Proyecto de software

Profesor

Jose Castro

Alumno

Johan Mauricio Ávila Sánchez

Id Banner: 100106768

Corporación Universitaria Iberoamericana

Proyecto de software

Diciembre - 2023

Tabla de contenido

Introducción 3
Home de la aplicación web 4
Login de la aplicación web 7
MenuBar de la aplicación web 12
Sección de post nuevo de la aplicación web 13
Post de la aplicación web 15

Introducción

Manual técnico

Home de la aplicación web

```
import { Card, Select, Button } from 'antd';
import { SearchOutlined } from '@ant-design/icons';
import MenuBar from './MenuBar';
import ciudades from '../data/ciudades.json'
import { useEffect, useState } from 'react';
import { obtenerTodosLosPost } from '../servicios/requests';

const { Meta } = Card;
const Home = () => {
    const [history, setHistory] = useState([])
    const [posts, setPosts] = useState([])
    const [ciudad, setCiudad] = useState()
    const [tipo, setTipo] = useState()
    const [razon, setRazon] = useState()
    useEffect( () => {
        async function fetchData() {
            const result = await obtenerTodosLosPost();
            console.log(result)
            setPosts(result)
            setHistory(result)
          }
          fetchData();
    }, [])

    const filterOption = (input, option) =>
    (option?.label ?? '').toLowerCase().includes(input.toLowerCase());
    const clearSelected = () => {
        setCiudad(null)
        setTipo(null)
        setRazon(null)
        setPosts([...history])
      }
    const aplicarFiltros = () => {
        let postsAux = [...posts]
        if (ciudad) {
            postsAux = postsAux.filter(item => item.mascota.ciudad ===
ciudad)
        }
        if (tipo) {
            postsAux = postsAux.filter(item => item.mascota.tipo_mascota ===
tipo)
        }
        if (razon) {
            postsAux = postsAux.filter(item => item.razon === razon)
        }
        setPosts(postsAux)
    }
    return (
        <>
        <MenuBar/>
            <div className='filters'>
            <Select
                showSearch
                placeholder="Selecciona una ciudad"
                optionFilterProp="children"
                filterOption={filterOption}
                options={ciudades}
                onChange={value => setCiudad(value)}
                value={ciudad}
            />
            <Select
            placeholder="Tipo de mascota"
                options={[
                    { value: 'Perro', label: 'Perro' },
                    { value: 'Gato', label: 'Gato' },
                    { value: 'Conejo', label: 'Conejo' },
                ]}
                onChange={value => setTipo(value)}
                value={tipo}
            />
            <Select
                placeholder="Razón"
                options={[
                    { value: 'Rescatado', label: 'Rescatado' },
                    { value: 'Adopción', label: 'Adopción' },
                ]}
                onChange={value => setRazon(value)}
                value={razon}
            />
            <Button type="primary" shape="circle" icon={<SearchOutlined />}
onClick={() => aplicarFiltros()}/>
            <Button type="primary" danger onClick={() => clearSelected()}>
                                    Limpiar filtros
                                </Button>

            </div>
            <div className='list'>
                <>
                {posts.map(post => (
                    <div className='list__card' key={post.id}>
                <Card
                    hoverable
                    style={{ width: 240 }}
                    cover={<img alt="example"
src={`http://localhost:8000/uploads/${post.mascota.url_foto}`} />}
                >
                    <Meta title={post.mascota.nombre} description={new
Date(post.fecha).toDateString()}/>
                    <p>{`Raza: ${post.mascota.raza}`}</p>
                    <p>{`Estado: ${post.razon}`}</p>
                    <p>{`Ciudad: ${post.mascota.ciudad}`}</p>
                    <p>{`Responsable: ${post.usuario.nombre}`} </p>
                    <p>{`Celuar: ${post.usuario.celular}`}</p>
                    <p>{`Descripción: ${post.descripcion}`} </p>
                </Card>
                </div>
                ))}
                </>

            </div>
        </>
    );
}
export default Home;
```

Login de la aplicación web

```
import { Button, Form, Input, Avatar } from 'antd';
import { useState } from 'react';
import { UserOutlined } from '@ant-design/icons';
const Login = () => {
    const formItemLayout = {
        labelCol: {
            xs: {
                span: 24,
            },
            sm: {
                span: 8,
            },
        },
        wrapperCol: {
            xs: {
                span: 24,
            },
            sm: {
                span: 16,
            },
        },
    };
    const tailFormItemLayout = {
        wrapperCol: {
            xs: {
                span: 24,
                offset: 0,
            },
            sm: {
                span: 16,
                offset: 8,
            },
        },
    };
    const secondForm = (<>
        <Form.Item
            name="email"
            label="E-mail"
            rules={[
                {
                    type: 'email',
                    message: 'The input is not valid E-mail!',
                },
                {
                    required: true,
                    message: 'Please input your E-mail!',
                },
            ]}
        >
            <Input />
        </Form.Item>
        <Form.Item
            name="password"
            label="Password"
            rules={[
                {
                    required: true,
                    message: 'Please input your password!',
                },
            ]}
            hasFeedback
        >
            <Input.Password />
        </Form.Item>
        <Form.Item
            name="confirm"
            label="Confirm Password"
            dependencies={['password']}
            hasFeedback
            rules={[
                {
                    required: true,
                    message: 'Please confirm your password!',
                },
                ({ getFieldValue }) => ({
                    validator(_, value) {
                        if (!value || getFieldValue('password') === value) {
                            return Promise.resolve();
                        }
                        return Promise.reject(new Error('The new password
that you entered do not match!'));
                    },
                }),
            ]}
        >
            <Input.Password />
        </Form.Item>
        <Form.Item
```

```
            name="nombre"
            label="Nombre"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa tu nombre!',
                    whitespace: true,
                },
            ]}
        >
            <Input />
        </Form.Item>
        <Form.Item
            name="direccion"
            label="Dirección"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa tu nombre!',
                    whitespace: true,
                },
            ]}
        >
            <Input />
        </Form.Item>
        <Form.Item
            name="celular"
            label="Celular"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa tu celular!',
                },
            ]}
        >
            <Input
                style={{
                    width: '100%',
                }}
            />
        </Form.Item>
    </>)
    const defaultForm = (
        <>
            <Form.Item
```

```
                name="email"
                label="E-mail"
                rules={[
                    {
                        type: 'email',
                        message: 'The input is not valid E-mail!',
                    },
                    {
                        required: true,
                        message: 'Please input your E-mail!',
                    },
                ]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                name="password"
                label="Password"
                rules={[
                    {
                        required: true,
                        message: 'Please input your password!',
                    },
                ]}
                hasFeedback
            >
                <Input.Password />
            </Form.Item>

        </>
    )
    const [form] = Form.useForm();
    const [optionsForm, setoptionsForm] = useState(defaultForm)
    const [toggleForm, setToggleForm] = useState(true)
    const onFinish = (values) => {
        console.log('Success:', values);
    };
    const onFinishFailed = (errorInfo) => {
        console.log('Failed:', errorInfo);
    };
    const register = () => {
        setToggleForm(!toggleForm);
    }
    return (
        <div className='content_form'>
```

```
            <Avatar
                icon={<UserOutlined />}
                size={{
                    xs: 24,
                    sm: 32,
                    md: 40,
                    lg: 64,
                    xl: 80,
                    xxl: 100,
                }} />

            <Form
                {...formItemLayout}
                form={form}
                name="register"
                style={{
                    maxWidth: 800,
                    minWidth: 500
                }}
                onFinish={onFinish}
                onFinishFailed={onFinishFailed}
                autoComplete="off"
                scrollToFirstError
            >
                {toggleForm ? optionsForm : secondForm}

                <Form.Item
                    {...tailFormItemLayout}
                >
                    <Button type="primary" htmlType="submit">
                        {toggleForm ? 'Ingresar' : 'Registrar'}
                    </Button>
                    <a onClick={() => register()}>{toggleForm ? 'Registrate
ahora!' : 'Ingresar!'}</a>
                </Form.Item>
            </Form>
        </div>
    );

}
export default Login;
```

MenuBar de la aplicación web

```
import { useState } from 'react';
import { HomeOutlined, PlusOutlined, FileTextOutlined } from '@ant-
design/icons';
import { useNavigate, useLocation   } from "react-router-dom";
import { Menu } from 'antd';
const MenuBar = () => {
    const navigate = useNavigate();
    let location = useLocation();
    const items = [
        {
            label: 'Inicio',
            key: 'home',
            icon: <HomeOutlined />,
        },
        {
            label: 'Publicar',
            key: 'post',
            icon: <PlusOutlined />,
        },
        {
            label: 'Mis publicaciones',
            key: 'myposts',
            icon: <FileTextOutlined  />,
        }
    ];
    const [current, setCurrent] = useState(location.pathname.split('/')[1]);
    const onClick = (e) => {

        setCurrent(e.key);
        navigate(`/${e.key}`)
    };
    return <Menu onClick={onClick} selectedKeys={[current]}
mode="horizontal" items={items} />
}
export default MenuBar;
```

Sección de post nuevo de la aplicación web

```
import { Card, Button, message} from 'antd';
import MenuBar from './MenuBar';
import { useEffect, useState } from 'react';
import { eliminarPost, obtenerTodosLosPost } from '../servicios/requests';

const { Meta } = Card;
const MyPosts = () => {
    const [messageApi, contextHolder] = message.useMessage();
    const [posts, setPosts] = useState([])
    useEffect(() => {
        async function fetchData() {
            await obtenerPost();
        }
        fetchData();
    }, [])
    const obtenerPost = async () => {
        const result = await obtenerTodosLosPost();
        console.log(result)
        setPosts(result)
    }
    const eliminarPublicacion = async (id) => {

        await eliminarPost(id)
        await obtenerPost()
        messageApi.open({
            type: 'success',
            content: 'Se ha eliminado exitosamente',
        });
    }

    return (
        <>
            <MenuBar />
            <div className='list'>
                <>
                    {posts.map(post => (
                        <div className='list__card' key={post.id}>
                            <Card
                                hoverable
                                style={{ width: 240 }}
                                cover={<img alt="example"
src={`http://localhost:8000/uploads/${post.mascota.url_foto}`} />}
                            >
```

```
                                <Meta title={post.mascota.nombre}
description={new Date(post.fecha).toDateString()} />
                                <Button type="primary" danger onClick={() =>
eliminarPublicacion(post.id)}>
                                    Eliminar publicación
                                </Button>
                            </Card>
                        </div>
                    ))}
                </>

            </div>
            {contextHolder}
        </>
    );
}
export default MyPosts;
```

Post de la aplicación web

```
import { Button, Form, Input, Upload, message, Select } from 'antd';
import MenuBar from './MenuBar';
import { UploadOutlined } from '@ant-design/icons';
import { guardarPost } from '../servicios/requests';
import { useNavigate } from 'react-router-dom';
import ciudades from '../data/ciudades.json'
const Post = () => {
    const navigate = useNavigate();
    const [messageApi, contextHolder] = message.useMessage();


    const onFinish = async (values) => {
        console.log('Success:', values);
        const form = new FormData();
        form.append('file', values.image.file)
        form.append('razon', values.razon)
        form.append('descripcion', values.descripcion)
        form.append('usuario_id', 1)
        form.append('raza', values.raza)
        form.append('ciudad', values.ciudad)
        form.append('tipo_mascota', values.tipo_mascota)
        form.append('nombre', values.nombre)
        form.append('fecha', new Date().toUTCString())
        await guardarPost(form)
        messageApi.open({
            type: 'success',
            content: 'Se ha publicado exitosamente',
        });
        setTimeout(() => {
            navigate(`/home`)
        }, 2000)
    };
    const onFinishFailed = (errorInfo) => {
        console.log('Failed:', errorInfo);
    };
    const tailFormItemLayout = {
        wrapperCol: {
            xs: {
                span: 24,
                offset: 0,
            },
            sm: {
                span: 16,
                offset: 8,
            },
        },
    };
    const formItemLayout = {
        labelCol: {
            xs: {
                span: 24,
            },
            sm: {
                span: 8,
            },
        },
        wrapperCol: {
            xs: {
                span: 24,
            },
            sm: {
                span: 16,
            },
        },
    };
    const filterOption = (input, option) =>
        (option?.label ?? '').toLowerCase().includes(input.toLowerCase());
    const [form] = Form.useForm();
    const secondForm = (<>
        <Form.Item
            name="nombre"
            label="Nombre"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa el nombre de la mascota!',
                    whitespace: true,
                },
            ]}
        >
            <Input />
        </Form.Item>
        <Form.Item
            name="raza"
            label="Raza"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa la raza de la mascota!',
                    whitespace: true,
                },
            ]}
        >
            <Input />
        </Form.Item>
        <Form.Item
            name="tipo_mascota"
            label="Tipo de mascota"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa la raza de la mascota!',
                    whitespace: true,
                },
            ]}
        >
            <Select
                options={[
                    { value: 'Perro', label: 'Perro' },
                    { value: 'Gato', label: 'Gato' },
                    { value: 'Conejo', label: 'Conejo' },
                ]}
            />
        </Form.Item>
        <Form.Item
            name="image"
            label="Foto de la mascota"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa una imagen de la mascota!',
                },
            ]}
        >
            <Upload name="logo" beforeUpload={() => false}
listType="picture">
                <Button icon={<UploadOutlined />}>Click to Upload</Button>
            </Upload>
        </Form.Item>
        <Form.Item
            name="ciudad"
            label="Ciudad"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa la ciudad!',
                    whitespace: true,
                },
            ]}
        >
            <Select
                showSearch
                placeholder="Selecciona una ciudad"
                optionFilterProp="children"
                filterOption={filterOption}
                options={ciudades}
            />
        </Form.Item>
        <Form.Item
            name="razon"
            label="Razón"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa si está en adopción o fue
rescatado!',
                    whitespace: true,
                },
            ]}
        >
            <Select
                options={[
                    { value: 'Rescatado', label: 'Rescatado' },
                    { value: 'Adopción', label: 'Adopción' },
                ]}
            />
        </Form.Item>
        <Form.Item
            name="descripcion"
            label="Descripción"
            rules={[
                {
                    required: true,
                    message: 'Por favor ingresa una breve descripción de la
mascota!',
                    whitespace: true,
                },
            ]}
        >
            <Input />
        </Form.Item>
    </>)

    return (
        <>
            <MenuBar />
            <Form
                {...formItemLayout}
                form={form}
                name="register"
                style={{
                    maxWidth: 800,
                    minWidth: 500
                }}
                onFinish={onFinish}
                onFinishFailed={onFinishFailed}
                autoComplete="off"
                scrollToFirstError
            >
                {secondForm}

                <Form.Item
                    {...tailFormItemLayout}
                >
                    <Button type="primary" htmlType="submit">
                        Publicar
                    </Button>
                </Form.Item>
            </Form>
            {contextHolder}
        </>
    )
}
export default Post;
```