

# Relatório Técnico – Sistema CRUD TechManage Solutions

## 1. Introdução e Justificativa

A TechManage Solutions enfrenta dificuldades no controle manual de informações internas, gerando inconsistências, perda de dados e baixa produtividade. Este projeto propõe uma aplicação web para automatizar o gerenciamento de usuários, projetos e tarefas, proporcionando eficiência e confiabilidade no registro de dados.

---

## 2. Objetivos do Sistema

O sistema tem como principais objetivos:

- Automatizar o controle de usuários, projetos e tarefas.
  - Garantir consistência e integridade dos dados armazenados.
  - Fornecer uma interface simples para cadastro, atualização, listagem e exclusão de registros.
  - Permitir testes de funcionalidades via API RESTful e interface web básica.
- 

## 3. Modelagem Conceitual e Lógica (MER e DER)

Entidades:

### 1. Usuários (Users)

- id (PK)
- nome
- email
- senha
- perfil

### 2. Projetos (Projects)

- id (PK)
- titulo
- descricao
- data\_inicio
- data\_fim
- id\_usuario (FK → Users.id)

### 3. Tarefas (Tasks)

- id (PK)
- titulo
- status

- prioridade
- id\_projeto (FK → Projects.id)

#### Relacionamentos:

- 1:N entre **Usuários** e **Projetos**.
  - 1:N entre **Projetos** e **Tarefas**.
- 

#### 4. Descrição das Entidades e Relacionamentos

- **Usuários:** representam colaboradores que podem criar projetos.
  - **Projetos:** vinculados a um usuário, contêm tarefas.
  - **Tarefas:** vinculadas a um projeto, com informações de status e prioridade.
- 

#### 5. Descrição Técnica das Tecnologias Utilizadas

- **Node.js:** ambiente de execução do servidor.
  - **Express.js:** framework para gerenciamento de rotas e middlewares.
  - **Sequelize ORM:** abstração para manipulação do banco MySQL.
  - **MySQL:** sistema gerenciador de banco de dados.
  - **Nodemon:** reinicia o servidor automaticamente durante o desenvolvimento.
  - **Frontend simples (HTML/CSS/JS):** interface básica para cadastro e listagem de dados.
  - **CORS:** habilitado para permitir comunicação entre frontend e backend local.
  - **.env:** armazenamento seguro de variáveis de ambiente.
- 

#### 6. Evidências de Execução

##### Exemplo de Cadastro de Usuário:

POST /users

Body: { "nome": "Carlos", "email": "carlos@mail.com", "senha": "123456", "perfil": "Admin" }

Response: { "success": true, "data": { "id": 1, "nome": "Carlos", "email": "carlos@mail.com", "perfil": "Admin" } }

##### Listagem de Projetos:

GET /projects

Response: { "success": true, "data": [ { "id": 1, "titulo": "Projeto X", "id\_usuario": 1 } ] }

##### Interface Web:

- Cadastro de usuários, projetos e tarefas via formulários HTML.
- Botões para listar registros carregam os dados da API em listas no frontend.

Inclua

CRUD TechManage

prints:

Usuários

Nome:

Carlos José

Email:

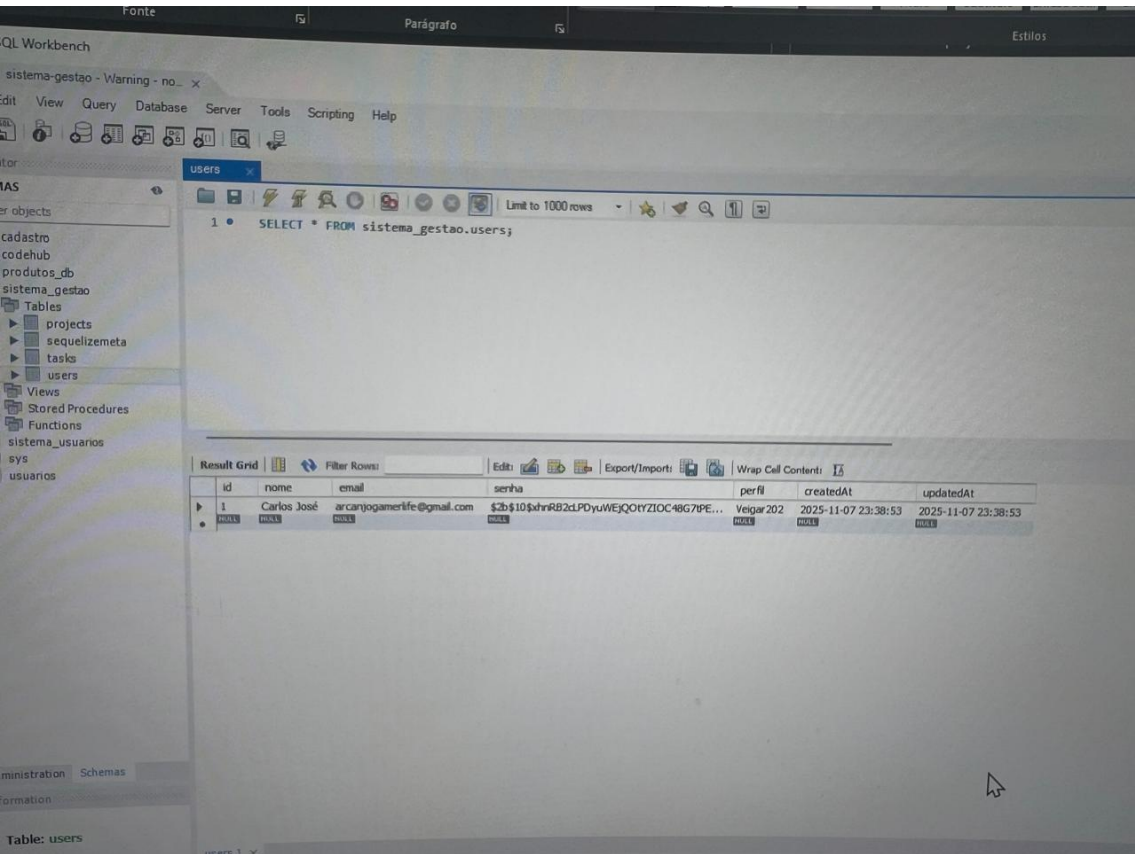
arcanjogamerlife@gmail.com

Perfil:

Veigar202

Criar Usuário

Listar Usuários



Projetos

Título:

Api-Fazenda

Descrição:

Farm de vacas

Data Início:

13/11/2025

Data Fim:

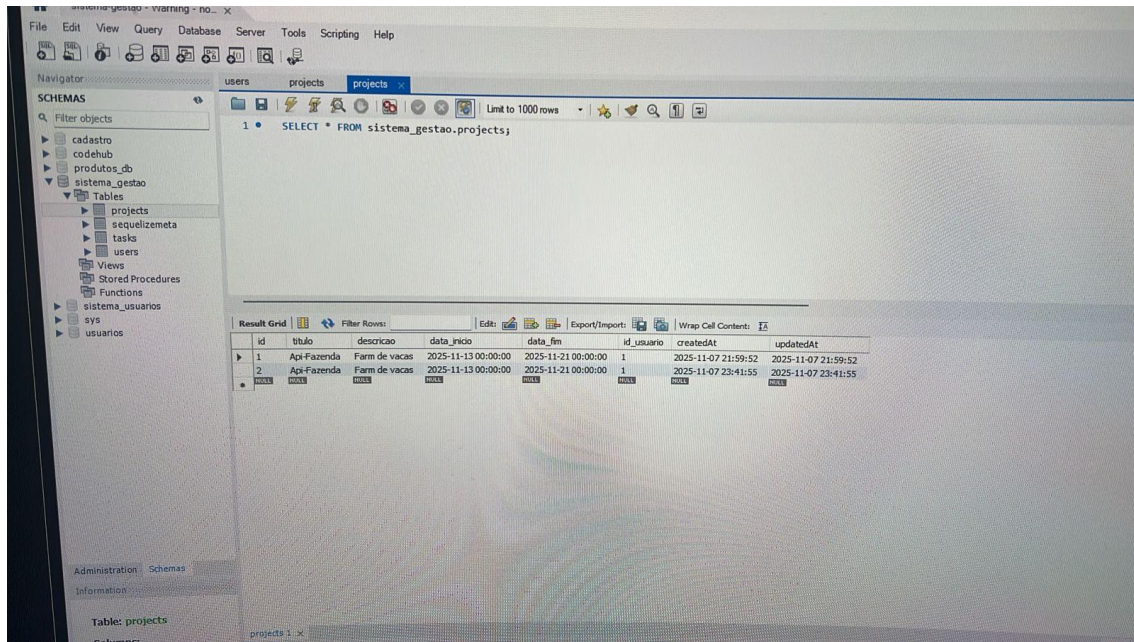
21/11/2025

ID Usuário:

1

Criar Projeto

Listar Projetos



#### Tarefas

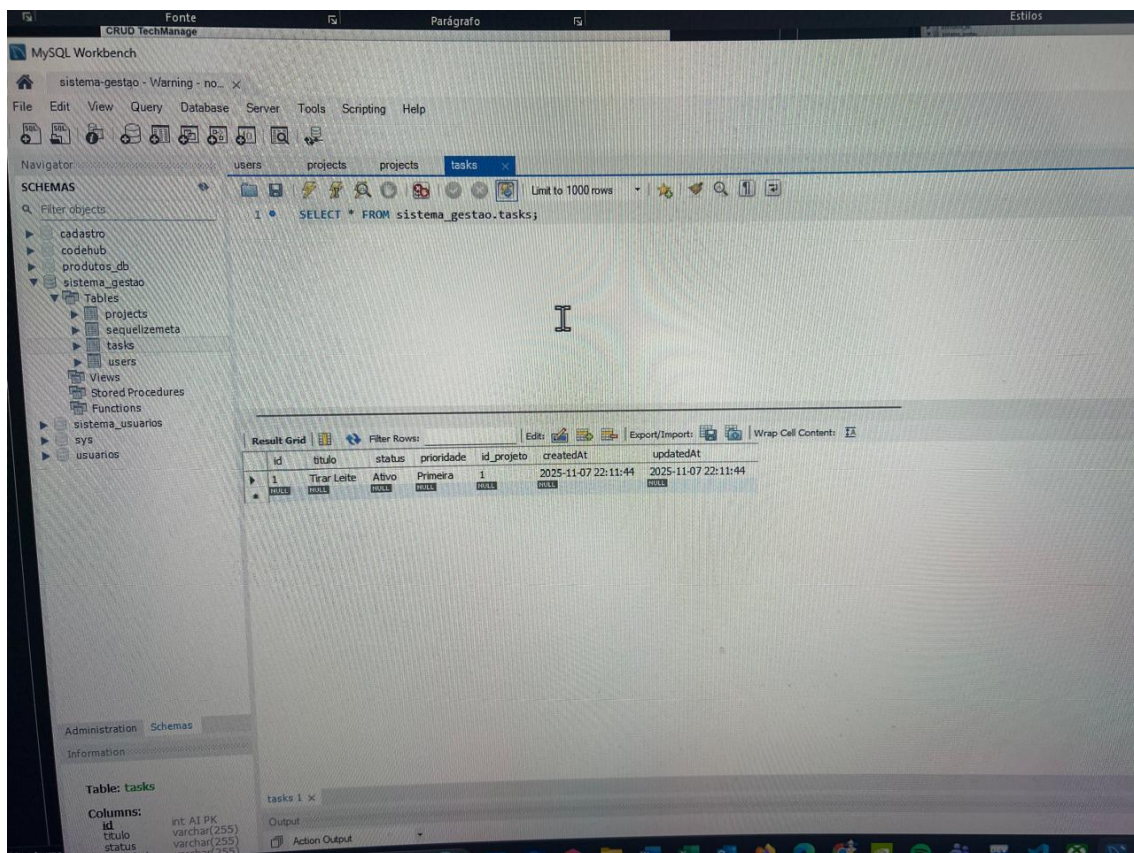
Form for creating a task (Tarefas):

Título:

Status:

Prioridade:

ID Projeto:



## 7. Repositório no GitHub

- Estrutura do projeto:

sistema-gestao/

```
|
|├─ src/
| |├─ controllers/
| | |├─ userController.js
| | |├─ projectController.js
| | |└─ taskController.js
| |├─ models/
| |├─ routes/
| |└─ server.js
|
|├─ frontend/
| |├─ index.html
| |├─ style.css
| |└─ script.js
|
|├─ .env.example
|├─ package.json
|└─ README.md
```

- **README.md:** contém instruções de instalação e uso do sistema.