização desta matriz oferece os seguintes benefícios:

* + Visibilidade Completa: Permite que o Product Owner, o Scrum Master e os desenvolvedores visualizem o progresso e a cobertura de cada requisito de forma clara e objetiva.
  + Gestão de Mudanças: Facilita a análise de impacto, pois ao alterar um requisito, é possível identificar rapidamente todos os artefatos de desenvolvimento e teste associados.
  + Garantia de Qualidade: Assegura que para cada requisito funcional existe um caso de uso correspondente e uma entrega planejada, e que todos foram contemplados ao longo das Sprints do projeto.
  + A tabela a seguir apresenta a matriz de rastreabilidade do sistema de Gestão de Funcionários.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID do Requisito** | **Descrição do Requisito** | **Caso de Uso Correspondente** | **Sprint de Implementação** |
| RF001 | Cadastro de colaboradores. | UC01: Cadastrar colaborador. | Sprint 2 |
| RF002 | Edição dos dados dos colaboradores. | UC02: Editar dados de Colaborador. | Sprint 2 |
| RF003 | Exclusão ou Inativação de colaboradores. | UC03: Excluir Funcionário. | Sprint 3 |
| RF004 | Consulta de colaboradores. | UC05: Consultar Funcionário. | Sprint 2 |
| RF005 | Login de Usuários. | UC06: Login. | Sprint 1 |
| RF006 | Controle de acessos e permissões. | UC06: Login (Pós-condição). | Sprint 1 |
| RF007 | Registro de frequência. | UC07: Registrar frequência. | Sprint 4 |
| RF008 | Consulta de calendário. | UC08: Consultar dias úteis do mês. | Sprint 4 |
| RF009 | Envio de comunicados. | UC04: Enviar comunicados aos colaboradores. | Sprint 5 |
| RF010 | Visualização de comunicados. | UC09: Visualizar comunicados da gestão. | Sprint 5 |
| RNF001 | Compatibilidade com Navegadores WEB. | N/A (Aplicável a todas as interfaces) | Sprint 1 |
| RNF002 | Suporte a dispositivos móveis. | N/A (Aplicável a todas as interfaces) | Sprint 3 |
| RNF003 | Suporte a sistemas operacionais. | N/A (Aplicável a todas as interfaces) | Sprint 3 |
| RNF004 | Autenticação obrigatória. | UC06: Login. | Sprint 1 |
| RNF005 | Dados protegidos. | N/A (Aplicável a todas as funcionalidades) | Sprint 2 e Sprint 5 |
| RNF006 | Senhas Criptografadas. | UC06: Login. | Sprint 1 |
| RNF007 | Fácil utilização. | N/A (Aplicável a todas as interfaces) | Sprint 2 e Sprint 4 |
| RNF008 | Interface Acessível. | N/A (Aplicável a todas as interfaces) | Sprint 3 |
| RNF009 | Interface responsiva. | N/A (Aplicável a todas as interfaces) | Sprint 1 e Sprint 4 |
| RNF010 | Mensagens intuitivas. | N/A (Aplicável a todas as interfaces) | Sprint 2 e Sprint 5 |

# 9. DESIGN E ARQUITETURA DO SISTEMA

## 9.1. ARQUITETURA DO SISTEMA

O sistema será estruturado utilizando o padrão MVC (Model-View-Controller) que separa as responsabilidades do sistema em três camadas principais:

* **Model:** Apresenta a lógica do sistema, regras de negócio e acesso aos dados.
* **View:** É a camada responsável pela interface interativa para o usuário (UI).
* **Controller:** Intermedia a comunicação entre Model e View.

O MVC tem como objetivo separar a lógica de negócios, a interface de usuário e o controle das requisições, promovendo organização, reutilização de código e facilidade na manutenção do sistema. É uma arquitetura robusta que tem como vantagens:

* **Facilidade de manutenção e testes:** Como cada camada é independente, modificar uma parte do sistema não causará impactos diretos à outros módulos, facilitando testes unitários e manutenção futura.
* **Escalabilidade:** O uso de MVC facilita a evolução do sistema, permitindo a adição de novas funcionalidades de forma organizada e modular, por conta da idepêndencia das camadas.
* **Alinhamento com as tecnologias escolhidas:** O projeto será desenvolvido utilizando o framework .NET (C#), que oferece suporte nativo ao padrão MVC, tornando seu desenvolvimento mais eficiente.

## 9.2. CAMADAS DO SISTEMA

#### **9.2.1. MODEL**

Responsável por representar as entidades do sistema, sua lógica e regras de negócio. Também interage com a base de dados. Exemplos de algumas entidades do sistema:

* Colaborador
* Gestor (Herança de colaborador)
* Comunicados

Entre suas principais responsabilidades, a validação de dados (validar campos como CPF e campos obrigatórios), estruturar a lógica das regras de negócio e interações com o banco de dados.

#### **9.2.2. VIEW**

Responsável por apresentar a interface (UI) ao usuário. Sua principal responsabilidade é a renderização das páginas dinâmicas do sistema (Formulários, comunicados, calendários) com interfaces distintas de acordo com a lógica do sistema e nível de acesso do usuário, através de tecnologias como HTML, CSS, JAVASCRIPT e outros frameworks e bibliotecas front-end.

#### **9.2.3. CONTROLLER**

Responsável por tratar as ações do usuário, assim como as requisições HTTP para retornar a resposta adequada para que a view seja exibida ao usuário de forma correta e dinâmica. Também realiza autenticações, além de acionar as regras de negócio no sistema e intermediar a comunicação entre as Views e as Models. Alguns exemplos de

* ColaboradorController
* FrequênciaController
* ComunicadoController
* CalendarioController

## 9.3. EXEMPLO DE UM FLUXO DE REQUISIÇÃO PADRÃO MVC

1. O usuário preenche um formulário e clica em “Cadastrar”.
2. A View envia os dados via HTTP para o Controller responsável.
3. O Controller valida permissões e repassa os dados para validação.
4. Após validação, os dados são enviados ao banco de dados.
5. O Controller redireciona para uma nova View (mensagem de sucesso ou erro).
6. A View exibe o resultado da operação ao usuário.

# 10. PRINCIPAIS TECNOLOGIAS UTILIZADAS

**.NET 8, ASP.NET CORE Com Razor Pages**

Plataforma de desenvolvimento da Microsoft para construção de aplicações de diversos tipos, no caso do projeto, será utilizada como base para a aplicação WEB, fornecendo recursos necessários para seu desenvolvimento, ASP.NET Core com Razor Pages (Páginas (Views) dinâmicas desenvolvidas com C# e HTML unificados).

**Entity Framework Core**

Framework de ORM (Object-Relational Mapping), tem o intuito de facilitar a comunicação entre a aplicação e o banco de dados, no caso de consultas, cadastros entre outros.

**Bootstrap**

Framework CSS para facilitar e unificar o desenvolvimento de páginas WEB com design limpo e responsivo.

**SQL Server**

Gerenciador de banco de dados relacionais da Microsoft, será utilizado para o armazenamento de dados do sistema, como os dados dos colaboradores cadastrados no sistema.

**Git e GitHub**

Tecnologia para a hospedagem e versionamento do código fonte do projeto.

**Gitlab CI/CD**

Integração e entrega contínua CI/CD para a automatização dos processos de build, teste e deploy do sistema quando forem detectadas alterações no repositório principal do sistema.

# 11. DEVOPS E DEPLOY

## 11.1 REPOSITÓRIO E ESTRUTURA DE PASTAS

**TECNOLOGIX\_GESTAO**

* **/Controllers**
* FuncionariosController.cs
* FrequenciaController.cs
* ComunicadosController.cs
* LoginController.cs
* **/Models**
* Funcionario.cs
* RegistroFrequencia.cs
* Comunicado.cs
* Usuario.cs
* **/Views**
* Index.cshtml
* CreateFuncionario.cshtml
* UpdateFuncionario.cshtml
* DeleteFuncionario.cshtml
* RegistrarFrequencia.cshtml
* Comunicados.cshtml
* **/Wwwroot** (Arquivos estáticos, CSS, JavaScript e imagens)
* **/css**
* **/js**
* **/img**
* **/Data** (Contexto de banco de dados)
* DbContext.cs
* **/Services**
* **/Workflows**
* ci-cd.yml (Pipeline de CI/CD com GitLab)
* **/Docs**

.gitignore

appsettings.json

Program.cs

Startup.cs

README.md

TECNOLOGIX\_GESTAO.sln

## 11.2 IMPLEMENTAÇÃO CI/CD

A integração contínua (CI) e o deploy contínuo (CD) são práticas de DevOps que automatizam o processo de compilar, testar e publicar o software.

### 11.2.1 FUNCIONALIDADES DA PIPELINE CI/CD

**Integração Contínua (CI)**

Quando um novo código é enviado para o repositório, é esperado que a pipeline execute as seguintes etapas:

* + Build do projeto
  + Execução de testes
  + Validação do código

**Deploy Contínuo (CD)**

Após validado e toda essa etapa de CI aprovada, a aplicação é preparada para realizar automaticamente o Deploy, com a publicação da aplicação.

# 12. PROTOTIPAGEM

Para garantir uma experiência de usuário (UX) alinhada aos requisitos de usabilidade e acessibilidade do projeto, a interface do sistema foi idealizada e validada através de protótipos de alta fidelidade desenvolvidos na plataforma Figma. Essa abordagem permitiu a visualização e o teste dos fluxos de navegação e da disposição dos elementos visuais antes da fase de codificação. O uso do Figma foi essencial para refinar o design e assegurar que a implementação final fosse intuitiva e responsiva, servindo como um guia claro para a equipe de desenvolvimento.

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Prototipagem -Tela de login

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Prototipagem – Menu Principal

**Link da prototipagem:** https://www.figma.com/proto/d7eZC0x6ZJCeZokFdYXJP4/Sistema-de-Gerenciamento?node-id=0-1&t=jbj1Tmi7O3is4fSp-1

# 13. CONCLUSÃO

O projeto A3 desenvolvido esse semestre teve como objetivo simular a elaboração da especificação técnica de um sistema web para gestão de colaboradores, proposto para atender às necessidades da empresa fictícia TecnoLogix Soluções Ltda. Através da definição dos requisitos funcionais e não funcionais, arquitetura baseada no padrão MVC com .NET, planejamento de backlog e sprints ágeis e estratégias de DevOps, foi possível estruturar uma possível solução teórica coerente com as necessidades da organização.

Mesmo sendo um projeto teórico (com empresa e software hipotéticos), todas as decisões foram tomadas com base em possíveis cenários das necessidades de uma empresa no perfil criado, analisando e considerando aspectos de gestão, qualidade e engenharia de software.