Especificação dos Requisitos do Software

Agenda Online

G3X 1.0

{ Versão revisada 4 }

Autores:

José Vitor de Sousa Feitosa
Júlia Gabriela Sobral Santos
Maria Monalisa Andrade Macedo Moura

Angical – PI

18–10–2025

Sumário

1 Introdução

1.1 Objetivo

Desenvolver uma aplicação web para gerenciamento de tarefas pessoais, com foco em agenda, notificações e histórico. A aplicação deve ser intuitiva, segura e responsiva.

1.2 Escopo

O sistema **Chronos** tem como escopo o desenvolvimento de uma aplicação web voltada para o **gerenciamento de tarefas pessoais**, permitindo que o usuário organize suas atividades diárias de forma prática e segura.

Dentro do escopo (funcionalidades contempladas):

- Cadastro e login de usuários, com autenticação em duas etapas (2FA);
- Adição, edição, exclusão e listagem de tarefas;
- Associação de tipo, data e horário às tarefas (com cores distintas por categoria);
- Visualização de tarefas em formato de calendário mensal;
- Sistema de alarmes e notificações para lembretes de tarefas;
- Histórico de tarefas concluídas, com opção de marcação de conclusão;
- Logout e controle de sessão do usuário.

Fora do escopo (funcionalidades não contempladas):

- Integração com e-mails externos (exceto o envio de códigos de autenticação 2FA);
- Compartilhamento de tarefas entre múltiplos usuários (funcionalidades colaborativas);
- Exportação de dados para outros formatos (como PDF, CSV ou Excel);
- Desenvolvimento de aplicativo móvel nativo;
- Integração com assistentes virtuais ou sistemas externos de agenda.

Assim, o sistema limita-se ao uso individual, localizando-se como uma ferramenta pessoal de produtividade, sem integração direta com plataformas de terceiros ou funcionalidades colaborativas.

1.3 Glossário

- **Modal:** Janela flutuante sobre a interface principal, usada para exibir formulários ou mensagens sem sair da tela atual.
- **CRUD:** Acrônimo para as operações básicas de banco de dados: Create (criar), Read (ler), Update (atualizar) e Delete (excluir).

- **JWT (JSON Web Token):** Padrão aberto utilizado para autenticação de usuários e troca segura de informações entre cliente e servidor.
- **Front-end:** Camada visual do sistema, responsável pela interação do usuário com a aplicação.
- **Back-end:** Camada lógica e de processamento do sistema, responsável por armazenar dados e executar regras de negócio.

2 Requisitos Funcionais

2.1 Gestão de Usuários

- **RF01:** Tela de login. Validação de credenciais no back-end.
- **RF02:** Se a conta não existir, opção de cadastro. Senha (mín. 8 caracteres, com hash bcrypt). Envio de e-mail de confirmação e ativação de 2FA.
- **RF03:** Após login bem-sucedido, redirecionar para tela principal.

2.2 Tela Principal

- **RF04:** Barra horizontal superior com ícones: Calendário (visualização mensal), Alarme (lista de notificações), Histórico (tarefas concluídas) e botão "+"para adicionar tarefas.
- **RF05:** Adicionar tarefa via partição (modal expansível): Campos obrigatórios (nome, tipo com cor associada, data, horário). Validações:Data futura, sem conflitos de horário. Armazenar no BD.
- **RF06:** Listagem de tarefas pendentes na tela principal, filtradas por data/horário atual. Suporte a edição (atualizar campos) e exclusão (confirmação via modal).

2.3 Funcionalidades Adicionais

- **RF07:** Calendário mensal: Visualização em grade (biblioteca como react-calendar). Clique em dia e exiba mini-bloco com tarefas desse dia (nome, tipo, horário).
- **RF08:** Alarme: Verificação automática de tarefas próximas (1 dia antes e 1 hora antes do horário). Exibir notificação push no navegador (usando Notification API) e badge no ícone de alarme.
- **RF09:** Histórico: Lista paginada de tarefas concluídas (marcar como concluída via checkbox na listagem). Filtros por data/tipo. Tarefas pendentes permanecem na listagem principal.
- **RF10:** Logout: Encerra sessão e redireciona para tela de login.

2.4 Fluxo Básico do Usuário

- Usuário acessa a página inicial (login/cadastro).
- Se novo, cadastre-se (validação e 2FA).
- Login → Validação → Acessa o painel principal.
- No painel: Adicionar/lista/edita/exclui tarefas; visualiza calendário/alarmes/histórico.
- Logout para encerrar a sessão.
- Tratamento de erros: Mensagens amigáveis (ex: "E-mail inválido") e retry options.

3 Requisitos Não Funcionais

3.1 Usabilidade

- Interface simples e intuitiva (UX/UI seguindo princípios de Material Design). Tempo médio para adicionar tarefa: < 30s. Suporte a acessibilidade (WCAG 2.1 AA: alt texts, navegação por teclado).

3.2 Segurança

- **Senhas:** Armazenadas hashed (bcrypt) no BD. HTTPS obrigatório. Proteção contra brute-force (rate limiting: 5 tentativas/5min). Apenas usuários logados acessam dados (JWT tokens para autenticação). 2FA via e-mail (biblioteca como pyotp no back-end).

3.3 Desempenho e Compatibilidade

- **Tempo de resposta:** < 2s para operações CRUD. Responsivo para mobile/desktop (media queries no CSS).
- **Navegadores:** Chrome 90+, Firefox 88+, Edge 90+. Sem suporte a IE.

3.4 Manutenibilidade e Escalabilidade

- Código modular (ex: componentes React reutilizáveis). Documentação inline (JSDoc). Cobertura de testes: $> 80\%$ (unitários e integração).
- **Escalabilidade:** Suporte a 100 usuários simultâneos iniciais (otimizar queries SQL).

3.5 Outros

- **Idiomas:** Português (BR) como padrão.
- **Métricas de Sucesso:** Sucesso: Taxa de retenção de usuários $> 70\%$ após 1 mês (via analytics opcionais).

4 Arquitetura Técnica

4.1 Front-End

- **Framework:** React (com hooks e context para estado global).
- **Estilos:** TailwindCSS v4 + ShadCN (priorizar sobre Material UI para evitar conflitos; usar MUI apenas para componentes específicos como DatePicker).
- **Linguagem:** TypeScript (tipagem estrita para props e APIs).
- **Build:** Vite (para desenvolvimento rápido e hot-reload).
- **Outros:** Axios para chamadas HTTP; react-router para navegação interna (SPA).

4.2 Back-End

- **Framework:** Django (Python) para API RESTful (CRUD de tarefas e autenticação).
- **Autenticação:** Django REST Framework com JWT; 2FA via e-mail (usando django-otp). Armazenamento de usuários e autenticação totalmente integrados ao Django ORM.
- **Banco de Dados:** PostgreSQL como banco único (relacional) para todas as entidades, incluindo usuários e tarefas. Migrações e gerenciamento via Django ORM (makemigrations/migrate).
- **Linguagem:** Python (Django); JavaScript opcional para scripts utilitários (ex: validações client-side, mas não no core back-end).
- **Outros:** CORS para integração front-back; Docker para conteinerização e ambiente padronizado de desenvolvimento e deploy.

4.3 Integração

- **API Endpoints:** /auth/login, /auth/register, /tasks/ (GET/POST/PUT/DELETE), /calendar/, /alarms/, /history/.
- **Deploy:** Front em Vercel/Netlify; Back em Heroku/Railway com Postgres gerenciado (ex: Supabase).

5 Risco e Prioridades

5.1 Riscos

- Falhas de autenticação ou envio de 2FA por e-mail - Mitigação: teste unitários e ambiente de e-mail configurado com fallback.
- Erros de migração no banco de dados (PostgreSQL) – Mitigação: uso de migrations controladas pelo Django ORM e backups automáticos.

- Problemas de CORS na integração front-and/back-and – Mitigação: configuração adequada dos domínios permitidos no Django e no front-and.
- Dependência de serviços externos (ex: deploy no Heroku/Railway, e-mail SMTP) – Mitigação: definir alternativas de contingência (ex: outro provedor de deploy).

5.2 Prioridades (MoSCoW)

- **Must:** Login, adicionar tarefas.
- **Should:** Calendário, alarme.
- **Could:** Filtros avançados.
- **Won't:** App mobile nativo.

5.3 UC01: Cadastro de Novo Usuário

- **Atores:** Usuário Anônimo.
- **Pré-condições:** Usuário acessou a tela inicial e não possui conta.
- **Fluxo Principal:**
  - Usuário clica em "Criar Conta".
  - Sistema exibe formulário: nome completo, e-mail, senha (confirmação).
  - Usuário submete; sistema valida (e-mail único, senha forte) e armazena hashed no PostgreSQL.
  - O sistema envia e-mail de confirmação com link de ativação.
  - Usuário clica no link; sistema ativa 2FA (código por e-mail).
  - Usuário insere código; conta é ativada.
- **Fluxo Alternativo:**
  - 2a: E-mail já existe → Mensagem de erro e opção de login.
- **Fluxo de Exceção:**
  - 3a: Senha fraca → Erro e prompt para correção.
  - 4a: Falha no e-mail → Retry ou notificação de erro.
- **Pós-condições:** Conta criada e ativada; usuário redirecionado para login.

5.4 UC02: Login do Usuário

- **Atores:** Usuário Anônimo (com conta existente).
- **Pré-condições:** Conta ativada no sistema.
- **Fluxo Principal:**

- Usuário insere e-mail e senha na tela de login.
- Sistema valida credenciais (hash no PostgreSQL) e gera JWT token.
- Sistema solicita 2FA (envia código por e-mail).
- Usuário insere código; sistema valida.
- Sessão iniciada; redireciona para tela principal.

- **Fluxo Alternativo:**
  - 1a: "Lembrar-me"→ Armazena cookie seguro por 7 dias.
- **Fluxo de Exceção:**
  - 2a: Credenciais inválidas → Erro após 5 tentativas (bloqueio temporário).
  - 4a: Código 2FA expirado → Reenvio automático.
- **Pós-condições:** Usuário logado; acesso a dados pessoais liberado.

5.5 UC03: Adicionar Tarefa

- **Atores:** Usuário Logado.
- **Pré-condições:** Usuário na tela principal, logado.
- **Fluxo Principal:**
  - Usuário clica no botão "+"na barra superior.
  - Modal expansível abre: Campos para nome, tipo (dropdown com cores), data (date picker), horário (time picker).
  - Usuário submete; sistema valida (data futura, sem conflitos) e salva no PostgreSQL (associado ao user_id).
  - Modal fecha; tarefa aparece na listagem pendente.
- **Fluxo Alternativo:**
  - 2a: Seleção rápida de tipo → Aplica cor automática na UI.
- **Fluxo de Exceção:**
  - 3a: Conflito de horário → Alerta e sugestão de ajuste.
  - 3b: Campos obrigatórios vazios → Validação em tempo real (vermelho no campo).
- **Pós-condições:** Tarefa adicionada; listagem atualizada.

6 UC04: Editar ou Excluir Tarefa

6.1 UC03: Editar ou Excluir Tarefa

- **Atores:** Usuário Logado.
- **Pré-condições:** Tarefa existe na listagem pendente.
- **Fluxo Principal (Editar):**
  - Usuário clica em ícone de edição na tarefa.
  - Modal abre com campos pré-preenchidos.
  - Usuário altera e submete; sistema atualiza no BD via `PUT`.
- **Fluxo Principal (Excluir):**
  - Usuário clica em ícone de lixeira.
  - Modal de confirmação: "Confirmar exclusão?".
  - Usuário confirma; sistema deleta via `DELETE`.
- **Fluxo de Exceção:**
  - 3a (Editar): Versão desatualizada (conflito) → Merge ou erro.
- **Pós-condições:** Tarefa editada/excluída; listagem atualizada.

7 UC05: Visualizar Calendário

- **Atores:** Usuário Logado.
- **Pré-condições:** Usuário na tela principal.
- **Fluxo Principal:**
  - Usuário clica no ícone de Calendário.
  - Partição expansível mostra grade mensal (dias destacados com tarefas).
  - Usuário clica em um dia; mini-bloco exibe tarefas (nome, tipo, horário).
- **Fluxo Alternativo:**
  - 2a: Navegação por mês (setas) → Carrega tarefas via API.
- **Fluxo de Exceção:**
  - 3a: Dia sem tarefas → Mensagem "Nenhuma tarefa".
- **Pós-condições:** Calendário visualizado; usuário pode clicar em detalhes.

7.1 Gerenciar Alarmes/Notificações

- **Atores:** Usuário Logado.
- **Pré-condições:** Tarefas agendadas próximas (1 dia/1h antes).
- **Fluxo Principal:**
  - Sistema verifica periodicamente (via `cron job` no back-end ou `setInterval` no front).
  - Para tarefas próximas, exibe notificação push (título:'Tarefa próxima: [Nome]').
  - Usuário clica no ícone de Alarme; lista de notificações ativas.
  - Usuário descarta notificação (marca como lida).
- **Fluxo Alternativo:**
  - 1a: Permissões de notificação negadas → Prompt para ativar.
- **Fluxo de Exceção:**
  - 2a: Erro no push → Fallback para badge na UI.
- **Pós-condições:** Usuário notificado; alarmes gerenciados.

7.2 UC07: Visualizar e Marcar Histórico

- **Atores:** Usuário Logado.
- **Pré-condições:** Tarefas concluídas existem.
- **Fluxo Principal:**
  - Usuário clica no ícone de Histórico.
  - Partição mostra lista paginada (tarefas marcadas como concluídas, com data de conclusão).
  - Para tarefa pendente na listagem principal: Usuário marca checkbox "Concluída"; sistema atualiza status no BD.
  - Tarefa move para histórico.
- **Fluxo Alternativo:**
  - 2a: Filtros (data/tipo) → API query com parâmetros.
- **Fluxo de Exceção:**
  - 3a: Falha na marcação → Retry ou erro de rede.
- **Pós-condições:** Histórico atualizado; tarefa transferida.

7.3 UC08: Logout do Usuário

- **Atores:** Usuário Logado.
- **Pré-condições:** Usuário na tela principal.
- **Fluxo Principal:**
  - Usuário clica em botão de logout (na barra superior ou menu).
  - Sistema invalida JWT token e limpa cookies/sessão.
  - Redireciona para tela de login.
- **Fluxo Alternativo:**
  - 1a: Confirmação modal → "Deseja sair?".
- **Fluxo de Exceção:**
  - 2a: Erro na invalidação → Logout forçado via front-end.
- **Pós-condições:** Sessão encerrada; acesso bloqueado.

8 Diagrama de Caso de Uso

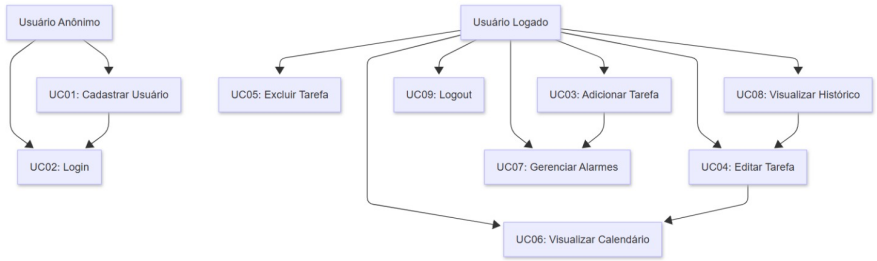

Figura 1: Fluxograma

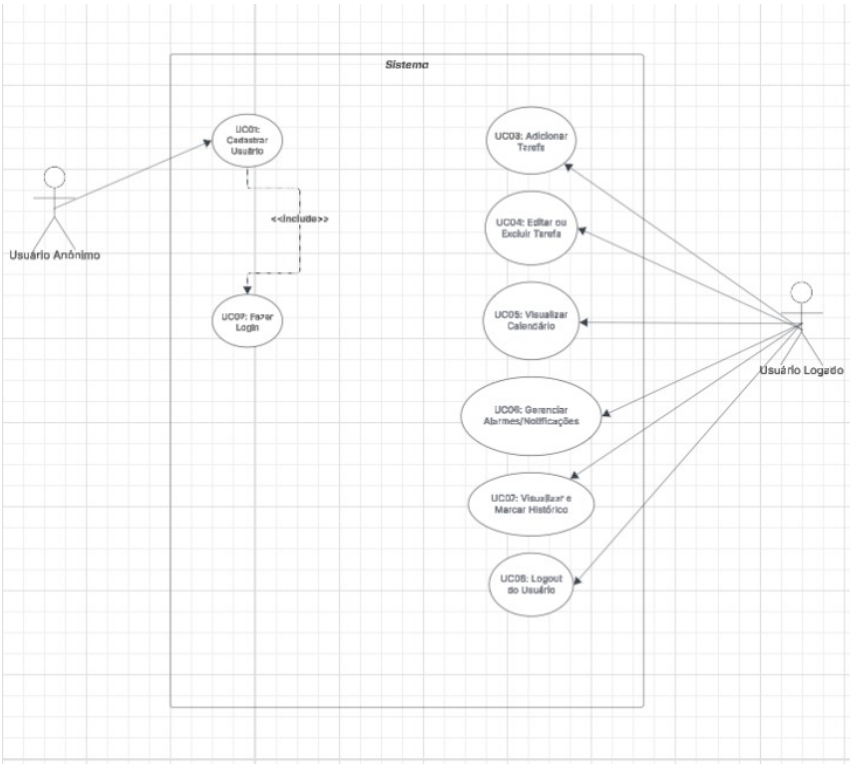

Figura 2: Diagrama

9 Layout do Sistema de Login (Figma)

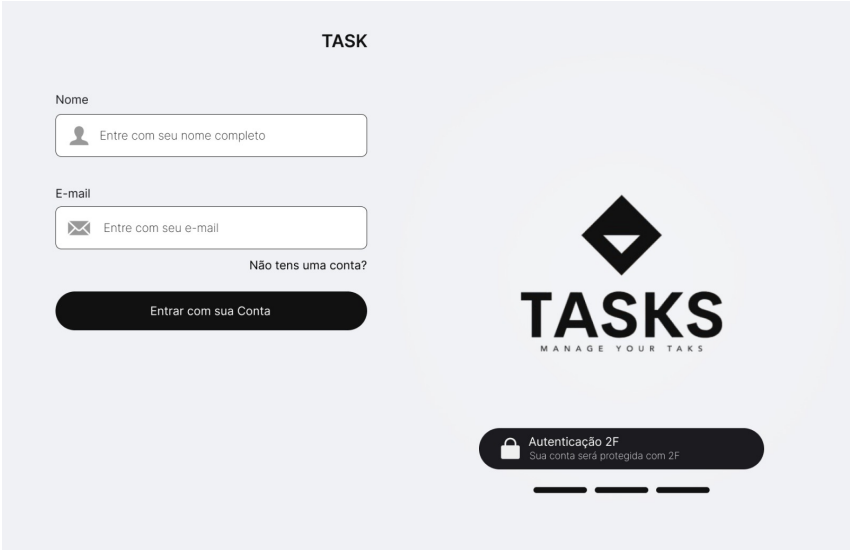

Figura 3: Tela:Login

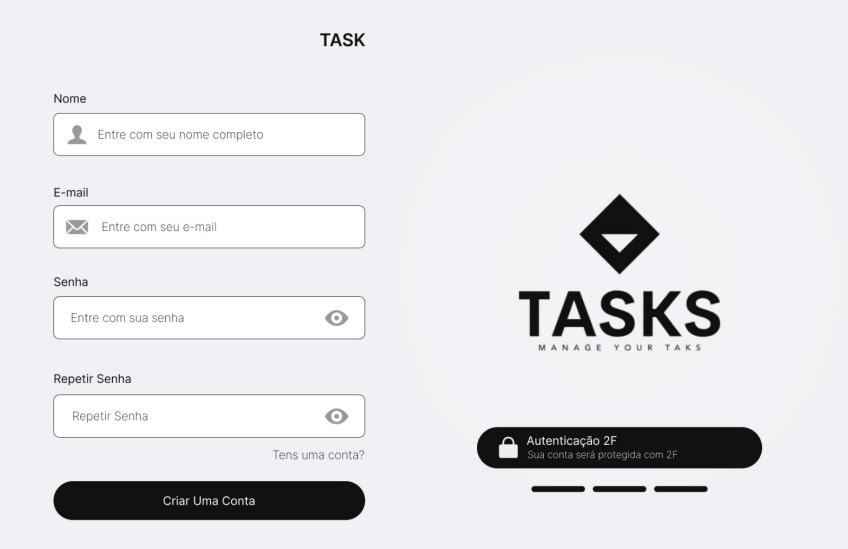

Figura 4: Tela:Cadastro

10 Primeiro módulo de entrega: Tela de Login e Cadastro

10.1 Tela de Cadastro

Figura 5: Tela de Cadastro

Descrição Geral:
Tela responsável pelo **cadastro de novos usuários** no sistema *Chronos*. Permite que o usuário insira seus dados pessoais e crie uma conta para acesso ao sistema. Inclui validações de formato e quantidade de caracteres, além de redirecionamento automático após o cadastro bem-sucedido.

Campos e Componentes:

- **Campo Nome:** Campo de texto para inserção do nome completo do usuário.
  Validações: Aceita apenas letras (maiúsculas e minúsculas), incluindo acentos comuns em português e o caractere cedilha (ç), além de espaços em branco entre os

nomes. Tamanho máximo de 50 caracteres. Não é permitido o uso de **caracteres especiais**, que são definidos como quaisquer símbolos que não correspondam a letras ou espaços, tais como: `! @ # $ % ^ & * ( ) - _ = + [ ] { } ; : , . < > / ? | \"' ‘ ~`.
Exemplos de entradas válidas: *José da Silva*, *Ana Lúcia Ferreira*, *João Cézar*.
Exemplos de entradas inválidas: *João123*, *Maria!*, *Carlos_Henrique*.

- **Campo E-mail:** Campo destinado à inserção do endereço de e-mail do usuário.
  Validações: O e-mail deve seguir um formato válido, contendo obrigatoriamente um nome de usuário, o símbolo `@` e um domínio.
  Aceita subdomínios (ex.: *usuario@sub.dominio.com*) e caracteres especiais permitidos no nome de usuário. Tamanho máximo de 50 caracteres.

- **Campo Senha:** Campo para criação da senha de acesso.
  Validações: Deve conter entre 8 e 20 caracteres. É obrigatório ter pelo menos uma letra maiúscula, um número e um caractere especial(`ex.: ! @ # $ % ^ & * ( ) _ + - = [ ] { } | ; : , . < > ?`).O ícone de "olho" permite visualizar ou ocultar a senha.

- **Campo Repetir Senha:** Campo para confirmação da senha digitada anteriormente.
  Validações: Deve coincidir exatamente com o campo "Senha". Exibe mensagem de erro caso não coincida.

- **Link "Tens uma conta?":** Link de navegação para usuários que já possuem cadastro.
  Ação: Redireciona para a **Tela de Login**.

- **Botão "Criar Uma Conta":** Botão principal da tela.
  Validações: Habilitado apenas quando todos os campos estão preenchidos corretamente e as senhas coincidem. Realiza validação dos dados antes do envio.
  Ação: Envia os dados para o servidor e redireciona para a Tela Principal.

- **Indicador "Autenticação 2FA":** Elemento informativo sobre segurança.
  Ação: Exibe a mensagem "Sua conta será protegida com 2FA", indicando que após o cadastro o usuário deverá ativar a autenticação de duas etapas.

Fluxo Principal:

1. Usuário acessa a tela de cadastro.
2. Preenche todos os campos obrigatórios (Nome, E-mail, Senha e Repetir Senha).
3. O sistema valida os dados inseridos.
4. Ao clicar em **"Criar Uma Conta"**, os dados são enviados para o servidor.
5. Se o cadastro for bem-sucedido, o usuário é redirecionado para a Tela Principal.

Fluxo Alternativo:

- Caso algum campo esteja inválido, o sistema exibe mensagem de erro e o botão "Criar Uma Conta" permanece desabilitado.
- Se o e-mail já estiver cadastrado, o sistema notifica o usuário para utilizar outro endereço.

10.2 Tela: Login de Usuário

Figura 6: Tela de Login

Descrição Geral:
Tela responsável por permitir que usuários cadastrados acessem o sistema *Chronos*. A autenticação é realizada com base nas credenciais cadastradas anteriormente (Nome e E-mail). Após a validação bem-sucedida, o sistema redireciona o usuário para a Tela Principal.

Campos e Componentes:

- **Campo Nome:** Campo de texto destinado à inserção do nome completo do usuário.
  Validações: Aceita apenas letras (maiúsculas e minúsculas), incluindo acentos comuns em português e o caractere cedilha (ç), além de espaços em branco entre os nomes.
  Não é permitido o uso de **caracteres especiais**, definidos como quaisquer símbolos que não correspondam a letras ou espaços, tais como:
  `(ex.: ! @ # $ % ^ & * ( ) _ + - = [ ] { } | ; : , . < > ?)`.
  Exemplo de entradas válidas: *José da Silva*, *Ana Lúcia Ferreira*.
  Exemplo de entradas inválidas: *Maria!*, *Carlos_Henrique*.

- **Campo E-mail:** Campo destinado à inserção do e-mail cadastrado.
  Validações: O e-mail deve conter obrigatoriamente um nome de usuário, o símbolo `@` e um domínio. Aceita subdomínios (ex.: *usuario@sub.dominio.com*) e caracteres especiais permitidos no nome de usuário. Permitir o máximo de 50 caracteres.
  Exemplo de expressão regular recomendada (simplificada e abrangente):
  `[A-Za-z0-9._%+\-]+@[A-Za-z0-9.\-]+\.[A-Za-z]{2,}` O sistema também deve validar se o e-mail informado está cadastrado.

- **Link "Não tens uma conta?":** Link de navegação para novos usuários.
  Ação: Redireciona o usuário para a **Tela de Criação de Conta (Cadastro)**.

- **Botão "Entrar":** Botão principal da tela de login.
  Validações: Habilitado apenas quando ambos os campos (Nome e E-mail) estão

preenchidos corretamente.
Ação: Ao clicar, o sistema valida as credenciais informadas. Se forem válidas, o usuário é redirecionado para a Tela Principal.

- **Indicador "Autenticação 2FA":** Elemento informativo sobre a segurança do sistema.
Ação: Exibe a mensagem "Sua conta será protegida com 2FA", informando que o sistema utiliza autenticação em duas etapas.

Fluxo Principal:

1. Usuário acessa a tela de login.
2. Preenche os campos de Nome e E-mail.
3. O sistema valida as informações inseridas.
4. Ao clicar em **"Entrar"**, o sistema verifica as credenciais.
5. Se forem válidas, o usuário é redirecionado para a Tela Principal.

Fluxo Alternativo:

- Caso o nome ou o e-mail estejam inválidos, o sistema exibe uma mensagem de erro e impede o login.
- Se o e-mail não estiver cadastrado, o sistema notifica o usuário e sugere criar uma nova conta.

10.3 Tecnologias Utilizadas

- Front-end: React + TailwindCSS + Axios
- Back-end: Django REST Framework + PostgreSQL
- Autenticação: JWT + django-otp (para 2FA)