React projekt – Régió App (komponensekre bontva + API szolgáltatás)

Az alábbiakban egy Vite + React alapú projektvázat kapsz. A korábbi egyfájlos megoldás szétszedve külön **oldalakra** (pages), **közös komponensekre** (components) és egy külön **API szolgáltatás** modulra (services/api.js). A menü külön komponens lett.

Könyvtárstruktúra

```
region-app/
├─ .env.example
├─ README.md
├─ index.html
├─ package.json
├─ vite.config.js
└─ src/
   ├─ main.jsx
   ├─ App.jsx
   ├─ components/
   │  └─ Menu.jsx
   ├─ pages/
   │  ├─ Regiok.jsx
   │  └─ UjRegio.jsx
   ├─ services/
   │  └─ api.js
   └─ styles.css
```

Telepítés (lépések)

1. **Projekt létrehozása (Vite):**

   ```
   npm create vite@latest region-app -- --template react
   cd region-app
   npm install
   ```

2. **Kötelező csomagok:**

   ```
   npm i axios react-router-dom bootstrap
   ```

3. **Bootstrap CSS/JS betöltés:** vagy az `index.html`-ben CDN-nel, vagy a `main.jsx`-ben importtal (lásd lent).

4. **Környezeti változó (opcionális, ajánlott):**

- Másold le az `.env.example` fájlt `.env`-nek, és állítsd be az API elérési útját.

5. **Fejlesztői szerver indítása:**

   ```
   npm run dev
   ```

.env.example

```
VITE_API_BASE_URL=http://localhost:3001
```

Ha nincs `.env`, a `services/api.js` alapértelmezésként `http://localhost:3001`-re áll.

---

src/services/api.js (API szolgáltatás)

```jsx
import axios from 'axios';

const baseURL = import.meta.env.VITE_API_BASE_URL || 'http://localhost:3001';

export const api = axios.create({
  baseURL,
  headers: { 'Content-Type': 'application/json' }
});

export async function getRegiok() {
  const { data } = await api.get('/regiok');
  return data; // [{ Rid, regionev, regio_tipusa }, ...]
}

export async function createRegio(payload) {
  // payload: { Rid, regionev, regio_tipusa }
  const { data } = await api.post('/ujregio', payload);
  return data;
}

export async function deleteRegio(id) {
  const { data } = await api.delete(`/torles/${id}`);
  return data;
}
```

---

src/components/Menu.jsx (Menü komponens)

```jsx
import { Link, NavLink } from 'react-router-dom';

export default function Menu() {
  return (
```

```
    <nav className="navbar navbar-expand-lg navbar-light bg-light mb-4">
      <div className="container-fluid">
        <Link className="navbar-brand" to="/">Régió App</Link>

        <button className="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#mainNav">
          <span className="navbar-toggler-icon"></span>
        </button>

        <div className="collapse navbar-collapse" id="mainNav">
          <ul className="navbar-nav me-auto mb-2 mb-lg-0">
            <li className="nav-item">
              <NavLink className={({isActive}) => `nav-link ${isActive ?
'active' : ''}`} to="/regiok">
                Régiók listája
              </NavLink>
            </li>
            <li className="nav-item">
              <NavLink className={({isActive}) => `nav-link ${isActive ?
'active' : ''}`} to="/ujregio">
                Új régió hozzáadása
              </NavLink>
            </li>
          </ul>
        </div>
      </div>
    </nav>
  );
}
```

---

src/pages/Regiok.jsx (Lista oldal)

```
import { useEffect, useState } from 'react';
import { getRegiok, deleteRegio } from '../services/api';

export default function Regiok() {
  const [regiok, setRegiok] = useState([]);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState('');

  useEffect(() => {
    (async () => {
      try {
        const data = await getRegiok();
        setRegiok(data);
      } catch (e) {
        setError('Nem sikerült betölteni a régiók listáját.');
      } finally {
```

```
        setLoading(false);
      }
    })();
  }, []);

  const handleDelete = async (id) => {
    if (!confirm(`Biztosan törlöd a(z) ${id} azonosítójú régiót?`)) return;
    try {
      await deleteRegio(id);
      setRegiok((prev) => prev.filter((r) => r.Rid !== id));
    } catch (e) {
      alert('Hiba történt a törléskor.');
    }
  };

  if (loading) return <p>Betöltés…</p>;
  if (error) return <div className="alert alert-danger">{error}</div>;

  return (
    <div className="container">
      <h2 className="mb-3">Régiók listája</h2>
      <div className="table-responsive">
        <table className="table table-striped table-bordered align-middle">
          <thead className="table-light">
            <tr>
              <th>Rid</th>
              <th>Régió neve</th>
              <th>Régió típusa</th>
              <th style={{width: 1}}>Műveletek</th>
            </tr>
          </thead>
          <tbody>
            {regiok.map((regio) => (
              <tr key={regio.Rid}>
                <td>{regio.Rid}</td>
                <td>{regio.regionev}</td>
                <td>{regio.regio_tipusa}</td>
                <td>
                  <button className="btn btn-sm btn-danger" onClick={() =>
handleDelete(regio.Rid)}>
                    Törlés
                  </button>
                </td>
              </tr>
            ))}
            {regiok.length === 0 && (
              <tr>
                <td colSpan={4} className="text-center text-muted">Nincs
megjeleníthető régió.</td>
              </tr>
```

```
            )}
          </tbody>
        </table>
      </div>
    </div>
  );
}
```

---

src/pages/UjRegio.jsx (Űrlap oldal)

```
import { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import { createRegio } from '../services/api';

export default function UjRegio() {
  const navigate = useNavigate();
  const [formData, setFormData] = useState({ Rid: '', regionev: '',
regio_tipusa: '' });
  const [saving, setSaving] = useState(false);
  const [error, setError] = useState('');

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormData((prev) => ({ ...prev, [name]: value }));
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    setError('');

    if (!formData.Rid || !formData.regionev || !formData.regio_tipusa) {
      setError('Minden mező kitöltése kötelező.');
      return;
    }

    try {
      setSaving(true);
      await createRegio(formData);
      alert('Új régió sikeresen hozzáadva!');
      navigate('/regiok');
    } catch (e) {
      setError('Hiba történt a mentés során.');
    } finally {
      setSaving(false);
    }
  };

  return (
```

```
    <div className="container">
      <h2 className="mb-3">Új régió hozzáadása</h2>
      {error && <div className="alert alert-danger">{error}</div>}

      <form onSubmit={handleSubmit} noValidate>
        <div className="mb-3">
          <label htmlFor="Rid" className="form-label">Régió ID (Rid)</label>
          <input id="Rid" name="Rid" className="form-control"
value={formData.Rid} onChange={handleChange} />
        </div>
        <div className="mb-3">
          <label htmlFor="regionev" className="form-label">Régió neve</label>
          <input id="regionev" name="regionev" className="form-control"
value={formData.regionev} onChange={handleChange} />
        </div>
        <div className="mb-3">
          <label htmlFor="regio_tipusa" className="form-label">Régió
típusa</label>
          <input id="regio_tipusa" name="regio_tipusa" className="form-
control" value={formData.regio_tipusa} onChange={handleChange} />
        </div>

        <button type="submit" className="btn btn-primary" disabled={saving}>
          {saving ? 'Mentés…' : 'Hozzáad'}
        </button>
      </form>
    </div>
  );
}
```

---

src/App.jsx

```
import { Routes, Route } from 'react-router-dom';
import Menu from './components/Menu';
import Regiok from './pages/Regiok';
import UjRegio from './pages/UjRegio';
import './styles.css';

export default function App() {
  return (
    <div className="container mt-3">
      <Menu />
      <Routes>
        <Route path="/regiok" element={<Regiok />} />
        <Route path="/ujregio" element={<UjRegio />} />
        <Route path="/" element={<h2>Üdvözöllek a Régió Appban!</h2>} />
      </Routes>
    </div>
```

```
  );
}
```

---

src/main.jsx

```jsx
import React from 'react';
import ReactDOM from 'react-dom/client';
import { BrowserRouter } from 'react-router-dom';
import App from './App.jsx';

// Bootstrap importok
import 'bootstrap/dist/css/bootstrap.min.css';
import 'bootstrap/dist/js/bootstrap.bundle.min.js';

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <BrowserRouter>
      <App />
    </BrowserRouter>
  </React.StrictMode>
);
```

---

src/styles.css (opcionális apróságok)

```css
body { background-color: #f8f9fa; }
.table td, .table th { vertical-align: middle; }
```

README.md (rövid leírás)

```md
# Régió App (React + Vite)

Egyszerű CRUD felület régiók kezeléséhez.

## Fő egységek
- `pages/Regiok.jsx` – lista + törlés
- `pages/UjRegio.jsx` – űrlap + mentés
- `components/Menu.jsx` – navigáció
- `services/api.js` – API hívások központosítva (axios)

## Beállítás
1. `npm i`
2. `npm i axios react-router-dom bootstrap`
3. `.env`: `VITE_API_BASE_URL=http://localhost:3001`
4. `npm run dev`
```

```
## Backend elvárások
- `GET    /regiok` – lista
- `POST   /ujregio` – létrehozás (Rid, regionev, regio_tipusa)
- `DELETE /torles/:id` – törlés

> Ha az endpointok más néven élnek, csak a `services/api.js`-t kell módosítanod.
```

Mi változott az eredetihez képest?

- **Menü külön komponensben** (átláthatóbb, új menüpont könnyen hozzáadható).
- **API hívások központosítva** a `services/api.js`-ben (egy helyen állítható a baseURL, endpointok, hibakezelés bővíthető).
- **Oldalakra bontás**: listázás és űrlap külön oldalon.
- **Alap hiba- és töltéskezelés** hozzáadva.
- **.env támogatás** a backend elérési út egyszerű cseréjéhez.

Tippek további fejlesztéshez

- Validáció: `Rid` legyen szám/egyedi; használhatsz `react-hook-form`-ot és yup-ot.
- Szerkesztés (PUT/PATCH) oldal hozzáadása.
- Központi állapotkezelés (pl. React Query a cache-hez, betöltési állapothoz).
- Törlés megerősítése modal ablakban.
- Nemzetköziesítés (i18n), pl. magyar/angol váltás.