Projeto UniPet

Sistema CRUD de Gerenciamento de Pets

@ Descrição Geral

O **UniPet** é um sistema de gerenciamento de **PetShop**, com foco em aprendizado prático de **backend** utilizando **Node.js**, **Express** e **JSON-Server**. Trata-se de um projeto que implementa as funcionalidades básicas de um sistema **CRUD** (**Create**, **Read**, **Update**, **Delete**), sendo ideal para fins didáticos e acadêmicos.

→ Funcionalidades

- 🛓 Cadastro de pets com dados como nome, espécie, raça, idade, tutor e ação necessária.
- Q Consulta de registros salvos em uma listagem clara e interativa.
- \ Edição de dados de um pet previamente cadastrado.
- W Remoção de registros do sistema.
- ⊕ Interface web dinâmica utilizando a template engine EJS.

X Tecnologias Utilizadas

- Node.js Plataforma para execução de JavaScript no servidor.
- Express Framework para criação de rotas e gerenciamento de requisições.
- JSON-Server Simulador de banco de dados REST baseado em arguivos .json.
- EJS Template engine para construção dinâmica de páginas HTML.
- Nodemon Utilitário para reinicialização automática do servidor em desenvolvimento.

📌 Pré-requisitos

- Node.js (versão 14.x ou superior)
- npm (gerenciador de pacotes do Node.js)
- Editor de código (VS Code recomendado)
- Terminal ou shell para execução dos scripts

Instalação e Execução

- 1. Clone o repositório do GitHub:
- 2. git clone https://github.com/Leandroooh/UniPet.git
- 3. cd UniPet
- 4. Instale as dependências do projeto:
- 5. npm install
- 6. Inicie o JSON-Server em um terminal:
- 7. npm run json-server
- 8. Em outro terminal, execute o servidor principal:
- 9. npm run dev
- 10. Acesse no navegador: http://localhost:3030

•

Estrutura dos Dados (JSON)

O arquivo db.json simula um banco de dados contendo os registros dos pets:

```
{
  "pets": [
    {
      "id": "33f6",
      "name": "Amora",
      "gender": "Gato",
      "breed": "Sphynx",
      "age": "2",
      "guardian": "Leandro",
      "action": "Exames de Rotina"
    }
]
```

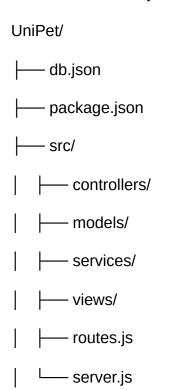
Cada pet possui os seguintes campos:

- id
- name
- gender
- breed
- age
- guardian
- action

Rotas Disponíveis (API REST)

- GET /pets Lista todos os pets
- GET /pets/:id Exibe os dados de um pet específico
- POST /pets Cria um novo pet
- PUT /pets/:id Atualiza totalmente os dados de um pet
- PATCH /pets/:id Atualiza parcialmente um pet
- DELETE /pets/:id Remove um pet pelo ID

Estrutura do Projeto



- controllers/: Lógica de negócio (ex: criação, edição, exclusão)
- models/: Estrutura da entidade Pet
- services/: Roteadores e middlewares
- views/: Página principal com EJS (index.ejs)
- db.json: Banco de dados simulado

77 Status do Projeto

≠ Em desenvolvimento contínuo. Melhorias e funcionalidades adicionais estão sendo avaliadas para versões futuras.

Contribuições

Contribuições são bem-vindas! Você pode abrir uma issue com sugestões ou enviar um pull request diretamente pelo repositório no GitHub.



Este projeto é uma aplicação completa de backend com CRUD e simulação de banco de dados, com o objetivo de praticar conceitos essenciais do desenvolvimento de APIs e organização de sistemas web. É uma base sólida para aprofundar conhecimentos em backend e serve também como excelente ponto de partida para projetos acadêmicos e TCCs.

Se quiser que eu adicione capa, introdução formal ou até referências para formatar como TCC completo, posso ajudar também!