Instituto Politécnico de Tomar

Projeto Final de Interfaces Web
----------Pawpath----------

Catarina Pereira de Matos 25976

Licenciatura de Engenharia Informática

repositório github: https://github.com/catmat111/pawpath

Tomar. Janeiro . 2025

Índice

INTRODUÇÃO

Este projeto tem como objetivo explorar o uso da biblioteca *React* no desenvolvimento de interfaces web. Para isso, decidi criar uma plataforma que ajuda a reunir animais de estimação perdidos com as suas famílias, e desta maneira contribui também para a redução do número de animais abandonados nas ruas.

A plataforma permite que os utilizadores publiquem informações sobre animais desaparecidos ou encontrados, incluindo fotos e descrições. Além disso, oferece um espaço para interação entre utilizadores, facilitando a troca de informações úteis.

Com esta iniciativa, pretendemos aliar o aprendizado técnico a uma solução prática e acessível, que possa ser útil na resolução de um problema comum no dia a dia.

API REST

Recorri à ferramenta *Sheety* para criar a nossa API REST. Esta solução permite utilizar o *Google Sheets* para criar folhas de cálculo, que representam as nossas entidades, e realizar operações CRUD *(Create, Read, Update, Delete)* através dos *endpoints* fornecidos pela ferramenta.

Entidades e as suas informações

Entidade do Utilizador

A Entidade do Utilizador foi criada com o objetivo de permitir que os utilizadores criem as suas próprias contas, realizem login e naveguem pelo site de forma personalizada, com a sua própria conta. No *Sheety*, esta é representada pela nossa Folha 1.
***Endpoint*:** https://api.sheety.co/13ac488bcfe201a0f16f2046b162a2e3/api/folha1
Atributos

- Id - Este atributo é determinado de forma crescente, ou seja, o ID é atribuído conforme a ordem de chegada do utilizador ao site. O **ID** é utilizado para identificar de forma única cada utilizador, permitindo que o mesmo possa navegar e interagir no site com a sua conta pessoal.
- Nome – Durante a criação da conta, o utilizador é solicitado a fornecer o nome ou apelido que deseja utilizar no site. Este nome deve ser único para cada utilizador, para facilitar o reconhecimento das pessoas na plataforma.
- Password – O utilizador deve definir uma palavra-passe para garantir que a sua conta esteja protegida. Este atributo contribui para a segurança e privacidade da conta do utilizador.
- *Image* – O utilizador tem a opção de adicionar uma imagem ao seu perfil. Caso prefira não se expor, a imagem é inicializada com uma imagem padrão. Se for adicionado imagem, a mesma é convertida para base 64, e se for a padrão ela está guardada na pasta *assets* do projeto.

  Exemplo:

id	nome	password	image
3	tomas	123	/src/assets/anonimo.png

Entidade de *Post*

A criação de um *post* pode ser feita na área "Fazer *Post*", onde é necessário fornecer informações como o nome do animal, uma descrição, a cor e, opcionalmente, uma imagem do animal desaparecido. No *Sheety*, esta entidade é representada pela Folha 2.

***Endpoint*:** https://api.sheety.co/13ac488bcfe201a0f16f2046b162a2e3/api/folha2

Atributos

- Iduti - Este atributo é o ID do utilizador, que foi criado anteriormente na Entidade de Utilizador. Ele serve para vincular o *post* ao utilizador que o criou.
- Nome – Nome do animal desaparecido. Este atributo é utilizado para identificar o animal.
- Texto – Este atributo é gerado automaticamente e consiste em 3 caracteres aleatórios, que podem ser alfanuméricos. O idpost é utilizado para identificar unicamente cada *post* na plataforma.
- idpost – Id Criado a partir de 3 caracteres aleatórios do alfabéticos ou numéricos, que serve para identificar o *post*.
- Encontrado – número de comentários de cada *post*.
- imagem – O utilizador pode adicionar uma imagem do animal desaparecido. A imagem não é obrigatória, mas, caso seja incluída, ela é convertida para o formato Base64 e armazenada na base de dados para ser exibida junto ao *post*.

Exemplo:

iduti	nome	cor	texto	idpost	encontrado	imagem
2	Lua	#000000	Gata preta, desapareceu a semana passada	IHD	3	Imagem em base 64

Entidade de Comentário

A Entidade de Comentário permite que os utilizadores interajam com os *posts*, fazendo comentários nos mesmos. Esta entidade está associada à Folha 3 no *Sheety*.

***Endpoint*:** https://api.sheety.co/13ac488bcfe201a0f16f2046b162a2e3/api/folha3

Atributos

- Iduti - Este é o ID do utilizador que está a fazer o comentário. O atributo refere-se ao mesmo ID criado na Entidade de Utilizador, e vincula o comentário ao utilizador responsável por ele.
- idpost – Identificador do *post* onde o comentário está a ser feito. Este ID é utilizado para relacionar o comentário ao *post* específico na plataforma.
- comentario – O texto do comentário em si, feito pelo utilizador na página do *post.*

Exemplo:

iduti	idpost	comentario
7	IHD	Encontrei!

Componente App

```
return (
  <Router>

        <Routes>
            <Route path="/" element={<Inicial />} /> {/* Página inicial */}
            <Route path="/Login" element={<Login />} /> {/* Página de Login */}
            <Route path="/sign_in" element={<Sign_in />} /> {/* Página de registro */}
            <Route path="/FeedProcurado" element={<FeedProcurado />} /> {/* Página do feed procurado */}
            <Route path="/User" element={<User />} /> {/* Página do user */}
            <Route path="/Password" element={<Password />} /> {/* Página para mudar a palavra pass */}
            <Route path="/Post" element={<Post />} /> {/* Página para criar Posts */}

        </Routes>
    </Router>
)
```

Neste componente, realizamos as redirecções necessárias para a navegação do site, utilizando os componentes “BrowserRouter”, “Routes” e “Route” da biblioteca “react-router-dom”.

Ecrãs no site

Ecrã Inicial

Esta é a página que se encontra quando alguém abre o site, onde é possível fazer login, caso o utilizador já possua conta ou criar uma conta, clicando nos respetivos botões.

código da Página:

```
return (
    <div>
        <div classname="logo">
            <img src={logo} />
        </div>
        <div className="bloco">
            <form>
                <Link to="/Login" className='route'>
                <button type="submit" className="botao" >Login</button>
                </Link>
                <Link to="/sign_in" className='route'>
                <button type="submit" className="botao" >Criar conta</button>
                </Link>
            </form>
        </div>
    </div>
);
```

Utilizei o componente “Link” para navegar para as páginas correspondentes com o botão

Página de login

Nesta página podemos fazer o log in de utilizador que já tenha criado conta, se colocar o nome ou a senha errada, dá erro e não é possível aceder ao site, se não é guardado o id e é redirecionado para o *feed* dos animais.

Código da Página:

```
<div>
    <div className="logo">
        <img src={logo} alt="Logo" />
    </div>
    <div className="bloco_login">
        <form onSubmit={validarLogin} id="login">
            <input
                type="text"
                className="inputs"
                id="nickName"
                placeholder="Nome do utilisador"
                value={nome}
                onChange={(e) => setNome(e.target.value)}
            />
            <input
                type="password"
                className="inputs"
                id="password"
                placeholder="Senha"
                value={senha}
                onChange={(e) => setSenha(e.target.value)}
            />
            <button type="submit" className="inputs" id="botao">Entrar</button>
        </form>
        {erro && <div className="erro">{erro}</div>}
    </div>
</div>
;
```

Foi implementado um método para validar as credenciais introduzidas, verificando se o nome de utilizador e a palavra-passe correspondem a um registo existente na base de dados. Caso as credenciais sejam válidas, o utilizador é redirecionado para a página do *feed*. Caso contrário, é exibida uma mensagem de erro.

```
// Função para validar o login
const validarLogin = (e) => {
    e.preventDefault();

    // Verifica se existe o utilisador na base de dados, com o nome e password inseridos
    const utilisador = utilisadores.find(user => user.nome === nome && user.password === senha);

    if (utilisador) {
        alert('Login bem-sucedido');
        navigate('/FeedProcurado', { state: { id: utilisador.id } }); // Envia o id via estado de
    } else {
        setErro('Nome de utilisador ou senha inválidos.');
    }
};
```

Página de criar conta

Nesta página é possível criar a conta do utilizador, quando é a sua primeira visita no site. A imagem *default* é a que está selecionada, mas é possível alterar clicando nela, o utilizador também tem de colocar um nome válido que seja diferente dos outros, e uma senha para proteger a conta.

Código da Página:

```
return (
  <div>
    <div>
      <img src={logo} alt="Logo" className="logo" />
    </div>
    <h2>Crie sua Conta</h2>

    <form className="form_signin" onSubmit={Submeter}>
      <div className="div-esquerda">
        <label className="imagem-input">
          <img id="anonimo" src={image} alt="Imagem do utilisador" />
          Escolha uma imagem
          <input
            className="inputs"
            type="file"
            accept="image/*"
            onChange={Mudar_imagem}
          />
        </label>
      </div>
      <div className="div-direita">
        <input
          className="inputs"
          type="text"
          placeholder="Nome"
          value={name}
          onChange={(e) => setName(e.target.value)}
          required
        />
        <input
          className="inputs"
          type="password"
          placeholder="Password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
          required
        />
        <button type="submit" className="inputs" id="botao">Criar conta</button>
      </div>
    </form>
  </div>
);
```

```
// Função para converter imagem em Base64
const convertToBase64 = (file) => {
  return new Promise((resolve, reject) => {
    const reader = new FileReader();
    reader.readAsDataURL(file);
    reader.onload = () => resolve(reader.result);
    reader.onerror = (error) => reject(error);
  });
};
```

Método que permite converter imagens para base 64, com o objeto *FileReader*.

Página *Feed*

Esta página exibe, por padrão, todos os *posts* realizados pelos utilizadores e inclui uma barra de pesquisa que permite a pesquisa tanto pelo nome do utilizador quanto pelo nome do animal desaparecido. A página está estruturada em dois componentes: a barra de navegação (Nav-bar) e o *feed* de *posts*.

Nav-bar

A barra de navegação contém o logotipo do projeto, um link (âncora) que redireciona para a página onde os utilizadores podem criar um *post*, e a imagem do utilizador. Ao clicar na imagem, o utilizador é redirecionado para o seu perfil.

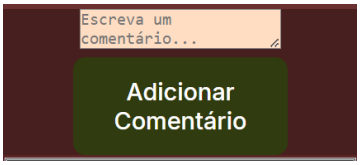
Feed

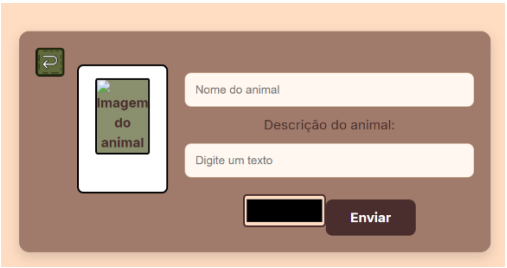
O *feed* é composto por uma área de pesquisa, onde os *posts* são exibidos de acordo com os critérios de pesquisa definidos pelo utilizador.

Post

Cada *post* é apresentado dentro de uma div, sendo a parte superior dedicada às informações do utilizador que criou o *post*, incluindo a sua imagem e nome. Na parte central, são apresentadas as informações relacionadas ao *post*, como o nome do animal, a descrição e a cor, além de uma imagem do animal, se fornecida. Também é mostrada a quantidade de comentários recebidos pelo *post*. Na parte inferior do *post*, encontra-se a zona de comentários, onde são exibidos o nome do comentador e o respetivo comentário.

Os utilizadores podem adicionar comentários clicando no *post*, o que abrirá uma caixa de texto onde poderão escrever o seu comentário.





Fazer *Post*

Nesta secção, os utilizadores têm a possibilidade de criar um *post* ao introduzir informações relevantes sobre o animal. É possível carregar uma imagem do animal, indicar o seu nome, adicionar uma descrição detalhada e especificar a cor. Após o preenchimento dessas informações, o *post* será criado e exibido no *feed* da plataforma.

Código da página:

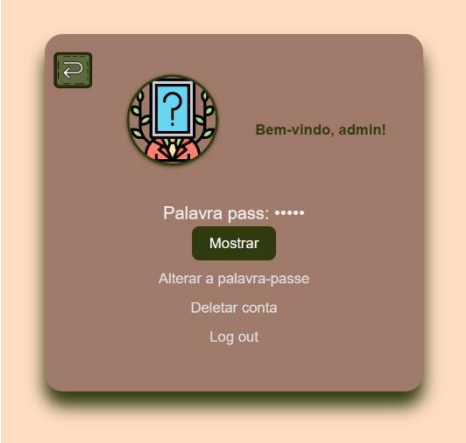
```jsx
return (
    <div className="user-container_post">
        <img
            src={voltar_img}
            onClick={voltar_botao}
            alt="Voltar"
            className="voltar_post"
        />
        <div className="div-esquerda_post">
            <label className="imagem-input_post">
                <img id="img" src={image} alt="Imagem do animal" />
                <input
                    className="inputs_post_post"
                    type="file"
                    accept="image/*"
                    onChange={Mudar_imagem}
                />
            </label>
        </div>
        <div className="div-direita_post">
            <input
                type="text"
                value={nome}
                onChange={(e) => setNome(e.target.value)}
                className="text-input_post"
                placeholder='Nome do animal'
            />
            <label>Descrição do animal:</label>
            <input
                type="text"
                value={text}
                onChange={(e) => setText(e.target.value)}
                placeholder="Digite um texto"
                className="text-input_post"
            />

            <input
                type="color"
                value={cor}
                onChange={(e) => setCor(e.target.value)}
            />

            <button
                className="submit_post"
                onClick={Submeter}
            >
                Enviar
            </button>
        </div>
    </div>
);
```

```jsx
// Função para voltar
const voltar_botao = () => {
    navigate('/FeedProcurado', { state: { id } });
};
```

Método que faz voltar para a página do *Feed*.

```jsx
const Submeter = async () => {
    if (!text || !nome || !cor) {
        alert('Por favor, preencha pelo menos o nome, a descrição e a cor');
        return;
    }

    // Limitar o tamanho do texto
    const maxTextLength = 50000;
    if (text.length > maxTextLength) {
        alert(`O texto ultrapassou o limite de ${maxTextLength} caracteres. Por favor, reduza o tamanho.`);
        return;
    }

    // Verificar se o total de caracteres não excede 50.000
    const totalPayloadLength = text.length + (image ? image.length : 0);
    if (totalPayloadLength > 50000) {
        alert('O total de caracteres ultrapassou o limite de 50.000. Por favor, reduza o texto ou a imagem.');
        return;
    }


    const id_post = generateUniqueId(); // Gerar ID único


    const url = 'https://api.sheety.co/13ac488bcfe281a0f16f2046b162a2e3/api/folha2';
    const payload = {
        folha2: {
            iduti: id,
            idpost: id_post,
            nome: nome,
            cor: cor,
            texto: text,
            imagem: image,
            encontrado: 0,
            comentarios:'',
        },
    };

    try {
        const response = await fetch(url, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify(payload),
        });

        if (!response.ok) {
            const errorMessage = await response.text();
            throw new Error(`Erro ao salvar os dados: ${response.status} - ${errorMessage}`);
        }

        alert('Dados enviados com sucesso!');
        setImage('');
        setText('');
        setNome('');
        setCor('#000000');
    } catch (error) {
        console.error('Erro ao enviar dados:', error);
        alert('Erro ao enviar os dados. Tente novamente. Detalhes: ' + error.message);
    }
};
```

Com o método submeter é possível criar o *Post*, e vai tudo para a entidade na Base de Dados.



Perfil utilizador

Aqui é possível ver as informações do utilizador e fazer algumas alterações com a conta. É possível mudar a imagem, clicando na imagem atual, alterar a palavra-pass, deletar a conta e fazer *logout*, as duas últimas fazem a página ser redirecionada para a Página principal.

Código da Página:

```
return (
    <>
    <img src= {logo}className='logo_user'></img>
    <div className="user-container">

        <img
            src={voltar}
            onClick={voltar_clicar}
            alt="Voltar"
            className="voltar"
        />
        <div className="top">
            <div className="image-container">
                <label htmlFor="file-input">
                    <img
                        src={userImage}
                        alt={userNome || "Utilisador"}
                        className="image"
                    />
                </label>
                <input
                    id="file-input"
                    className="inputs"
                    type="file"
                    accept="image/*"
                    onChange={Mudar_imagem}
                />
            </div>
            <p className="nome">Bem-vindo, {userNome}!</p>
        </div>
        <div className="password-container">
            <p className="password">
                Palavra pass: {hidden ? '•'.repeat(userPass?.length) : userPass}
            </p>
            <button onClick={Mudar_visibilidade} className="mostrar">
                {hidden ? 'Mostrar' : 'Esconder'}
            </button>
        </div>
        <div className="actions">
            <p onClick={Mudar_password}>Alterar a palavra-passe</p>
            <p onClick={Deletar}>Deletar conta</p>
            <p onClick={logout}>Log out</p>
        </div>
    </div></>
);
```

```
//altera a visibilidade da palavra pass
const Mudar_visibilidade = () => setHidden(!hidden);
```

Muda o estado de visibilidade da palavra *pass*.

```
const Mudar_password = () => {
    navigate('/Password', { state: { id } });
};
```

Leva para a página de mudar a password.

```
// Deletar conta
const Deletar = async () => {
    const confirmDelete = window.confirm('Você tem certeza que deseja excluir sua conta? Esta ação não pode ser desfeita.');

    if (confirmDelete) {
        try {
            const response = await fetch(`https://api.sheety.co/13ac488bcfe201a0f16f2046b162a2e3/api/folha1/${id}`, {
                method: 'DELETE',
            });

            if (!response.ok) {
                throw new Error(`Erro ao deletar a conta: ${response.status}`);
            }

            alert('Conta deletada com sucesso!');
            navigate('/'); // Redireciona para a página inicial
        } catch (error) {
            console.error('Erro ao deletar a conta:', error);
            alert('Erro ao deletar a conta. Tente novamente.');
        }
    }
};
```

Apaga a conta do utilizador que está a navegar.

Mudar a Palavra-pass

O utilizador consegue alterar a sua palavra-pass para uma mais conveniente.

Limitações do trabalho

Este site tem algumas limitações como: tem um limite de 49Kb para adicionar imagens, isto é assim pois a base de dados não permite que o texto enviado que está em base 64 seja tão grande, por esse motivo não é possível adicionar uma imagem com boa qualidade sem ser *icons* pequenos, algumas imagens também se encontram na pasta "assets", por isso alguém sem essa pasta no lugar certo não conseguiria visualizar o site de forma correta. Também não é possível alterar nem deletar os *posts* já criados.

Conclusões

Durante o desenvolvimento deste projeto, consolidámos os conhecimentos adquiridos sobre o uso de *React*, explorando a criação de interfaces dinâmicas e a navegação entre componentes. Adquirimos também experiência prática na realização de operações CRUD em uma API, reforçando a compreensão sobre a integração de aplicações *frontend* com bases de dados. Além disso, o projeto foi desenvolvido com o propósito de oferecer uma solução útil à sociedade, focando na localização de animais perdidos. Este objetivo atribuiu significado ao trabalho, destacando a importância de criar soluções tecnológicas que respondam a necessidades reais e promovam impacto positivo na comunidade.