📅 Breakdown Harian - Minggu 7: Konsumsi API dan State Management

🎯 Tujuan Minggu Ini

- Memahami cara mengambil data dari API menggunakan `fetch` dan `axios`.
- Menggunakan `useEffect` hook untuk side effects, termasuk fetching data.
- Menampilkan data produk yang diambil dari backend.
- Mengimplementasikan halaman login dan register.
- Membuat halaman detail produk.
- Memahami dasar-dasar state management untuk data global (opsional, pengantar).

📅 Hari 1 (Senin) - Mengambil Data dengan `fetch` dan `useEffect`

Materi Inti (2 Jam)

- **Pengantar Side Effects di React:**
  - Apa itu side effect? (Contoh: fetching data, manipulasi DOM langsung, setup subscriptions).
  - Mengapa kita butuh mengelola side effects di komponen fungsional?
- **`useEffect` Hook:**
  - Sintaks dasar `useEffect`.
  - Kapan `useEffect` dijalankan? (Setelah render pertama, setelah setiap update).
  - Dependency Array (`[]`, `[dep]`, tanpa array): Mengontrol kapan effect dijalankan kembali.
  - Cleanup Function: Membersihkan side effect (misalnya, membatalkan request, membersihkan timer, unsubscribe).
- **Mengambil Data dengan Native `fetch` API:**
  - Pengantar `fetch` API bawaan browser.
  - Sintaks `fetch` (`fetch(url)`).
  - Menggunakan `async/await` dengan `fetch`.
  - Menangani response (`.then()`, `.json()`).
  - Menangani error (`.catch()`).
- **Menggabungkan `useEffect` dan `fetch`:**
  - Contoh implementasi fetching data saat komponen pertama kali di-render.
  - Menangani loading state dan error state saat fetching.

Praktik Mandiri (8 Jam)

- Buat proyek React baru atau gunakan proyek dari minggu sebelumnya.
- Buat komponen sederhana (misalnya `DataFetcher.jsx`).
- Di dalam komponen tersebut, gunakan `useEffect` untuk melakukan fetching data dari dummy API (misalnya JSONPlaceholder).
- Tampilkan data yang berhasil diambil di UI komponen.
- Implementasikan loading state (tampilkan pesan "Loading...") saat data sedang diambil.
- Implementasikan error state (tampilkan pesan error) jika fetching gagal.
- Eksplorasi penggunaan dependency array pada `useEffect`.

- Coba implementasikan cleanup function sederhana (misalnya, menggunakan `AbortController` untuk membatalkan fetch request jika komponen unmount sebelum request selesai).

📆 Hari 2 (Selasa) - Mengambil Data dengan `axios` dan Menampilkan Data Produk

Materi Inti (2 Jam)

- **Pengantar `axios`:**
  - Library HTTP client berbasis Promise untuk browser dan Node.js.
  - Keunggulan `axios` dibandingkan `fetch` (interceptors, automatic JSON transformation, better error handling, cancellation).
- **Instalasi dan Penggunaan `axios`:**
  - Instalasi: `npm install axios` atau `yarn add axios`.
  - Sintaks dasar `axios` (`axios.get(url)`, `axios.post(url, data)`).
  - Menggunakan `async/await` dengan `axios`.
  - Menangani response dan error dengan `axios`.
- **Menampilkan Data Produk:**
  - Struktur data produk (contoh: array of objects dengan `id`, `name`, `price`, `description`, `imageUrl`).
  - Membuat komponen `ProductList.jsx`.
  - Menggunakan `useEffect` dan `axios` untuk mengambil daftar produk dari API backend (asumsikan ada endpoint `/api/products`).
  - Melakukan mapping data produk ke dalam daftar elemen UI (misalnya, menggunakan method `map` pada array).
  - Membuat komponen `ProductItem.jsx` untuk menampilkan detail satu produk dalam daftar.

Praktik Mandiri (8 Jam)

- Instal `axios` di proyek Anda.
- Buat komponen `ProductList.jsx`.
- Di dalam `ProductList`, gunakan `useEffect` dan `axios.get` untuk mengambil data dari dummy API yang mengembalikan array produk (Anda bisa membuat file JSON lokal atau menggunakan API publik dummy).
- Buat komponen `ProductItem.jsx` untuk menampilkan satu item produk (nama, harga, gambar).
- Render daftar produk di `ProductList` menggunakan `map` dan komponen `ProductItem`.
- Pastikan loading dan error state juga ditangani di `ProductList`.
- Tampilkan daftar produk di halaman utama aplikasi Anda.

📆 Hari 3 (Rabu) - Halaman Login dan Register

Materi Inti (2 Jam)

- **Konsep Autentikasi Sederhana:**
  - Proses login (mengirim kredensial ke backend, menerima token/sesi).
  - Proses register (mengirim data user baru ke backend).

    - Pentingnya keamanan (jangan simpan password di frontend, gunakan HTTPS).
- **Membuat Halaman Login:**
    - Membuat komponen `LoginPage.jsx`.
    - Menggunakan state (`useState`) untuk mengelola input username/email dan password pada form.
    - Menangani event `onSubmit` pada form.
    - Menggunakan `axios.post` untuk mengirim kredensial ke endpoint login backend (asumsikan `/api/auth/login`).
    - Menyimpan token autentikasi (misalnya di `localStorage` atau Context API/State Management Library).
    - Navigasi programatik (`useNavigate`) setelah login berhasil.
- **Membuat Halaman Register:**
    - Membuat komponen `RegisterPage.jsx`.
    - Menggunakan state untuk mengelola input form register (nama, email, password, dll.).
    - Menangani event `onSubmit` pada form register.
    - Menggunakan `axios.post` untuk mengirim data user baru ke endpoint register backend (asumsikan `/api/auth/register`).
    - Navigasi programatik setelah register berhasil (misalnya ke halaman login).
- **Menangani Error Form dan Validasi Sederhana:**
    - Menampilkan pesan error dari backend jika login/register gagal.
    - Implementasi validasi input sederhana di frontend (misalnya, cek field kosong).

Praktik Mandiri (8 Jam)

- Buat komponen `LoginPage.jsx` dan `RegisterPage.jsx`.
- Buat form di masing-masing komponen dengan input yang diperlukan.
- Gunakan `useState` untuk mengelola nilai input form.
- Implementasikan handler `onSubmit` untuk form.
- Gunakan `axios.post` untuk mengirim data form ke dummy API endpoint (Anda bisa simulasikan respons sukses/gagal di frontend untuk latihan).
- Setelah login/register berhasil (simulasi), gunakan `useNavigate` untuk pindah halaman.
- Tampilkan pesan error sederhana jika ada masalah (simulasi dari respons API).
- Tambahkan link antar halaman Login dan Register.
- Konfigurasi routing di aplikasi Anda untuk halaman Login dan Register.

📆 Hari 4 (Kamis) - Halaman Detail Produk dan Mengirim Header Autentikasi

Materi Inti (2 Jam)

- **Mengambil Data Detail Produk:**
    - Membuat komponen `ProductDetailPage.jsx`.
    - Menggunakan `useParams` hook dari React Router untuk mendapatkan ID produk dari URL (misalnya `/products/:id`).
    - Menggunakan `useEffect` dan `axios.get` untuk mengambil data detail produk dari API backend berdasarkan ID (asumsikan endpoint `/api/products/:id`).
    - Menampilkan detail produk yang berhasil diambil di UI.

    - Menangani loading dan error state untuk detail produk.
- **Mengirim Header Autentikasi:**
    - Mengapa perlu mengirim token autentikasi? (Mengakses endpoint yang dilindungi).
    - Cara menambahkan header `Authorization` dengan token (misalnya `Bearer token`).
    - Menggunakan konfigurasi `headers` pada request `axios`.
    - Contoh: Mengambil data profil user dari endpoint yang dilindungi (`/api/user/profile`) setelah login.
- **Mengelola Token Autentikasi:**
    - Menyimpan token di `localStorage` (kelebihan dan kekurangan).
    - Menggunakan Context API atau State Management Library untuk menyimpan status autentikasi dan token secara global (pengantar).

Praktik Mandiri (8 Jam)

- Buat komponen `ProductDetailPage.jsx`.
- Konfigurasi route di aplikasi Anda untuk `/products/:id` yang merender `ProductDetailPage`.
- Di `ProductList.jsx`, buat link pada setiap item produk yang mengarah ke `/products/:id` yang sesuai.
- Di `ProductDetailPage`, gunakan `useParams` untuk mendapatkan ID dari URL.
- Gunakan `useEffect` dan `axios.get` untuk mengambil data detail produk dari dummy API berdasarkan ID (simulasikan data detail produk).
- Tampilkan detail produk (nama, harga, deskripsi lengkap) di halaman ini.
- (Opsional) Jika Anda sudah mengimplementasikan login, coba buat endpoint dummy yang dilindungi dan coba akses menggunakan `axios` dengan menambahkan header `Authorization` yang berisi token dummy.

📆 Hari 5 (Jum'at) - Pengantar State Management dan Tugas Mingguan

Materi Inti (2 Jam)

- **Tantangan State Management di Aplikasi Skala Besar:**
    - "Prop drilling": Melewatkan props melalui banyak level komponen.
    - Mengelola state yang perlu diakses oleh banyak komponen yang tidak berelasi langsung.
- **Solusi State Management:**
    - **Context API:**
        - Pengantar Context API bawaan React.
        - Kapan menggunakan Context API? (Untuk data yang "global" di sebagian besar tree komponen, misalnya tema, user info, bahasa).
        - Kelebihan dan kekurangan Context API untuk state management kompleks.
    - **State Management Libraries (Pengantar):**
        - Mengapa library eksternal? (Fitur lebih canggih, performa, tooling).
        - Contoh library populer: Redux, Zustand, Recoil, Jotai.
        - Pengantar singkat tentang konsep dasar salah satu library (misalnya Zustand: simple, fast, scalable).
- **Review Materi Minggu Ini:**
    - Diskusi dan tanya jawab tentang `useEffect`, `fetch`, `axios`, fetching data, routing dengan parameter, dan autentikasi sederhana.

Tugas Mingguan (Waktu Pengerjaan: Sisa Minggu + Akhir Pekan)

Tema Tugas: Mengembangkan Aplikasi E-commerce Sederhana (Bagian Frontend)

Persyaratan Fungsional:

1. **Daftar Produk:** Menampilkan daftar produk yang diambil dari API backend (gunakan `axios`).
2. **Detail Produk:** Menampilkan detail lengkap satu produk saat diklik dari daftar produk (gunakan React Router dengan parameter URL dan `axios`).
3. **Halaman Login & Register:** Implementasikan halaman login dan register (gunakan `axios` untuk simulasi request ke backend).
4. **Navigasi:** Gunakan React Router untuk navigasi antar halaman (Daftar Produk, Detail Produk, Login, Register, dan halaman lain jika perlu).
5. **Styling:** Gunakan Tailwind CSS untuk styling seluruh aplikasi.
6. **Penanganan State:** Gunakan `useState` dan `useEffect` untuk mengelola state lokal komponen dan fetching data.
7. **Struktur Proyek:** Atur proyek dengan rapi (folder `components`, `pages`, `api`, dll.).
8. **Dokumentasi:** Buat `README.md` yang menjelaskan proyek, cara setup, struktur, dan teknologi yang digunakan.

Fitur Opsional (Nilai Tambah):

- Menambahkan fitur "Add to Cart" sederhana (menggunakan state lokal atau Context API).
- Menampilkan pesan loading/error yang lebih baik.
- Implementasi proteksi route (misalnya, halaman checkout hanya bisa diakses setelah login).
- Menggunakan Context API untuk mengelola state autentikasi global.

Output Tugas:

- Kode sumber proyek di repositori GitHub.
- File `README.md`.

💡 Tips Belajar Tambahan

- **Pahami Lifecycle Komponen dengan `useEffect`:** Latihan dengan berbagai kombinasi dependency array untuk memahami kapan effect dijalankan.
- **Gunakan Browser Developer Tools:** Tab Network sangat berguna untuk melihat request API yang dikirim dan respons yang diterima.
- **Simulasikan API:** Jika backend belum siap, gunakan JSON Server atau buat file JSON lokal dan import untuk simulasi data API.
- **Fokus pada Error Handling:** Selalu pertimbangkan apa yang terjadi jika request API gagal atau data yang diterima tidak sesuai harapan.
- **Mulai Pikirkan State Global:** Identifikasi data apa saja yang mungkin perlu diakses oleh banyak komponen dan pertimbangkan cara mengelolanya (Context API atau library).

🔗 Referensi Tambahan

- Dokumentasi React: Using the Effect Hook
- MDN Web Docs: Fetch API
- Dokumentasi Axios

- Dokumentasi React Router
- Dokumentasi Tailwind CSS
- Dokumentasi React: Context
- Dokumentasi Zustand

Minggu ini akan fokus pada bagaimana aplikasi frontend berinteraksi dengan backend melalui API, yang merupakan bagian krusial dari pengembangan fullstack. Memahami cara mengambil, mengirim, dan mengelola data dari API adalah kunci untuk membangun aplikasi web yang dinamis.