Universidade Cidade de São Paulo

Ciência da Computação

Kayky Fernandes Alves, Jennifer Monteiro Santos, Felipe Santana Portela Cordeiro, Pedro Henrique dos Santos Oliveira, Nicolas Frutuoso Lima de Souza, Luccas André de Souza Melim, Guilherme Irving Trama, Bruna Nunes Ribeiro Mota, Laura Alves Feliciano e Christian Dantas de Oliveira.

Projeto laboratório de Banco de dados Avançado:

Sistema de Gestão Kanban - Ruby Alignment

São Paulo

2025

Grupo:

Kayky Fernandes Alves RGM: 31182003	Pedro Henrique dos Santos Oliveira RGM: 31274820
Felipe Santana Portela Cordeiro RGM: 30477531	Jennifer Monteiro Santos RGM: 29259193
Nicolas Frutuoso Lima de Souza RMG: 31810233	Christian Dantas de Oliveira RGM: 30017891
Luccas André de Souza Melim RGM: 29725824	Guilherme Irving Trama RGM: 30208581
Bruna Nunes Ribeiro Mota RGM: 31007325	Laura Alves Feliciano RGM: 31702635

Tema: Sistema de Gestão Kanban - Ruby Alignment

1. Definição de Objetivos:
   O objetivo do projeto é desenvolver um Sistema de Gestão Kanban para a empresa fictícia RUBY Alignment, com foco em organizar, monitorar e otimizar o fluxo de tarefas e projetos internos.
   O sistema será uma aplicação web desenvolvida em HTML, CSS e JavaScript, com base de dados estruturada em MySQL, permitindo o gerenciamento visual dos processos de forma ágil e intuitiva.
   Os objetivos principais são:
   - Fornece uma visão clara e em tempo real do andamento das tarefas.
   - Melhorar a organização das equipes por meio da metodologia Kanban.
   - Garantir segurança e centralização dos dados em um ambiente único.
   - Apoiar a tomada de decisões gerenciais por meio de consultas e relatórios.
   - Promover a colaboração entre os membros da RUBY Alignment, diferenciando os acessos de acordo com os perfis (Administrador, Colaborador e Solicitante).
2. Identificação de Stakeholders:
   a. Stakeholders Internos (Scrum Team):
      - Product Owner (PO): Guilherme Irving Trama
        - Entender as necessidades da empresa/usuários.
        - Traduzir essas necessidades em requisitos no backlog.
        - Definir prioridades das entregas.
        - Validar os incrementos desenvolvidos ao final das Sprints.
      - Scrum Master: Jennifer Monteiro Santos
        - Garantir que a equipe siga os princípios do Scrum.
        - Atuar como facilitadora nas reuniões e eventos do Scrum.
        - Remover impedimentos que atrapalhem o andamento da equipe.
        - Apoiar a comunicação entre o PO, o time e os stakeholders externos.
      - Development Team (Time de Desenvolvimento):
        - Front-End: Felipe Santana Portela Cordeiro, Bruna Nunes Ribeiro Mota e Laura Alves Feliciano.
        - Back-End: Pedro Henrique dos Santos Oliveira, Luccas André de Souza Melim,
        - Banco de Dados: Jennifer Monteiro Santos, Kayky Fernandes Alves,

        - BI e Dashboard: Guilherme Irving Trama, Nicolas Frutuoso Lima de Souza e Christian Dantas de Oliveira.
    - Implementar o sistema de acordo com o backlog priorizado pelo PO.
    - Trabalhar de forma colaborativa e multidisciplinar (front-end, back-end, banco de dados e BI e Dashboard).
    - Garantir a qualidade do código e da base de dados.
    - Entregar incrementos utilizáveis ao final de cada Sprint.
  b. Stakeholders externos (projeto acadêmico / empresa fictícia):
    i. Equipe acadêmica:
      - Professora orientadora: Ângela Perez Barcellos
    ii. Gestores/Diretoria da RubyAlignment:
      - Diretora Geral (CEO): Fernanda Oliveira – responsável pela visão estratégica da empresa.
      - Gestor Operacional (CTO): Ricardo Mendes – responsável pela gestão técnica e acompanhamento dos processos de desenvolvimento de software.
      - Usuários finais do sistema (funcionários da RUBY Alignment):
      - Administrador: Carlos Silva – Gerente de Projetos da RUBY Alignment.
      - Colaborador: Mariana Souza – Desenvolvedora Full Stack da Ruby Alignment.
      - Solicitante: Pedro Lima – Analista de Qualidade de Software (QA).
      - Colaborador: João Pereira – Estagiário de Desenvolvimento.
3. Escopo do Projeto:
  a. Incluído:
    - Modelagem do banco de dados para suportar usuários, tarefas e fluxo Kanban.
    - Implementação da aplicação web utilizando HTML, CSS e Javascript.

    Funcionalidades:

    - Cadastro e gerenciamento de usuários.
    - Sistema de login com perfis distintos (Administrador, Colaborador e Solicitante).
    - Criação, edição e movimentação de tarefas nas colunas Kanban (A Fazer, Em Andamento, Concluído).
    - Relatórios de acompanhamento das atividades.
    - Registro de logs das ações realizadas.
  b. Excluído:
    - Integrações externas com sistemas como Trello, Jira ou Slack.
    - Escalabilidade para múltiplas empresas ou grandes volumes de usuários.
4. Cronograma e Orçamento: (fase 1 até aqui)
  a. Cronograma (macro):
    Início do projeto: 27/08/2025

- Fase 1: Planejamento e Iniciação (Data de Entrega: 04 de setembro) - Concluído.

- Fase 2: Levantamento e Análise de Requisitos (Data de Entrega: 11 de setembro):
    - Planejamento do Sprint: Presencial - Quinta-feira 04/09.
    - Reuniões diárias: Online -15 minutos -Sexta-feira 05/09.
    - Revisão e retrospectiva do Sprint: Online - Sábado 06/09.
    - Reuniões diárias: Online -15 minutos - Segunda-feira 08/09.
    - Reuniões diárias: Presencial - 15 minutos - Terça-feira 09/09.
    - Reuniões diárias: Presencial - 15 minutos - Quarta-feira 10/09.
    - Execução do Sprint: 04/09 a 11/09.
    - Entrega da fase: 11/09
- Fase 3: Modelagem Conceitual (Data de Entrega: 18 de setembro)
    - Planejamento do Sprint: Presencial - Quinta-feira 11/09.
    - Reuniões diárias: Online -15 minutos -Sexta-feira 12/09.
    - Revisão e retrospectiva do Sprint: Online - Sábado 13/09.
    - Reuniões diárias: Online -15 minutos - Segunda-feira 15/09.
    - Reuniões diárias: Presencial - 15 minutos - Terça-feira 16/09.
    - Reuniões diárias: Presencial - 15 minutos - Quarta-feira 17/09.
    - Execução do Sprint: 11/09 a 18/09.
    - Entrega da fase: 18/09
- Fase 4: Modelagem Lógica (Data de Entrega: 25 de setembro):
    - Planejamento do Sprint: Presencial - Quinta-feira 18/09.
    - Reuniões diárias: Online -15 minutos -Sexta-feira 19/09.
    - Revisão e retrospectiva do Sprint: Online - Sábado 20/09.
    - Reuniões diárias: Online -15 minutos - Segunda-feira 22/09.
    - Reuniões diárias: Presencial - 15 minutos - Terça-feira 23/09.
    - Reuniões diárias: Presencial - 15 minutos - Quarta-feira 24/09.
    - Execução do Sprint: 18/09 a 25/09.
    - Entrega da fase: 25/09
- Fase 5: Modelagem Física (Data de Entrega: 09 de outubro):
    - Planejamento do Sprint: Presencial - Quinta-feira 25/09 e 02/10.
    - Reuniões diárias: Online -15 minutos -Sexta-feira 26/09 e 03/10.
    - Revisão e retrospectiva do Sprint: Online - Sábado 27/09 e 04/10.
    - Reuniões diárias: Online -15 minutos - Segunda-feira 29/09 e 06/10.
    - Reuniões diárias: Presencial - 15 minutos - Terça-feira 30/09 e 07/10.
    - Reuniões diárias: Presencial - 15 minutos - Quarta-feira 01/10 e 08/10.
    - Execução do Sprint: 25/09 a 09/10.
    - Entrega da fase: 09/10
- Fase 6: Apresentação do Projeto. (Datas da Apresentaçôes: 16.10 | 23.10 | 30.10):
    - Planejamento do Sprint: Presencial - Quinta-feira 09/10.
    - Reuniões diárias: Online -15 minutos -Sexta-feira 10/10.
    - Revisão e retrospectiva do Sprint: Online - Sábado 11/10.
    - Reuniões diárias: Online -15 minutos - Segunda-feira 13/10.
    - Reuniões diárias: Presencial - 15 minutos - Terça-feira 14/10.
    - Reuniões diárias: Presencial - 15 minutos - Quarta-feira 15/10.
    - Execução do Sprint: 09/10 a 16/10.
    - Entrega da fase: 16/10
    - b. Orçamento:

Cargo / Função	**Ferramentas / Tecnologias**	**Faixa salarial nível Júnior (R$ / mês)**

Product Owner (PO)	Planejamento, backlog, HTML, CSS e Javascript.	R$ 6.000 - 12.000
Scrum Master	Facilitação de Sprints, HTML, CSS e Javascript.	R$ 6.000 - 12.000
Dev Front-End	HTML, CSS e Javascript.	R$ 2.500 - 4.500
Dev Back-End	Python	R$ 3.500 – 5.000
Engenheiro de Banco de Dados	MySQL	R$ 2.700 – 5.000

5. Documento de Requisitos Funcionais:
   Sistema de Distribuição de Demandas com Inteligência Artificial e BI:
   a. Objetivo do Sistema:

      O sistema tem como finalidade otimizar a distribuição de demandas entre colaboradores através de um direcionamento inteligente baseado em hard skills. A plataforma permite o cadastro de usuários, criação e monitoramento de tarefas, além de oferecer dashboards analíticos em tempo real. Um algoritmo de IA sugere automaticamente o colaborador mais qualificado para cada demanda.
   b. Perfis de Usuário:
      Colaborador: Receber demandas e atualizar status
      Solicitante: Criar demandas, acompanhar status, receber notificações.
      Administrador: Visualizar e gerenciar colaboradores/demandas, reatribuir manualmente, gerar relatórios.
   c. Funcionalidades do Sistema:
      - Gestão de usuários (cadastro, edição, exclusão)
      - Criação de demandas com título, descrição, prioridade e prazo
      - IA sugere colaboradores com base nas hard skills
        - Barra de aderência mostra compatibilidade
      - Status: pendente, em andamento, concluída
      - Dashboard/BI com métricas e relatórios
   d. Banco de Dados:
      Tabelas principais:

      Colaboradores: id, nome, departamento, hard_skills, senha

      Solicitantes: id, nome, departamento

      Demandas: id, título, descrição, prioridade, prazo, status, id_solicitante, id_colaborador, aderência

   e. Inteligência Artificial:
      Objetivo: sugerir colaboradores com base em hard skills e disponibilidade.
      Tecnologias: Python, scikit-learn (TF-IDF + Cosine Similarity)

Fluxo: Demanda criada → IA analisa requisitos (TF-IDF) → compara skills dos colaboradores → gera ranking por cosine similarity → colaborador é atribuído → acompanhamento via Kanban.

f. BI e Dashboard:
Objetivo: monitorar produtividade, status e aderência.
Tecnologias: Power BI, Python + Plotly/Dash, ETL.
Métricas: concluídas, tempo médio, produtividade por colaborador e aderência média.

g. Linguagens e Tecnologias:
- Backend: Python (Flask), MySQL, scikit-learn, Pandas, NumPy.
- Frontend: HTML, CSS, JavaScript.
- Banco de Dados: MySQL.
- IA: Python, scikit-learn (TF-IDF + Cosine Similarity), Pandas, NumPy.
- BI: Power BI, Plotly/Dash
- Segurança: JWT
- Infraestrutura: Docker, Git, Celery/RabbitMQ

h. Regras de Negócio:
I. Cada demanda precisa de pelo menos um colaborador disponível.
II. Hard skills são obrigatórias no cadastro.
III. Barra de aderência deve ser exibida.
IV. Histórico registra todas as alterações.
V. Gestores podem reatribuir demandas manualmente.

i. Integração do Sistema:
- Frontend → Backend: Consumo direto da API REST do Flask via Fetch API.
- Backend → Banco de Dados: Conexão direta com MySQL via Flask-MySQLdb.
- IA/ML: Processamento interno com scikit-learn (TF-IDF + Cosine Similarity).
- BI: Conexão direta ao MySQL para extração de dados.
- Autenticação: Sistema de sessão baseado em cookies.
- Logs: Registro em banco de dados para auditoria.

6. Documento de Requisitos Não Funcionais:
   a. Desempenho:
      i. O carregamento da interface principal (quadro Kanban) deve ocorrer em até 3 segundos após o login.
      ii. As consultas ao banco de dados devem ter tempo de resposta máximo de 2 segundos para até 1.000 registros.
      iii. A movimentação de tarefas entre colunas deve ser processada de forma imediata (< 1 segundo).
   b. Segurança:

i. O sistema deve restringir o acesso mediante autenticação por login e senha, garantindo armazenamento seguro das credenciais.
ii. Senhas dos perfis NORMAL e ESTAG devem ser armazenadas utilizando hash SHA-256 com codificação Base64, conforme especificação de segurança.
iii. O sistema deve registrar logs de todas as ações críticas, incluindo login, criação de tarefas, alterações de status e exclusão de dados.
iv. Perfis de usuário terão níveis de acesso diferentes:
Administrador: gerenciamento total do sistema.
Solicitante: cadastro e acompanhamento de projetos/tarefas.
Colaborador: atribuição a projetos e atualização de status.

c. Escalabilidade:
i. O sistema deve suportar inicialmente até 100 usuários simultâneos sem perda perceptível de desempenho.
ii. O banco de dados deve armazenar no mínimo 10.000 registros de tarefas sem comprometer a performance.
iii. A arquitetura deve permitir a expansão para maior volume de usuários apenas ajustando a infraestrutura (ex.: planilha maior, integração futura a um SGBD robusto).

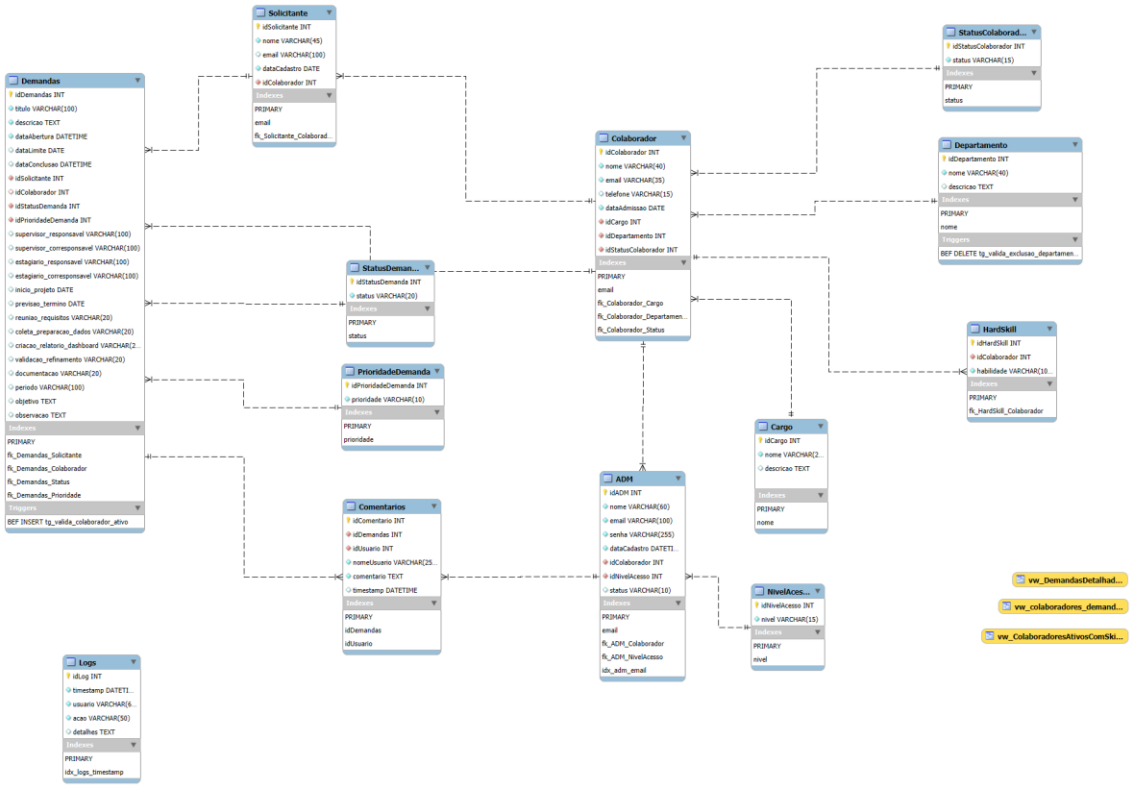
d. Usabilidade:
i. A interface deve ser responsiva, adaptando-se a desktops, notebooks, tablets e celulares.
ii. O layout do Kanban deve seguir padrões visuais claros, com colunas bem definidas: A Fazer, Em Andamento, Concluído.
iii. A navegação deve ser intuitiva, sem necessidade de treinamento avançado para usuários novos.
iv. Ícones, elementos visuais e cores devem ser utilizados para permitir a identificação rápida de status e ações, minimizando a curva de aprendizado.

e. Confiabilidade e Disponibilidade:
i. O sistema deve permanecer disponível 24 horas por dia, 7 dias por semana, considerando sua execução na infraestrutura Google.
ii. Deve existir mecanismo de recuperação de dados, incluindo backup automático diário no Google Drive, para minimizar impactos de falhas.
iii. O sistema não deve permitir a perda de dados em operações críticas, como movimentação de tarefas ou exclusão de usuários.

f. Manutenibilidade:
i. O código deve ser documentado e estruturado para permitir fácil manutenção e evolução.
ii. As alterações futuras (ex.: novas colunas Kanban ou relatórios adicionais) devem ser implementadas com impacto mínimo no sistema existente.

g. Conformidade Regulamentar:
    i. O sistema deve estar em conformidade com a LGPD (Lei Geral de Proteção de Dados), garantindo confidencialidade e controle sobre dados pessoais dos usuários.
    ii. Apenas administradores podem acessar ou modificar informações críticas dos usuários.
h. Portabilidade:
    i. O sistema deve ser acessível via navegador web, sem necessidade de instalação local.
    ii. Compatibilidade garantida com sistemas operacionais Windows, Linux e macOS.
i. Interoperabilidade:
    i. O sistema deve permitir futura integração com ferramentas de produtividade (ex.: exportar relatórios para PDF ou Excel).
    ii. Dados armazenados na planilha devem ser estruturados para fácil migração para outros SGBDs.

7. Diagrama de Entidade-Relacionamento (ER):



8. Dicionário de Dados:

Tabela: PrioridadeDemanda - Define os níveis de prioridade das demandas

Atributo	Tipo	Tamanho	Descrição	Restrições
idPrioridadeDemanda	INT	-	Identificador único da prioridade	PRIMARY KEY, AUTO_INCREMENT
prioridade	VARCHAR	10	Nível de prioridade ('Baixa', 'Média', etc.)	NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Demandas - Registro principal das demandas do sistema

Atributo	Tipo	Tamanho	Descrição	Restrições
idDemandas	INT	-	Identificador único da demanda	PRIMARY KEY, AUTO_INCREMENT
titulo	VARCHAR	100	Título/resumo da demanda	NOT NULL
descricao	TEXT	-	Descrição detalhada da demanda	NOT NULL
databáctrum	DATETIME	-	Data e hora de abertura da demanda	NOT NULL
dataUnite	DATE	-	Data limite para conclusão	Pode ser NULL
dataCondusso	DATETIME	-	Data e hora de conclusão efetiva	Pode ser NULL
idSolicitante	INT	-	Solicitante da demanda	FOREIGN KEY (Solicitante), NOT NULL
idColaborador	INT	-	Colaborador responsável pela demanda	FOREIGN KEY (Colaborador), NOT NULL
idStatusDemanda	INT	-	Status atual da demanda	FOREIGN KEY (StatusDemanda), NOT NULL
idPrioridadeDemanda	INT	-	Prioridade da demanda	FOREIGN KEY (PrioridadeDemanda), NOT NULL
supervisor_regromavei	VARCHAR	100	Supervisor responsável	Pode ser NULL
supervisor_corresponsavel	VARCHAR	100	Supervisor corresponsável	Pode ser NULL
cetagismo_responsavel	VARCHAR	100	Cetagismo responsável	Pode ser NULL
cetagismo_corresponsavel	VARCHAR	100	Cetagismo corresponsável	Pode ser NULL
inicio_projeto	DATE	-	Data de início do projeto	Pode ser NULL
previstos_termino	DATE	-	Previsão de término	Pode ser NULL
reunião_requisitos	VARCHAR	20	Status da reunião de requisitos	Pode ser NULL
coleta_preparacao_dados	VARCHAR	20	Status da coleta e preparação de dados	Pode ser NULL
críticos_relatório_dashboard	VARCHAR	20	Status dos críticos/relatórios/dashboards	Pode ser NULL
validacao_refinamento	VARCHAR	20	Status da validação e refinamento	Pode ser NULL
documentacao	VARCHAR	20	Status da documentação	Pode ser NULL

Atributo	Tipo	Tamanho	Descrição	Restrições
periodo	VARCHAR	100	Período da demanda	Pode ser NULL
objetivo	TEXT	-	Objetivo da demanda	Pode ser NULL
observacao	TEXT	-	Observações adicionais	Pode ser NULL

Tabela: Solicitante - Armazena informações dos solicitantes das demandas

Atributo	Tipo	Tamanho	Descrição	Restrições
idSolicitante	INT	-	Identificador único do solicitante	PRIMARY KEY, AUTO_INCREMENT
nome	VARCHAR	40	Nome completo do solicitante	NOT NULL
email	VARCHAR	30	E-mail do solicitante	NOT NULL, UNIQUE
datacolaborator	DATE	-	Data do cadastro do solicitante	NOT NULL
idColaborador	INT	-	Colaborador vinculado	FOREIGN KEY (Colaborador), NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: StatusColaborador - Controla o status do colaborador (ativo/inativo)

Atributo	Tipo	Tamanho	Descrição	Restrições
idStatusColaborador	INT	-	Identificador único do status	PRIMARY KEY, AUTO_INCREMENT
status	VARCHAR	15	Status do colaborador ('ativo', 'inativo')	NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Colaborador - Armazena informações dos colaboradores que atendem demandas

Atributo	Tipo	Tamanho	Descrição	Restrições
idColaborador	INT	-	Identificador único do colaborador	PRIMARY KEY, AUTO_INCREMENT
nome	VARCHAR	40	Nome completo do colaborador	NOT NULL
email	VARCHAR	35	E-mail corporativo do colaborador	NOT NULL, UNIQUE
telefone	VARCHAR	15	Telefone para contato	Pode ser NULL
databalmissao	DATE	-	Data de admissão do colaborador	NOT NULL
idCargo	INT	-	Cargo/função do colaborador	FOREIGN KEY (Cargo), NOT NULL
idDepartamento	INT	-	Departamento do colaborador	FOREIGN KEY (Departamento), NOT NULL
idStatusColaborador	INT	-	Status atual do colaborador	FOREIGN KEY (StatusColaborador), NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Hardskill - Habilidades técnicas dos colaboradores

Atributo	Tipo	Tamanho	Descrição	Restrições
idHardskill	INT	-	Identificador único da habilidade	PRIMARY KEY, AUTO_INCREMENT
idColaborador	INT	-	Colaborador que possui a habilidade	FOREIGN KEY (Colaborador), NOT NULL
habilidade	VARCHAR	100	Descrição da habilidade técnica	NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Departamento - Cadastro dos departamentos da empresa

Atributo	Tipo	Tamanho	Descrição	Restrições
idDepartamento	INT	-	Identificador único do departamento	PRIMARY KEY, AUTO_INCREMENT
nome	VARCHAR	40	Nome do departamento	NOT NULL
descricao	TEXT	-	Descrição detalhada do departamento	Pode ser NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Logs - Registro de logs do sistema

Atributo	Tipo	Tamanho	Descrição	Restrições
idLog	INT	-	Identificador único do log	PRIMARY KEY, AUTO_INCREMENT
timestamp	DATETIME	-	Data e hora do log	NOT NULL
usuario	VARCHAR	50	Usuário que realizou a ação	NOT NULL
acao	VARCHAR	50	Ação realizada	NOT NULL
detalhes	TEXT	-	Detalhes da ação	Pode ser NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: StatusDemanda - Define os possíveis status de uma demanda

Atributo	Tipo	Tamanho	Descrição	Restrições
idStatusDemanda	INT	-	Identificador único do status	PRIMARY KEY, AUTO_INCREMENT
status	VARCHAR	20	Status da demanda ('Aberta', 'Em andamento', etc.)	NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: ADM - Usuários administrativos do sistema

Atributo	Tipo	Tamanho	Descrição	Restrições
idADM	INT	-	Identificador único do administrador	PRIMARY KEY, AUTO_INCREMENT
nome	VARCHAR	60	Nome completo do administrador	NOT NULL
email	VARCHAR	100	E-mail do administrador	NOT NULL, UNIQUE
senha	VARCHAR	255	Senha criptografada do administrador	NOT NULL
datacolaborator	DATETIME	-	Data e hora do cadastro	NOT NULL
idColaborador	INT	-	Colaborador vinculado ao usuário ADM	FOREIGN KEY (Colaborador), NOT NULL
idRivelAcceso	INT	-	Nível de acesso do administrador	FOREIGN KEY (NivelAcesso), NOT NULL

Atributo	Tipo	Tamanho	Descrição	Restrições
nivel	VARCHAR	15	Nível de acesso ('super admin', 'admin comum')	NOT NULL
indicaco	-	-	Indicações adicionais	Pode ser NULL

Tabela: Cargo - Define os cargos/funções dos colaboradores

Atributo	Tipo	Tamanho	Descrição	Restrições
idCargo	INT	-	Identificador único do cargo	PRIMARY KEY, AUTO_INCREMENT
nome	VARCHAR	20	Nome do cargo/função	NOT NULL, UNIQUE
descricao	TEXT	-	Descrição das responsabilidades do cargo	Pode ser NULL

Tabela: NivelAcesso - Define os níveis de acesso administrativo

Atributo	Tipo	Tamanho	Descrição	Restrições
idNivelAcesso	INT	-	Identificador único do nível de acesso	PRIMARY KEY, AUTO_INCREMENT
nivel	VARCHAR	15	Nível de acesso ('super admin', 'admin comum')	NOT NULL, UNIQUE

9. Casos de Uso:

a. Primeiro Acesso:

Ao receber suas credenciais por e-mail, siga os passos abaixo para acessar o sistema pela primeira vez.

i. Fazendo Login:

1. Acesse o link do sistema fornecido.
2. Uma tela de "Acesso Restrito" será exibida.
3. Digite o Usuário e Senha que você recebeu por e-mail.
4. Clique em "Entrar".

ii. Alterando sua Senha

Por segurança, no seu primeiro acesso, você será obrigado a criar uma nova senha.

1. Na tela "Alteração de Senha Obrigatória", digite uma nova senha no campo "Nova Senha". A senha deve ter no mínimo 6 caracteres.
2. Digite a mesma senha no campo "Confirme a Nova Senha".
3. Clique em "Salvar Nova Senha".
4. Após a confirmação, você será desconectado. Faça o login novamente com seu usuário e sua nova senha.

b. Perfis de Usuário e Funcionalidades:

O sistema possui três tipos de perfis, cada um com acesso a diferentes funcionalidades.

i. Perfil Solicitante:

Contém: Nome do usuário, e-mail, Telefone, CPF, Senha

Nome: Nome definido pelo usuário no momento do cadastro, o mesmo deve ser igual ao seu documento oficial (RG/RNE);

E-mail: Endereço para atualização do status da solicitação e usado para acessar a plataforma;

Telefone: Telefone para possíveis contatos e alinhamentos;

CPF: Validação de identidade do usuário;

Funcionalidade Principal: Cadastro de Projeto

1. No menu de navegação, clique em "Cadastro de Projeto".
2. Preencha os campos do formulário:
    a. Nome do Projeto: Um título claro para a demanda.
    b. Supervisor Responsável/Corresponsável: Nomes dos supervisores envolvidos.
    c. Objetivo: Uma breve descrição do propósito do projeto.
    d. Descritivo: Detalhes adicionais sobre o que precisa ser feito.
3. Clique no botão "Cadastrar" para salvar o projeto.

ii. Perfil Colaborador:

Usuários com este perfil são responsáveis por executar os projetos.
Contém: Nome, E-mail, Cargo, Departamento, Telefone, Status, Data de Admissão (Para visualização da gestão), Status, Hard Skill.
Nome: Nome definido pelo usuário no momento do cadastro, o mesmo deve ser igual ao seu documento oficial (RG/RNE);
E-mail: Endereço para atualização do status da solicitação e recebimento das demandas (pode ser gerado um e-mail corporativo). Usado para acessar a plataforma;
Telefone: Telefone para possíveis contatos e alinhamentos;
CPF: Validação de identidade do usuário e geração da matrícula pessoal;
Senha: Senha pessoal para acessos na plataforma.
Status: Necessário para verificar se o colaborador está ativo na empresa, se está com a caixa de demandas cheia;
Hard Skill: Campo essencial para validação da aderência usada para redirecionamento das demandas.

Funcionalidade 1: Aderir a Projetos:

a. No menu, clique em "Aderir a Projetos".
b. Você verá uma lista de projetos com o status "NÃO INICIADO".
c. Clique em um projeto para expandir seus detalhes.
d. Preencha o formulário de adesão:

i. Estagiário Responsável/Corresponsável: Seu nome e o de um colega, se aplicável.
ii. Início do Projeto: A data em que o trabalho começará.
iii. Previsão de Término: A data estimada para a conclusão.

e. Clique em "Aderir ao Projeto". O status do projeto mudará para "EM ANDAMENTO".

Funcionalidade 2: Andamento dos Projetos:

f. No menu, clique em "Andamento dos Projetos".
g. Você verá os projetos que estão "EM ANDAMENTO" ou "EM MONITORAMENTO".
h. Expanda o projeto que deseja atualizar.
i. Atualize o status de cada etapa (Reunião, Coleta de Dados, etc.) para "EM ANDAMENTO" ou "REALIZADO".
j. Se necessário, ajuste a Previsão de Término ou adicione uma Data de Conclusão.
k. Quando todas as etapas estiverem concluídas, mude o Status geral para "EM MONITORAMENTO".
l. Clique em "Atualizar Andamento" para salvar as alterações.

iii. Perfil Administrador:

Usuários com este perfil têm funções de gerenciamento e supervisão.

Contém: Nome, E-mail, Telefone, Senha, Nível de Acesso, Data admissão, Organograma, Relatório de solicitações, gerenciador de perfis.

Nome: Nome definido pelo usuário no momento do cadastro, o mesmo deve ser igual ao seu documento oficial (RG/RNE);

E-mail: Endereço para atualização do status da solicitação e usado para acessar a plataforma;

Telefone: Telefone para possíveis contatos e alinhamentos;

Senha: Senha pessoal para acessos na plataforma.

Nível de Acesso: Necessário para gerenciar o controle das funcionalidades;

Organograma: Arvore de hierarquia;

Relatório de solicitações: Verificar status e andamento das solicitações, assim como verificar históricos;

Gerenciador de perfis: Necessário para alteração de status, tais como promoções, desligamentos ou contratação.

Funcionalidade 1: Gerenciar Urgência:

a. No menu, clique em "Gerenciar Urgência".
b. Expanda o projeto cuja urgência você deseja alterar.
c. Clique no botão "Alterar Urgência".
d. Na janela que aparecer, selecione a nova urgência (Baixa, Média ou Alta).
e. Clique em "Salvar".

Funcionalidade 2: Gerenciar Usuários:

a. No menu, clique em "Gerenciar Usuários".
b. Para adicionar um novo usuário, preencha o formulário na parte superior da tela:
   i. Nome de Usuário: O login do novo usuário.
   ii. E-mail: O endereço de e-mail para onde as credenciais será enviado.
   iii. Perfil: Escolha entre **Solicitante**, **Colaborador** ou **Administrador**.
c. Clique em "Adicionar Usuário". O sistema criará o usuário e enviará um e-mail com a senha padrão.

c. Dúvidas Frequentes:

i. Esqueci minha senha. O que eu faço?
Entre em contato com um administrador do sistema. Atualmente, não há uma função de recuperação de senha automática.

ii. Cadastrei um projeto com informações erradas. Como corrijo?
No momento, não é possível editar os dados de um projeto após o cadastro. Peça a um administrador para fazer a correção diretamente na planilha de dados.

iii. Como saio do sistema?
Clique no ícone de porta no canto superior direito da tela.

10. Minutas de Reuniões e Workshops: (fase 2 até aqui)(tem relatórios na fase 3)

Ata de Reunião:

Empresa: RUBY Alignment

Data da Reunião: 06/09/2025

Horário de Início: 08:00 horas

Modalidade da Reunião: [ ] Presencial [x] Online

Horário de Término: 11:00 horas

Resumo da reunião:

Na reunião realizada no dia **06/09/2025**, a equipe deu continuidade às atividades de desenvolvimento do projeto. Foram abordados os seguintes pontos:

- Discussão sobre a **fase 2 do projeto de Banco de Dados**, com foco na evolução do modelo e nas etapas subsequentes do desenvolvimento.
- Elaboração do **Diagrama de Entidade-Relacionamento (DER)**, consolidando a modelagem dos dados e suas inter-relações.
- Construção do **Dicionário de Dados**, definindo de maneira clara os atributos, domínios e significados das informações utilizadas no sistema.
- Desenvolvimento dos **Casos de Uso**, descrevendo as interações entre os atores e o sistema, visando a garantir uma visão estruturada dos requisitos levantados.
- Organização e registro da **documentação de requisitos funcionais e não funcionais**, consolidando as necessidades do sistema e as restrições técnicas a serem consideradas.

Essas atividades tiveram como objetivo alinhar a equipe quanto à modelagem do sistema e fornece uma base sólida para as próximas etapas do projeto.

Data da Reunião: 10/09/2025

Modalidade da Reunião: [ ] Presencial [x] Online

Horário de Início: 23:00 horas

Horário de Término: :00 hora

- Aprovação Formal dos Requisitos.
- Definição do banco de dados a ser utilizado.

Data da Reunião: 11/09/2025 Modalidade da Reunião: [x] Presencial [] Online

Horário de Início: 08:30 horas Horário de Término: 11:10 ho

- **Desenvolvimento de Diagramas DER no MySQL Workbench**:

A equipe iniciou a elaboração dos diagramas de entidade-relacionamento (DER) utilizando o MySQL Workbench, uma ferramenta essencial para o mapeamento e estruturação do banco de dados da empresa. Durante o processo, foram discutidas as principais entidades do sistema, suas inter-relações e os fluxos de informações necessários para garantir que o modelo de dados fosse robusto e eficiente. A colaboração entre os membros da equipe foi ativa, permitindo que todas as possíveis interações entre as entidades fossem adequadamente representadas e visualizadas na ferramenta.

- Criação **do Logotipo da Empresa:**

Um dos principais focos da reunião foi o design do logotipo da RUBY Alignment. A equipe discutiu conceitos de identidade visual, alinhando a estética da marca aos valores e objetivos da empresa. A proposta de logo foi revisada e ajustada em conjunto, levando em consideração as sugestões de cada membro da equipe. O logotipo será uma representação gráfica da empresa, destacando seus principais pilares e garantindo que a imagem da marca seja marcante e facilmente reconhecível no mercado.

11. Modelo Lógico de Dados: Diagramas e documentação detalhada de tabelas, colunas, chaves primarias e estrangeiras:

```
CREATE DATABASE GestaoDemandas;
use GestaoDemandas;
CREATE TABLE Solicitante (
idSolicitante INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(45) NOT NULL,
email VARCHAR(30) NOT NULL UNIQUE,
telefone VARCHAR(15),
cpf VARCHAR(14) NOT NULL UNIQUE,
dataCadastro DATE NOT NULL,
senha VARCHAR(255) NOT NULL
);
CREATE TABLE Departamento (
idDepartamento INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(40) NOT NULL UNIQUE,
descricao TEXT
);
CREATE TABLE Cargo (
idCargo INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(20) NOT NULL UNIQUE,
descricao TEXT
);
CREATE TABLE StatusColaborador (
idStatusColaborador INT AUTO_INCREMENT PRIMARY KEY,
status VARCHAR(15) NOT NULL UNIQUE -- 'ativo', 'inativo'
);
CREATE TABLE Colaborador (
idColaborador INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nome VARCHAR(40) NOT NULL,
    email VARCHAR(35) NOT NULL UNIQUE,
    telefone VARCHAR(15),
    dataAdmissao DATE NOT NULL,
    idCargo INT NOT NULL,
    idDepartamento INT NOT NULL,
    idStatusColaborador INT NOT NULL,

    CONSTRAINT fk_Colaborador_Cargo
      FOREIGN KEY (idCargo) REFERENCES Cargo(idCargo)
      ON DELETE RESTRICT
      ON UPDATE CASCADE,

    CONSTRAINT fk_Colaborador_Departamento
      FOREIGN KEY (idDepartamento) REFERENCES Departamento(idDepartamento)
      ON DELETE RESTRICT
      ON UPDATE CASCADE,

    CONSTRAINT fk_Colaborador_Status
      FOREIGN KEY (idStatusColaborador) REFERENCES
    StatusColaborador(idStatusColaborador)
      ON DELETE RESTRICT
      ON UPDATE CASCADE
    );
    CREATE TABLE HardSkill (
    idHardSkill INT AUTO_INCREMENT PRIMARY KEY,
    idColaborador INT NOT NULL,
    habilidade VARCHAR(100) NOT NULL,

    CONSTRAINT fk_HardSkill_Colaborador
      FOREIGN KEY (idColaborador) REFERENCES Colaborador(idColaborador)
      ON DELETE CASCADE
      ON UPDATE CASCADE
    );
    CREATE TABLE StatusDemanda (
    idStatusDemanda INT AUTO_INCREMENT PRIMARY KEY,
    status VARCHAR(15) NOT NULL UNIQUE -- 'Aberta', 'Em andamento', etc.
    );
    CREATE TABLE PrioridadeDemanda (
    idPrioridadeDemanda INT AUTO_INCREMENT PRIMARY KEY,
    prioridade VARCHAR(10) NOT NULL UNIQUE -- 'Baixa', 'Média', etc.
    );
    CREATE TABLE Demandas (
    idDemandas INT AUTO_INCREMENT PRIMARY KEY,
    titulo VARCHAR(100) NOT NULL,
    descricao TEXT NOT NULL,
    dataAbertura DATETIME NOT NULL,
    dataLimite DATE,
    dataConclusao DATETIME,
    idSolicitante INT NOT NULL,
```

```
    idColaborador INT NOT NULL,
    idStatusDemanda INT NOT NULL,
    idPrioridadeDemanda INT NOT NULL,

    CONSTRAINT fk_Demandas_Solicitante
      FOREIGN KEY (idSolicitante) REFERENCES Solicitante(idSolicitante)
      ON DELETE CASCADE
      ON UPDATE CASCADE,

    CONSTRAINT fk_Demandas_Colaborador
      FOREIGN KEY (idColaborador) REFERENCES Colaborador(idColaborador)
      ON DELETE CASCADE
      ON UPDATE CASCADE,

    CONSTRAINT fk_Demandas_Status
      FOREIGN KEY (idStatusDemanda) REFERENCES StatusDemanda(idStatusDemanda)
      ON DELETE RESTRICT
      ON UPDATE CASCADE,

    CONSTRAINT fk_Demandas_Prioridade
      FOREIGN KEY (idPrioridadeDemanda) REFERENCES
    PrioridadeDemanda(idPrioridadeDemanda)
      ON DELETE RESTRICT
      ON UPDATE CASCADE
    );
    CREATE TABLE NivelAcesso (
    idNivelAcesso INT AUTO_INCREMENT PRIMARY KEY,
    nivel VARCHAR(15) NOT NULL UNIQUE -- 'super admin', 'admin comum'
    );
    CREATE TABLE ADM (
    idADM INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(60) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL,
    dataCadastro DATETIME NOT NULL,
    idColaborador INT NOT NULL,
    idNivelAcesso INT NOT NULL,


    CONSTRAINT fk_ADM_Colaborador
      FOREIGN KEY (idColaborador) REFERENCES Colaborador(idColaborador)
      ON DELETE CASCADE
      ON UPDATE CASCADE,

    CONSTRAINT fk_ADM_NivelAcesso
      FOREIGN KEY (idNivelAcesso) REFERENCES NivelAcesso(idNivelAcesso)
      ON DELETE RESTRICT
      ON UPDATE CASCADE
);
```

12. Regras de Negócios:

1. Cadastro de Solicitantes

1.1 Unicidade de CPF e E-mail

Cada solicitante deve possuir CPF único no sistema

E-mail deve ser único para cada solicitante

Cadastro é bloqueado se CPF ou e-mail já existirem

1.2 Validação de Dados Obrigatórios

Nome completo é obrigatório (mínimo 2 caracteres)

CPF deve ser válido e formatado (XXX.XXX.XXX-XX)

Data de cadastro é automática (data atual)

Senha deve ter mínimo de 8 caracteres (validação no backend)

1.3 Status de Conta

Conta fica ativa imediatamente após cadastro

Não há processo de ativação por e-mail

2. Gestão de Colaboradores

2.1 Vínculo com Departamento e Cargo

Todo colaborador deve estar vinculado a um departamento existente

Todo colaborador deve possuir um cargo definido

Não é possível excluir departamento/cargo com colaboradores ativos

2.2 Status do Colaborador

Colaborador pode estar "ativo" ou "inativo"

Colaboradores inativos não podem receber novas demandas

Demandas existentes de colaboradores inativos mantêm o vínculo

2.3 Data de Admissão

Data de admissão não pode ser futura

Data deve ser igual ou anterior à data atual

3. Cadastro de Demandas

3.1 Dados Obrigatórios

Título é obrigatório (máximo 100 caracteres)

Descrição detalhada é obrigatória

Data de abertura é automática (data/hora atual)

Deve ter solicitante válido

Deve ter colaborador atribuído

3.2 Atribuição de Colaboradores

Só podem ser atribuídos colaboradores com status "ativo"

Sistema pode sugerir colaboradores baseado em hardskills

Administradores podem reassignar demandas entre colaboradores

3.3 Prazos e Datas

Data limite é opcional mas não pode ser anterior à data de abertura

- Data de conclusão deve ser igual ou posterior à data de abertura
- Sistema alerta sobre demandas próximas do prazo

4. Status e Prioridades de Demandas

4.1 Fluxo de Status

- Status possíveis: "Aberta", "Em andamento", "Concluída", "Cancelada"
- Mudanças de status são registradas com data/hora
- Demanda concluída deve ter data de conclusão preenchida

4.2 Hierarquia de Prioridades

- Prioridades: "Baixa", "Média", "Alta", "Urgente"
- Demandas com prioridade mais alta têm precedência
- Sistema pode reordenar fila por prioridade e data

5. Gestão de Hardskills

5.1 Vinculação com Colaboradores

- Hardskills são vinculadas a colaboradores específicos
- Mesma habilidade pode existir para múltiplos colaboradores
- Exclusão do colaborador exclui suas hardskills automaticamente

5.2 Busca por Competências

- Sistema permite buscar colaboradores por hardskills específicas
- Sugere colaboradores mais qualificados para demandas técnicas

6. Administração do Sistema

6.1 Níveis de Acesso

- "super admin": Acesso total ao sistema
- "admin comum": Gestão de demandas e colaboradores
- Todo ADM deve estar vinculado a um colaborador existente

6.2 Permissões por Nível

- Super admin pode gerenciar usuários ADM e níveis de acesso
- Admin comum não pode alterar estrutura de departamentos/cargos
- Ambos podem reassignar demandas entre colaboradores

7. Regras de Exclusão e Integridade

7.1 Exclusão em Cascata

- Excluir solicitante exclui suas demandas
- Excluir colaborador exclui suas hardskills e usuário ADM vinculado
- Excluir demanda não afeta solicitante ou colaborador

7.2 Restrições de Exclusão

- Não é possível excluir departamento com colaboradores ativos
- Não é possível excluir cargo vinculado a colaboradores
- Status e prioridades não podem ser excluídos se usados em demandas

8. Validações de Datas

8.1 Consistência Temporal

- Data de cadastro não pode ser futura
- Data de admissão não pode ser futura
- Data limite não pode ser anterior à data de abertura
- Data de conclusão deve ser >= data de abertura

9. Regras de Negócio Específicas

9.1 Limite de Demandas por Colaborador

Colaborador pode ter no máximo 10 demandas "Em andamento" simultaneamente

Sistema bloqueia novas atribuições acima do limite

9.2 Notificações Automáticas

Alerta para demandas próximas do prazo (3 dias de antecedência)

Notificação quando demanda fica atrasada

Aviso ao completar 80% do prazo total

9.3 Métricas e Relatórios

Tempo médio de resolução por departamento

Taxa de conclusão dentro do prazo

Volume de demandas por prioridade

13. Views:

1. VIEW: vw_DemandasDetalhadas - Mostrar todas as demandas com informações completas em uma única consulta

```
CREATE VIEW vw_DemandasDetalhadas AS
SELECT
   d.idDemandas,
   d.titulo,
   d.descricao,
   d.dataAbertura,
   d.dataLimite,
   d.dataConclusao,
   -- Informações do Solicitante
   s.idSolicitante,
   s.nome AS nome_solicitante,
   s.email AS email_solicitante,
   s.telefone AS telefone_solicitante,
   -- Informações do Colaborador
   c.idColaborador,
   c.nome AS nome_colaborador,
   c.email AS email_colaborador,
   -- Informações do Departamento
   dep.idDepartamento,
   dep.nome AS nome_departamento,
   -- Status e Prioridade
   sd.status AS status_demanda,
   pd.prioridade,
   -- Cálculos
   DATEDIFF(d.dataLimite, CURDATE()) AS dias_restantes,
   CASE
      WHEN d.dataConclusao IS NOT NULL THEN 'Concluída'
      WHEN d.dataLimite < CURDATE() THEN 'Atrasada'
      WHEN DATEDIFF(d.dataLimite, CURDATE()) <= 3 THEN
'Próximo do prazo'
      ELSE 'No prazo'
   END AS situacao_prazo
```

```
FROM Demandas d
INNER JOIN Solicitante s ON d.idSolicitante = s.idSolicitante
INNER JOIN Colaborador c ON d.idColaborador = c.idColaborador
INNER JOIN Departamento dep ON c.idDepartamento = dep.idDepartamento
INNER JOIN StatusDemanda sd ON d.idStatusDemanda = sd.idStatusDemanda
INNER JOIN PrioridadeDemanda pd ON d.idPrioridadeDemanda = pd.idPrioridadeDemanda;
```

2. VIEW: vw_ColaboradoresAtivosComSkills - Listar colaboradores ativos com suas habilidades agrupadas

```
CREATE VIEW vw_ColaboradoresAtivosComSkills AS
SELECT
   c.idColaborador,
   c.nome,
   c.email,
   c.telefone,
   c.dataAdmissao,
   -- Informações do Cargo
   car.idCargo,
   car.nome AS cargo,
   -- Informações do Departamento
   dep.idDepartamento,
   dep.nome AS departamento,
   -- Habilidades agrupadas
   COALESCE(GROUP_CONCAT(hs.habilidade ORDER BY hs.habilidade SEPARATOR ', '), 'Sem habilidades') AS habilidades,
   COUNT(hs.idHardSkill) AS total_habilidades,
   -- Demandas em aberto
   (SELECT COUNT(*) FROM Demandas d
    WHERE d.idColaborador = c.idColaborador
    AND d.idStatusDemanda IN (1, 2)) AS demandas_ativas
FROM Colaborador c
INNER JOIN Cargo car ON c.idCargo = car.idCargo
INNER JOIN Departamento dep ON c.idDepartamento = dep.idDepartamento
INNER JOIN StatusColaborador sc ON c.idStatusColaborador = sc.idStatusColaborador
LEFT JOIN HardSkill hs ON c.idColaborador = hs.idColaborador
WHERE sc.status = 'ativo'
GROUP BY c.idColaborador, c.nome, c.email, c.telefone, car.nome, dep.nome;
```

14. Esquema Físico do Banco de Dados:

```
CREATE DATABASE GestaoDemandas;
use GestaoDemandas;
CREATE TABLE Solicitante (
idSolicitante  INT AUTO_INCREMENT PRIMARY KEY,
```

```
nome VARCHAR(45) NOT NULL,
email VARCHAR(30) NOT NULL UNIQUE,
telefone VARCHAR(15),
cpf VARCHAR(14) NOT NULL UNIQUE,
dataCadastro DATE NOT NULL,
senha VARCHAR(255) NOT NULL
);
CREATE TABLE Departamento (
idDepartamento INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(40) NOT NULL UNIQUE,
descricao TEXT
);
CREATE TABLE Cargo (
idCargo INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(20) NOT NULL UNIQUE,
descricao TEXT
);
CREATE TABLE StatusColaborador (
idStatusColaborador INT AUTO_INCREMENT PRIMARY KEY,
status VARCHAR(15) NOT NULL UNIQUE -- 'ativo', 'inativo'
);
CREATE TABLE Colaborador (
idColaborador INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(40) NOT NULL,
email VARCHAR(35) NOT NULL UNIQUE,
telefone VARCHAR(15),
dataAdmissao DATE NOT NULL,
idCargo INT NOT NULL,
idDepartamento INT NOT NULL,
idStatusColaborador INT NOT NULL,

CONSTRAINT fk_Colaborador_Cargo
  FOREIGN KEY (idCargo) REFERENCES Cargo(idCargo)
  ON DELETE RESTRICT
  ON UPDATE CASCADE,

CONSTRAINT fk_Colaborador_Departamento
  FOREIGN KEY (idDepartamento) REFERENCES
Departamento(idDepartamento)
  ON DELETE RESTRICT
  ON UPDATE CASCADE,

CONSTRAINT fk_Colaborador_Status
  FOREIGN KEY (idStatusColaborador) REFERENCES
StatusColaborador(idStatusColaborador)
  ON DELETE RESTRICT
  ON UPDATE CASCADE
);
CREATE TABLE HardSkill (
idHardSkill INT AUTO_INCREMENT PRIMARY KEY,
```

```
idColaborador INT NOT NULL,
habilidade VARCHAR(100) NOT NULL,

CONSTRAINT fk_HardSkill_Colaborador
  FOREIGN KEY (idColaborador) REFERENCES
Colaborador(idColaborador)
  ON DELETE CASCADE
  ON UPDATE CASCADE
);
CREATE TABLE StatusDemanda (
idStatusDemanda INT AUTO_INCREMENT PRIMARY KEY,
status VARCHAR(15) NOT NULL UNIQUE -- 'Aberta', 'Em
andamento', etc.
);
CREATE TABLE PrioridadeDemanda (
idPrioridadeDemanda INT AUTO_INCREMENT PRIMARY KEY,
prioridade VARCHAR(10) NOT NULL UNIQUE -- 'Baixa', 'Média',
etc.
);
CREATE TABLE Demandas (
idDemandas INT AUTO_INCREMENT PRIMARY KEY,
titulo VARCHAR(100) NOT NULL,
descricao TEXT NOT NULL,
dataAbertura DATETIME NOT NULL,
dataLimite DATE,
dataConclusao DATETIME,
idSolicitante INT NOT NULL,
idColaborador INT NOT NULL,
idStatusDemanda INT NOT NULL,
idPrioridadeDemanda INT NOT NULL,

CONSTRAINT fk_Demandas_Solicitante
  FOREIGN KEY (idSolicitante) REFERENCES
Solicitante(idSolicitante)
  ON DELETE CASCADE
  ON UPDATE CASCADE,

CONSTRAINT fk_Demandas_Colaborador
  FOREIGN KEY (idColaborador) REFERENCES
Colaborador(idColaborador)
  ON DELETE CASCADE
  ON UPDATE CASCADE,

CONSTRAINT fk_Demandas_Status
  FOREIGN KEY (idStatusDemanda) REFERENCES
StatusDemanda(idStatusDemanda)
  ON DELETE RESTRICT
  ON UPDATE CASCADE,

CONSTRAINT fk_Demandas_Prioridade
```

```
    FOREIGN KEY (idPrioridadeDemanda) REFERENCES
PrioridadeDemanda(idPrioridadeDemanda)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);
CREATE TABLE NivelAcesso (
idNivelAcesso INT AUTO_INCREMENT PRIMARY KEY,
nivel VARCHAR(15) NOT NULL UNIQUE -- 'super admin', 'admin
comum'
);
CREATE TABLE ADM (
idADM INT AUTO_INCREMENT PRIMARY KEY,
nome VARCHAR(60) NOT NULL,
email VARCHAR(100) NOT NULL UNIQUE,
senha VARCHAR(255) NOT NULL,
dataCadastro DATETIME NOT NULL,
idColaborador INT NOT NULL,
idNivelAcesso INT NOT NULL,


CONSTRAINT fk_ADM_Colaborador
    FOREIGN KEY (idColaborador) REFERENCES
Colaborador(idColaborador)
    ON DELETE CASCADE
    ON UPDATE CASCADE,

CONSTRAINT fk_ADM_NivelAcesso
    FOREIGN KEY (idNivelAcesso) REFERENCES
NivelAcesso(idNivelAcesso)
    ON DELETE RESTRICT
    ON UPDATE CASCADE
);
```

15. Triggers:

1 - Trigger: Impedir Colaborador Inativo de Receber Demandas:

```
DELIMITER $$
CREATE TRIGGER tg_valida_colaborador_ativo
BEFORE INSERT ON Demandas
FOR EACH ROW
BEGIN
  DECLARE status_colaborador VARCHAR(15);

  -- Busca status do colaborador
  SELECT sc.status INTO status_colaborador
  FROM Colaborador c
  INNER JOIN StatusColaborador sc ON c.idStatusColaborador =
sc.idStatusColaborador
  WHERE c.idColaborador = NEW.idColaborador;
```

```
   IF status_colaborador = 'inativo' THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Não é possível atribuir
demanda a colaborador inativo.';
   END IF;
END$$
DELIMITER ;
```

2 - Trigger: Impedir Exclusão de Departamento com Colaboradores Ativos:

```
DELIMITER $$
CREATE TRIGGER tg_valida_exclusao_departamento
BEFORE DELETE ON Departamento
FOR EACH ROW
BEGIN
   DECLARE colaboradores_ativos INT;

   SELECT COUNT(*) INTO colaboradores_ativos
   FROM Colaborador c
   INNER JOIN StatusColaborador sc ON c.idStatusColaborador =
sc.idStatusColaborador
   WHERE c.idDepartamento = OLD.idDepartamento
   AND sc.status = 'ativo';

   IF colaboradores_ativos > 0 THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Não é possível excluir
departamento com colaboradores ativos.';
   END IF;
END$$
DELIMITER ;
```

16. Configurações de Performance:

```
SET GLOBAL innodb_buffer_pool_size = 256 * 1024 * 1024;
SET GLOBAL innodb_log_file_size = 64 * 1024 * 1024;
SET GLOBAL query_cache_size = 32 * 1024 * 1024;
SET GLOBAL query_cache_type = 1;

SET GLOBAL innodb_flush_log_at_trx_commit = 2;
SET GLOBAL sync_binlog = 0;
SET GLOBAL innodb_flush_method = O_DIRECT;

SET GLOBAL max_connections = 151;
SET GLOBAL thread_cache_size = 16;
SET GLOBAL wait_timeout = 600;
SET GLOBAL interactive_timeout = 600;

SET GLOBAL innodb_file_per_table = 1;
SET GLOBAL innodb_buffer_pool_instances = 1;
```

- Health check


```
{
  "database": "connected",
  "environment": "railway",
  "status": "healthy",
  "timestamp": "2025-10-08T01:37:20.883690"
}
```


perfect-creativity-rubya.up.railway.app/health

17. Plano de Segurança:
    a. Roles e Grupos:
    ROLE_SOLICITANTE - Perfil Solicitante
    ROLE_COLABORADOR - Perfil Colaborador
    ROLE_ADMIN - Perfil Administrador Matriz de Permissões por Perfil:
    b. Regras de Negócio para Permissões:
        i. Perfil Solicitante:

        Permissões:

        - Criar demandas/projetos
        - Acompanhar status das próprias demandas
        - Receber notificações sobre andamento
        - Visualizar histórico próprio

        Funcionalidades restritas:

        - Gerenciar outros usuários
        - Visualizar demandas de outros solicitantes
        - Alterar urgência de projetos
        - Aderir a projetos como colaborador
        - Visualizar histórico próprio

        ii. Perfil Colaborador:

        Permissões:

        - Visualizar demandas disponíveis
        - Aceitar/recusar demandas
        - Atualizar status de projetos atribuídos
        - Visualizar histórico próprio e barra de aderência
        - Aderir a projetos com status "NÃO INICIADO"
        - Gerenciar andamento de projetos em execução

        Funcionalidades restritas:

- Gerenciar outros usuários
- Criar novas demandas
- Gerenciar outros usuários
- Alterar urgência de projetos
- Visualizar relatórios gerais

iii. Perfil Administrador:

Permissões:

- Visualizar e gerenciar todos os colaboradores
- Visualizar e gerenciar todas as demandas
- Reatribuir demandas manualmente
- Gerar relatórios do sistema
- Gerenciar urgência de projetos (Baixa, Média, Alta)
- Adicionar/editar/remover usuários
- Visualizar organograma hierárquico
- Acessar histórico completo do sistema

Acesso total a todas as funcionalidades

iv. Acesso a Dados Pessoais
- Administradores têm acesso a todos os dados cadastrais

v. Gestão de Demandas:
- Solicitantes só visualizam suas próprias demandas
- Colaboradores visualizam demandas atribuídas e disponíveis
- Administradores visualizam todas as demandas

vi. Operações de Sistema:
- Aderência a projetos limitada a colaboradores ativos
- Alteração de status segue fluxo pré-definido por perfil
- Reatribuição manual exclusiva para administradores

18. Testes:

Todos os testes realizados no banco de dados foram executados conforme descrito nesta documentação e cuidadosamente validados para garantir o correto funcionamento do sistema. Para referência visual, um vídeo demonstrando a execução prática dos testes também será disponibilizado via Google Drive. O vídeo apresenta operações como inserções, consultas, atualizações, exclusões, verificação da integridade dos dados, chaves primárias e estrangeiras, bem como o funcionamento de gatilhos (triggers) e demais regras implementadas, servindo como registro adicional da validação do banco de dados.

https://drive.google.com/file/d/1cx8eMWBzGLLdSYE2Q4s3CSGdJv2-9Y_q/view?usp=drive_link

```
use railway;
```

```
-- Listar tabelas
SHOW TABLES;

-- Verificar estrutura das tabela
DESCRIBE ADM;
DESCRIBE Cargo;
DESCRIBE Colaborador;
DESCRIBE Comentarios;
DESCRIBE Demandas;
DESCRIBE Departamento;
DESCRIBE HardSkill;
DESCRIBE Logs;
DESCRIBE NivelAcesso;
DESCRIBE PrioridadeDemanda;
DESCRIBE Solicitante;
DESCRIBE StatusColaborador;
DESCRIBE StatusDemanda;
DESCRIBE vw_ColaboradoresAtivosComSkills;
DESCRIBE vw_DemandasDetalhadas;


-- Tentar inserir um colaborador com um idDepartamento inexistente
INSERT INTO Colaborador (nome, email, telefone, dataAdmissao, idCargo, idDepartamento, idStatusColaborador)
VALUES ('Teste', 'teste@email.com', '11999999999', '2025-10-09', 1, 999, 1);


SELECT * FROM Cargo;
SELECT * FROM Departamento;
SELECT * FROM StatusColaborador;
SELECT * FROM Colaborador;

-- Teste de Inserção Válida
INSERT INTO Colaborador (nome, email, telefone, dataAdmissao, idCargo, idDepartamento, idStatusColaborador)
VALUES ('Maria Silva', 'maria@empresa.com', '11987654321', '2024-05-20', 2, 1, 1);

SELECT * FROM Colaborador;


-- Teste de Atualização
UPDATE Colaborador
SET dataAdmissao = '2025-10-08'
WHERE email = 'maria@empresa.com';
SELECT * FROM Colaborador;

-- Teste de Exclusão com Restrição
DELETE FROM Colaborador
WHERE idColaborador = '26';
```

```
SELECT * FROM Colaborador;

-- Teste de Relacionamento Entre Tabelas
SELECT c.nome AS Colaborador, ca.nome AS Cargo, d.nome AS Departamento
FROM Colaborador c
JOIN Cargo ca ON c.idCargo = ca.idCargo
JOIN Departamento d ON c.idDepartamento = d.idDepartamento;

-- Teste de Views
SELECT * FROM vw_DemandasDetalhadas;
SELECT * FROM vw_ColaboradoresAtivosComSkills;

-- Teste de Consultas Específicas (Filtros e Relatórios)
-- Colaboradores ativos com Hard Skills
SELECT c.nome, h.habilidade
FROM Colaborador c
JOIN HardSkill h ON c.idColaborador = h.idColaborador
WHERE c.idStatusColaborador = 1;

-- Teste de Desempenho Básico
EXPLAIN SELECT * FROM Demandas WHERE idColaborador = 1;

SELECT * FROM Colaborador;
SELECT * FROM Solicitante;

-- Trigger: Validar Email Único entre Solicitante e Colaborador
INSERT INTO Solicitante (nome, email, dataCadastro)
VALUES ('Teste', 'admin@ruby.com', '2025-10-09'); -- ERRO

INSERT INTO Solicitante (nome, email, dataCadastro, idColaborador)
VALUES ('Novo Solicitante', 'novo@ruby.com', '2025-10-09', 20);

SELECT * FROM Solicitante;

INSERT INTO Colaborador (nome, email, telefone, dataAdmissao, idCargo,
idDepartamento, idStatusColaborador)
VALUES ('Teste Colaborador', 'novo@ruby.com', '11999999999', '2025-10-09', 3, 1, 1);
-- ERRO

-- Trigger: Impedir Colaborador Inativo de Receber Demandas
-- Criando um colaborador inativo
INSERT INTO Colaborador (nome, email, telefone, dataAdmissao, idCargo,
idDepartamento, idStatusColaborador)
VALUES ('Inativo', 'inativo@ruby.com', '11900000000', '2025-10-09', 3, 1, 2);

SELECT * FROM Colaborador;
INSERT INTO Demandas (titulo, descricao, dataAbertura, idColaborador, idSolicitante,
idStatusDemanda, idPrioridadeDemanda)
VALUES ('Demanda teste', 'Descrição da demanda teste', '2025-10-09 10:00:00', 24,3,1,
1); -- ERRO
```

```
INSERT INTO Demandas (titulo, descricao, dataAbertura, idColaborador, idSolicitante,
idStatusDemanda, idPrioridadeDemanda)
VALUES ('Demanda para ativo', 'Descrição da demanda para colaborador ativo', '2025-
10-09 10:00:00', 18, 3, 1, 1);



-- Trigger: Impedir Exclusão de Departamento com Colaboradores Ativos
DELETE FROM Departamento
WHERE idDepartamento = 1;

INSERT INTO Departamento (nome)
VALUES ('Departamento Vazio');

DELETE FROM Departamento
WHERE nome = 'Departamento Vazio';
```