Épico 1 - MVP Funcionalidades básicas

Objetivo: Criar as funcionalidades básicas para que o usuário gerencie suas tarefas de forma organizada.

História 1.1 — CRUD de Tarefas

- ➔ **Como Usuário, quero criar, editar e excluir atividades para organizar meu trabalho**
  - ◆ **Critérios de Aceitação:**
    - O usuário consegue criar uma nova atividade com título e descrição.
    - O usuário consegue editar o título e a descrição de uma atividade existente.
    - O usuário consegue excluir uma atividade.
    - O sistema exibe feedback visual claro para cada ação (criação, edição, exclusão).
  - ◆ **Subtarefas:**
    - Desenvolver telas de formulário para criar, editar e excluir de atividades.
    - Implementar feedback visual das ações (erro ou confirmação)
    - Implementar a lógica de back-end no banco de dados (criar,editar e excluir).
    - Validar dados inseridos nos formulários (ex: campos obrigatórios, formato).
    - Desenvolver integração da interface com o back-end para operações de CRUD

História 1.2 — Anexos em Tarefas

- ➔ **Como Usuário, quero anexar arquivos em tarefas ao concluir para garantir a entrega**
  - ◆ **Critérios de Aceitação:**
    - O usuário consegue anexar um arquivo a uma tarefa concluída ou em edição.
    - O usuário consegue visualizar os arquivos anexados à tarefa.
    - O usuário consegue baixar os arquivos anexados.
  - ◆ **Subtarefas:**
    - Desenvolver interface para um campo de upload de arquivos na tela de conclusão/edição da tarefa.
    - Implementar lógica de back-end para receber e armazenar os arquivos anexados
    - Implementar funcionalidade para permitir ao usuário visualizar os arquivos anexados à tarefa.
    - Implementar funcionalidade para permitir ao usuário baixar os arquivos anexados.

História 1.3 — Atribuição a membros

- ➔ **Como usuário, quero atribuir atividades a membros da equipe para distribuir as responsabilidades**
  - ◆ **Critérios de Aceitação:**

- O usuário consegue selecionar um membro da equipe e atribuí-lo a uma atividade.
- O sistema exibe claramente o membro da equipe atribuído a cada atividade.

◆ **Subtarefas:**

- Desenvolver interface para exibir a lista de membros da equipe (ex: dropdown de seleção ou lista de cards).
- Desenvolver interface para permitir ao usuário selecionar um membro da equipe e associá-lo a uma atividade (no momento de criar ou editar a atividade).
- Implementar lógica de back-end para armazenar a informação de quem está atribuído a cada atividade.
- Implementar lógica de back-end para permitir a filtragem de atividades por membro da equipe.
- Implementar feedback visual na interface para mostrar claramente a quem uma atividade está atribuída.

História 1.4 — Autenticação Básica

➔ **Como usuário, quero acessar a plataforma com Login próprio para maior autonomia e segurança**

◆ **Critérios de Aceitação:**

- O usuário consegue registrar uma nova conta.
- O usuário consegue fazer login com suas credenciais.
- O sistema valida as credenciais de login e oferece feedback em caso de falha.

◆ **Subtarefas:**

- Desenvolver interface de login (campos de usuário/email e senha).
- Implementar funcionalidade de registro de novos usuários.
- Implementar lógica de back-end para autenticação e autorização de usuários
- Desenvolver telas de feedback para login bem-sucedido ou falha (ex: mensagens de erro).

História 1.5 — Log de Auditoria

➔ **Como usuário, quero visualizar um log de auditoria para ter melhor controle das alterações da equipe em relação às tarefas**

◆ **Critérios de Aceitação:**

- O sistema registra automaticamente criações, edições em campos chave da tarefa e a exclusão lógica, incluindo usuário, data/hora, tipo de alteração e dados relevantes (antigos/novos para edições).
- O administrador acessa uma interface que exibe o log de auditoria de forma legível, mostrando quem, quando e o que foi alterado em cada registro.
- Os registros no log de auditoria são imutáveis após a criação.

◆ **Subtarefas:**

- Criar modelo de dados para armazenar informações de auditoria (usuário, data/hora, tipo de alteração, dados antigos/novos, ID da tarefa).

- Implementar lógica para registrar alterações em todos os campos da tarefa (nome, descrição, tema, status, responsável, data de entrega e anexo) no log de auditoria.
- Implementar lógica para registrar a exclusão lógica (soft delete) da tarefa no log de auditoria.
- Criar uma interface de usuário para exibir o log de auditoria, acessível apenas por usuários com perfil de administrador.
- Integrar o log de auditoria com as funcionalidades de criação, edição e exclusão de tarefas (História 1.1).

História 1.6 — Visualização de Tarefas

➔ **Como usuário, quero visualizar tanto as tarefas gerais da equipe quanto às minhas por meio de um perfil, para melhor gerenciamento**

◆ **Critérios de Aceitação:**

- Para usuários com perfil de administrador:
  - O sistema exibe todas as tarefas da equipe.
  - O sistema permite alternar para visualizar apenas as tarefas atribuídas ao próprio administrador.
  - O sistema indica claramente qual visualização está ativa (todas as tarefas ou minhas tarefas).
- Para usuários comuns:
  - O sistema exibe apenas as tarefas atribuídas a ele.

◆ **Subtarefas:**

- Desenvolver e implementar a lógica de perfis de usuário (administrador e comum).
- Implementar funcionalidade de filtro na interface para que o administrador possa alternar entre "Minhas Tarefas" e "Todas as Tarefas da Equipe".
- Implementar lógica de back-end para filtrar as tarefas atribuídas ao administrador logado e para exibir todas as tarefas da equipe.
- Implementar feedback visual claro na interface para indicar qual visualização (minhas tarefas ou tarefas da equipe) está ativa.
- Desenvolver interface para exibir exclusivamente a lista de tarefas atribuídas ao usuário comum logado, sem acesso a tarefas de outros usuários ou da equipe geral.
- Implementar lógica de back-end para filtrar as tarefas garantindo que apenas as atribuídas ao usuário comum logado sejam retornadas, impedindo o acesso a dados de tarefas não relacionadas a ele.

História 1.7 - Notificações de Tarefas

- **Como usuário, quero receber notificações das minhas tarefas na interface para assim ter melhor autonomia do meu tempo.**
  - **Critérios de Aceitação:**
    - O sistema exibe notificações automáticas na interface web quando ocorrem eventos relevantes relacionados às tarefas do usuário (ex: tarefa criada, editada, excluída, prazo próximo, tarefa atrasada, tarefa atribuída).

        - As notificações contém informações relevantes sobre a tarefa (ex: título, prazo, quem atribuiu/editou/excluiu).
        - O usuário consegue acessar a tarefa diretamente através da notificação na interface.
        - O sistema exibe um histórico de notificações recebidas na interface.
    - **Subtarefas:**
        - Desenvolver interface para exibir notificações (Componente de notificação com dropdown).
        - Implementar lógica de back-end para disparar notificações na interface para tarefas relacionadas ao usuário logado, com base em eventos de CRUD (criação, edição, exclusão, prazo, atribuição, etc.).
        - Desenvolver templates para os diferentes tipos de notificações a serem exibidas no sistema.
        - Implementar funcionalidade para que, ao clicarem nas notificações, os usuários sejam direcionados às tarefas.
        - Implementar funcionalidade para exibir o histórico de notificações no componente de notificação com dropdown.

História 1.8 - Recuperação de Senha via E-mail

- **Como usuário, quero recuperar minha senha via e-mail para garantir maior segurança da minha conta.**
    - **Critérios de Aceitação:**
        - O usuário consegue solicitar a recuperação de senha informando seu e-mail cadastrado.
        - O sistema envia um e-mail com um link para redefinição de senha para o e-mail do usuário.
        - O link de redefinição de senha possui uma validade limitada (30 minutos).
        - O usuário consegue redefinir sua senha através do link.
        - O sistema valida a nova senha de acordo com os requisitos de segurança (ex: mínimo de caracteres, complexidade).
        - Após a redefinição, a senha antiga não é mais válida.
        - O sistema exibe feedback visual claro sobre o sucesso ou falha da solicitação e da redefinição de senha.
    - **Subtarefas:**
        - Desenvolver interface para solicitação de recuperação de senha (componente na tela de login e campo de e-mail e botão de envio).
        - Implementar lógica de back-end para gerar e enviar e-mails com links temporários de redefinição de senha.
        - Desenvolver interface para redefinição de senha (campos para nova senha e confirmação).
        - Implementar lógica de back-end para validar e atualizar a senha do usuário.
        - Definir e implementar a validade do link temporário de redefinição de senha.

- Implementar validação da nova senha no front-end e back-end.
- Desenvolver telas de feedback para sucesso e erro na recuperação e redefinição de senha.

História 1.9 - Gerenciamento de Equipes

➔ **Como Administrador, quero criar e gerenciar equipes e seus membros para organizar a colaboração e a atribuição de tarefas.**

◆ **Critérios de Aceitação:**

- O administrador consegue criar uma nova equipe, definindo um nome e uma descrição.
- O administrador consegue adicionar membros existentes ao sistema buscando por e-mail a uma equipe.
- O administrador consegue remover membros de uma equipe.
- O administrador consegue visualizar a lista de todas as equipes e seus respectivos membros.
- O administrador consegue editar o nome e a descrição de uma equipe existente.
- O sistema exibe feedback visual claro para cada ação (criação, adição/remoção de membros, edição de equipe).

◆ **Subtarefas:**

- Desenvolver telas de formulário para criar e editar equipes.
- Desenvolver interface para adicionar e remover membros de uma equipe (ex: lista de seleção de usuários).
- Implementar a lógica de back-end para criar, editar e excluir equipes no banco de dados.
- Implementar a lógica de back-end para associar usuários a equipes.
- Validar dados inseridos nos formulários (ex: nome da equipe obrigatório, evitar nomes duplicados).
- Desenvolver integração da interface com o back-end para operações de gerenciamento de equipes.
- Implementar feedback visual das ações (erro ou confirmação).

História 2.0 — Prazo e Calendário

➔ **Como usuário, quero sincronizar os prazos das minhas tarefas com meus calendários para planejar e gerenciar meu tempo de forma eficiente.**

◆ **Critérios de Aceitação:**

- O sistema exibe a data de criação e o prazo de conclusão para cada atividade.
- O usuário consegue definir/editar o prazo de conclusão de uma atividade.

- O sistema sinaliza visualmente atividades próximas do prazo ou atrasadas.
- O usuário consegue conectar sua conta do Google Calendar ao sistema.
- As atividades com prazo de conclusão são automaticamente sincronizadas como eventos nos calendários conectados do usuário.
- Alterações no prazo de conclusão da atividade no sistema são refletidas nos eventos do calendário conectado.
- O usuário consegue desativar a sincronização com o calendário a qualquer momento.

◆ **Subtarefas:**

- Implementar feedback visual para atividades que estão próximas do prazo ou atrasadas (ex: mudança de cor, ícone).
- Desenvolver interface para as configurações de integração com calendários (botões para conectar/desconectar Office 365,Google Calendar ou outro calendário).
- Implementar a lógica de back-end para autenticação e autorização via APIs do Office 365,Google Calendar ou outro calendário.
- Implementar a lógica de back-end para criar, atualizar e excluir eventos nos calendários externos com base nos prazos das atividades.
- Garantir a independência na sincronização de eventos para evitar duplicações ou inconsistências.
- Desenvolver feedback visual para o status da integração com o calendário (conectado/desconectado, erros de sincronização).

Épico 2 - Interface Intuitiva e Multiplataforma

Objetivo: Proporcionar uma experiência de usuário responsiva e intuitiva em diferentes dispositivos, com foco na autonomia do usuário e na entrega de informações importantes.

História 2.1 — Categorização de Tarefas

➔ **Como usuário, quero categorizar atividades por prioridade, tipo, ou área para facilitar a visualização**

◆ **Critérios de Aceitação:**

- O usuário consegue criar/definir categorias para as atividades.
- O usuário consegue associar categorias a uma atividade.
- O usuário consegue filtrar atividades por categoria (prioridade, tipo, data).
- As categorias são visualmente distintivas na interface.

◆ **Subtarefas:**

- Desenvolver interface para permitir a criação/definição de categorias (ex: "Urgente", "Marketing", "Desenvolvimento", "Reunião").
- Desenvolver interface para permitir ao usuário associar/selecionar categorias a uma tarefa(no momento de criar ou editar).
- Implementar lógica de back-end para armazenar as informações de categoria para cada tarefa.
- Desenvolver funcionalidades de filtro na interface (ex: botões ou dropdowns) para que o usuário possa visualizar atividades por prioridade, tipo ou área.
- Implementar lógica de back-end para permitir a filtragem de atividades por categoria
- Implementar feedback visual na interface para que as categorias sejam claramente visíveis (ex: cores diferentes para prioridades, ícones para tipos).