

Entendendo o Projeto

O que você precisa fazer?

Criar um sistema de gerenciamento de tarefas (tipo Trello) para uma indústria alimentícia, onde:

- Funcionários cadastram tarefas
 - As tarefas são organizadas em 3 colunas: "A Fazer" → "Fazendo" → "Pronto"
 - Cada tarefa tem prioridade (baixa/média/alta) e pertence a um setor
-

1. BANCO DE DADOS

Você precisa criar 2 tabelas:

Tabela USUARIOS:

- `id` (número automático)
- `nome` (texto)
- `email` (texto único)

Tabela TAREFAS:

- `id` (número automático)
- `id_usuario` (liga com o usuário que criou)
- `descricao` (texto da tarefa)
- `setor` (ex: "Produção", "Qualidade")
- `prioridade` (baixa/média/alta)

- `status` (a fazer/fazendo/pronto)
- `data_cadastro` (data automática)

Relacionamento: 1 usuário pode ter várias tarefas, mas 1 tarefa pertence a apenas 1 usuário.

2. TECNOLOGIAS

Opção 1: Java Moderno (Recomendado)

- **Frontend:** HTML + CSS + JavaScript puro
- **Backend:** Java (Spring Boot)
- **Banco:** MySQL

Opção 2: Spring Boot com Thymeleaf

- **Frontend:** Thymeleaf (template engine integrado)
- **Backend:** Java (Spring Boot)
- **Banco:** MySQL ou SQLite

Opção 3: JavaScript Full Stack

- **Frontend:** HTML + CSS + JavaScript
 - **Backend:** Node.js (JavaScript no servidor)
 - **Banco:** MySQL ou SQLite
-

3. AS TELAS QUE VOCÊ DEVE CRIAR

Tela 1: Cadastro de Usuários

Um formulário com:

- Campo: Nome
- Campo: E-mail (com validação)
- Botão: Cadastrar

Quando clicar em cadastrar:

1. Valida se todos os campos estão preenchidos
 2. Valida se o e-mail é válido
 3. Salva no banco de dados
 4. Mostra mensagem: "Cadastro concluído com sucesso"
-

Tela 2: Cadastro de Tarefas

Um formulário com:

- Campo: Usuário (lista os usuários do banco)
- Campo: Descrição da tarefa
- Campo: Setor
- Campo: Prioridade (baixa/média/alta)

Quando clicar em cadastrar:

1. Valida se tudo está preenchido
2. Salva no banco com status "a fazer" automaticamente
3. Mostra mensagem de sucesso

Tela 3: Gerenciamento (Principal) - A MAIS IMPORTANTE

Dividida em **3 colunas**:

A FAZER	FAZENDO	PRONTO	
[Tarefa 1]	[Tarefa 3]	[Tarefa 5]	
[Tarefa 2]	[Tarefa 4]		

Cada card de tarefa mostra:

- Descrição
- Setor
- Prioridade
- Nome do usuário
- Botões: Editar | Excluir
- Dropdown: mudar status

Funcionalidades:

1. **Ver tarefas:** Carrega do banco e mostra cada uma na coluna correta
 2. **Mudar status:** Seleciona novo status → clica "Atualizar" → tarefa muda de coluna
 3. **Editar:** Clica "Editar" → volta pra tela de cadastro com dados preenchidos → salva atualização
 4. **Excluir:** Clica "Excluir" → confirma → remove do banco
-

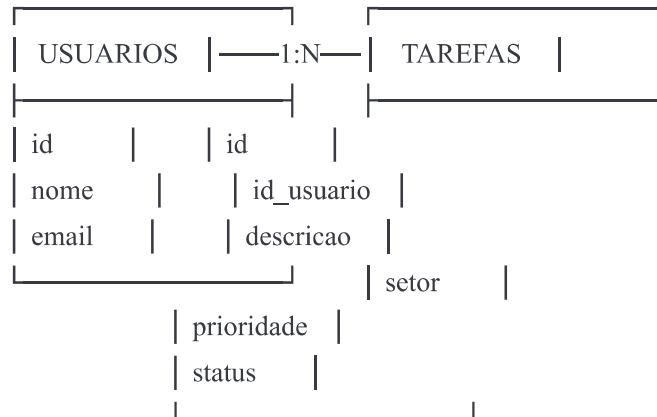
4. COMO FUNCIONA O FLUXO

1. Usuário acessa o sistema
↓
2. Vê a tela de Gerenciamento (3 colunas)
↓
3. Pode ir para:
 - Menu → Cadastrar Usuário
 - Menu → Cadastrar Tarefa
↓
4. Depois de cadastrar, volta pro Gerenciamento
↓
5. Vê as tarefas organizadas por status
↓
6. Pode mover, editar ou excluir tarefas

5. DIAGRAMAS QUE VOCÊ PRECISA FAZER

DER (Diagrama Entidade-Relacionamento):

Desenha as tabelas, seus campos e a ligação entre elas.



Caso de Uso:

Desenha os atores (usuário) e ações (cadastrar, editar, excluir, etc).

6. PONTOS DE ATENÇÃO

- **Todos os campos são obrigatórios** (valide isso!)
- **E-mail precisa ser válido** (use regex ou validação HTML5)
- **Sempre mostre mensagens** de sucesso/erro
- **Use as cores especificadas:** ☒ #0056b3, ☐ #FFFFFF, ☒ #000000
- **Fonte:** Segoe UI
- **Menu principal** para navegar entre telas

7. QUAL TECNOLOGIA ESCOLHER?

- **Se você quer usar Java:** Use Spring Boot + MySQL

- **Se você sabe mais JavaScript:** Use Node.js + MySQL
 - **Para banco de dados:** MySQL é mais recomendado (SQLite é mais limitado)
-

8. ESTRUTURA DE ARQUIVOS RECOMENDADA

Para Java com Spring Boot:

```
projeto/
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   ├── com/empresa/gerenciamento/
│   │   │   │   ├── GerenciamentoApplication.java
│   │   │   │   ├── model/
│   │   │   │   │   ├── Usuario.java
│   │   │   │   │   └── Tarefa.java
│   │   │   │   ├── repository/
│   │   │   │   │   ├── UsuarioRepository.java
│   │   │   │   │   └── TarefaRepository.java
│   │   │   │   ├── service/
│   │   │   │   │   ├── UsuarioService.java
│   │   │   │   │   └── TarefaService.java
│   │   │   │   └── controller/
│   │   │   │       ├── UsuarioController.java
│   │   │   │       └── TarefaController.java
│   │   └── resources/
│   │       ├── application.properties
│   │       ├── static/
│   │       │   ├── css/
│   │       │   └── style.css
```

```
| | | | └─ js/
| | | |   └─ script.js
| | | └─ templates/
| | └─ cadastro-usuario.html
| └─ cadastro-tarefa.html
└─ gerenciamento.html
database.sql
pom.xml
```

Para JavaScript/Node.js:

```
projeto/
├─ backend/
|   ├── server.js
|   ├── routes/
|   |   ├── usuarios.js
|   |   └─ tarefas.js
|   └─ controllers/
|       └─ database.js
├─ frontend/
|   ├── index.html
|   ├── cadastro-usuario.html
|   ├── cadastro-tarefa.html
|   ├── css/
|   |   └─ style.css
|   ├── js/
|   |   └─ app.js
└─ database.sql
```

9. CONCEITOS IMPORTANTES

Backend (Java ou JavaScript):

- Recebe requisições do frontend
- Faz validações
- Conecta com o banco de dados
- Retorna dados para o frontend

Frontend (HTML + CSS + JS):

- Interface que o usuário vê
- Formulários de cadastro
- Exibe as tarefas organizadas
- Envia requisições para o backend

Banco de Dados (MySQL):

- Armazena todos os dados
- Usuários e tarefas
- Mantém integridade dos relacionamentos

Dúvidas Frequentes**Posso usar outras tecnologias?**

- Sim, desde que atenda aos requisitos do projeto

Preciso fazer login/autenticação?

- Não, o documento diz que não é necessário controle de acesso

Como organizar as tarefas em colunas?

- Use CSS Grid ou Flexbox para dividir em 3 colunas
- Filtre as tarefas por status usando JavaScript

O que entregar?

1. Diagrama ER (imagem JPG)
2. Script SQL do banco de dados
3. Diagrama de Caso de Uso (imagem JPG)
4. Código fonte completo (pasta compactada)