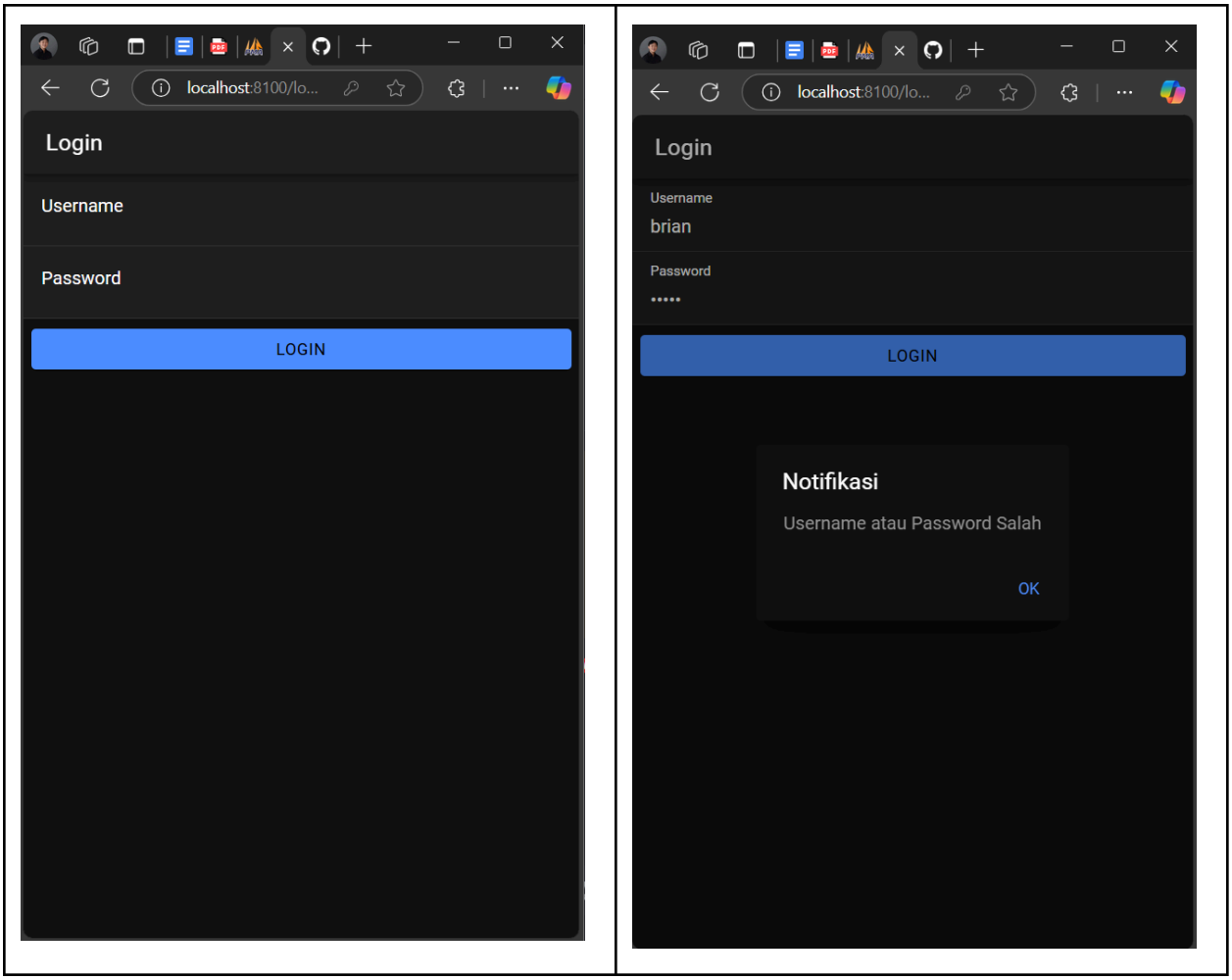
Nama : Brian Cahya Purnama
NIM : H1D022009
Shift Lama : C
Shift Baru : D

RESPONSI 2 PAKET C

Screenshot Aplikasi dan Penjelasan CRUD dan Login:

1. LOGIN



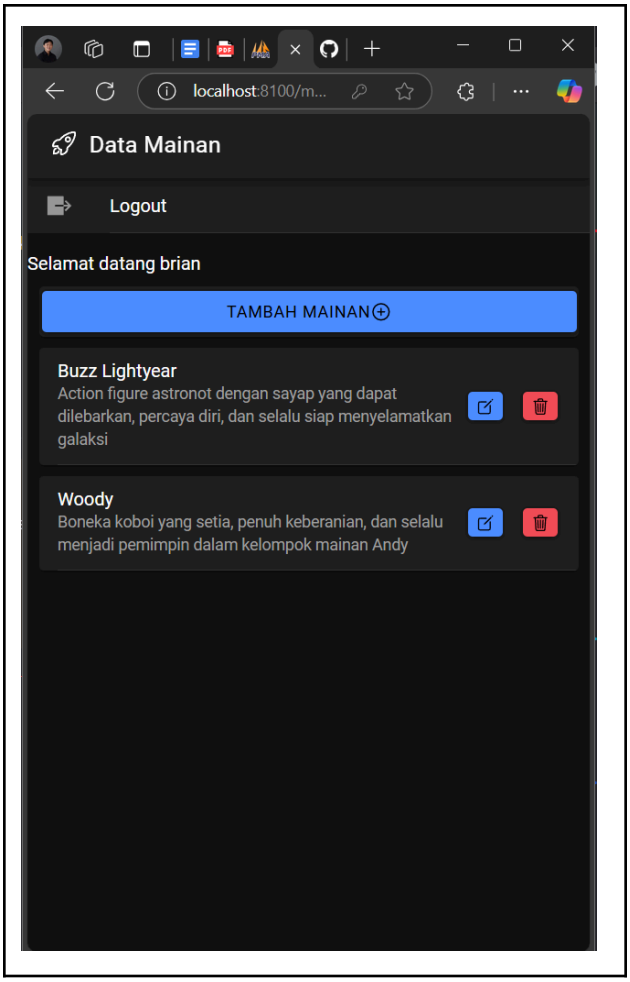
Dalam desain antarmuka login, sebuah header ion dengan judul "Login" menandai bagian atas halaman, sementara form login menggunakan komponen ion-item untuk memberikan struktur yang rapi. Input username menggunakan ion-input bertipe teks, sedangkan input password menggunakan tipe password untuk melindungi informasi sensitif. Tombol login diimplementasikan dengan ion-button berwarna primer dan mode block untuk memaksimalkan kemudahan akses.

Proses validasi input dilakukan dengan memeriksa apakah username dan password telah diisi. Jika salah satu field kosong, sistem akan menampilkan notifikasi menggunakan authentication service untuk memberi tahu pengguna bahwa kedua field wajib diisi. Mekanisme login melibatkan pengiriman data ke server melalui metode POST ke endpoint 'login.php', dengan tujuan menerima token dan username pengguna.

Penanganan response dari server dibagi menjadi dua skenario utama. Pada proses login sukses, sistem akan menyimpan token dan username, mengosongkan form, kemudian mengarahkan pengguna ke halaman mainan. Sebaliknya, jika login gagal, akan ditampilkan pesan error yang spesifik, termasuk penanganan berbagai kemungkinan masalah seperti kredensial tidak valid atau gangguan koneksi server.

Sistem menyediakan berbagai jenis notifikasi untuk membimbing pengguna, mencakup peringatan tentang field kosong, pesan kesalahan kredensial, serta informasi tentang potensi masalah koneksi server atau status layanan seperti Laragon dan XAMPP yang mungkin belum aktif.

2. READ

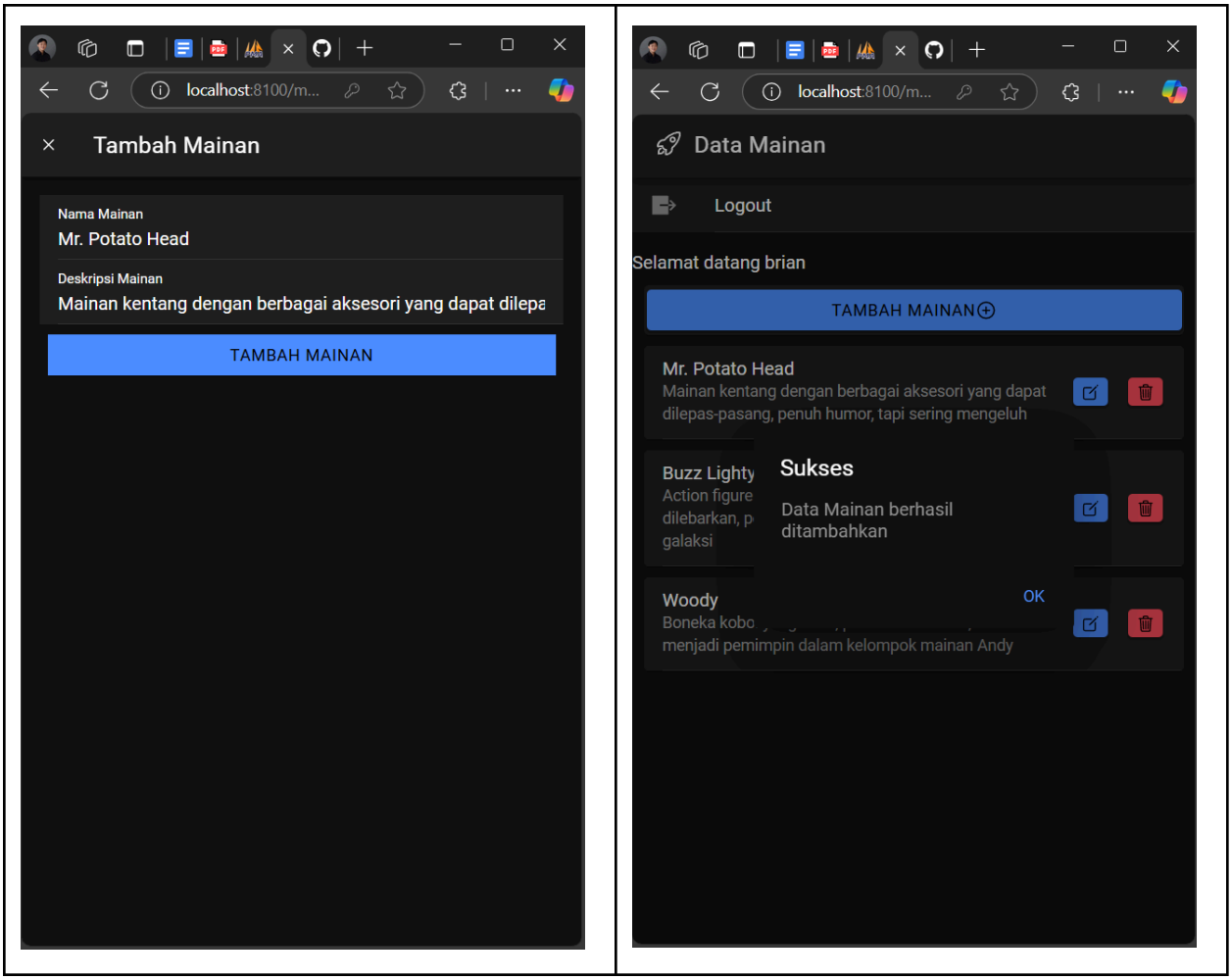


Dalam implementasi fitur Read pada aplikasi, header halaman menampilkan ikon roket di samping judul "Data Mainan", menciptakan identitas visual yang jelas. Penggunaan ionicon pada berbagai tombol aksi seperti edit dan hapus memperjelas pada antarmuka.

Bagian atas halaman menampilkan informasi pengguna yang sedang login, memberikan konteks username pengguna. Daftar mainan disajikan dalam format card yang rapi, dengan setiap card memuat detail mainan dan dilengkapi tombol aksi untuk pengeditan dan penghapusan data.

Proses pengambilan data dilakukan melalui metode GET dengan endpoint 'tampil.php'. Fungsi getMainan() mengimplementasikan logika pengambilan data.

3. CREATE



Dalam fitur Create pada aplikasi, modal tambah mainan dirancang sebagai komponen interaktif dan user-friendly. Form input dilengkapi dengan validasi untuk memastikan data yang dimasukkan memenuhi kriteria yang diperlukan.

Tombol close dan submit ditempatkan secara strategis, memungkinkan pengguna dengan mudah membatalkan atau mengirimkan data baru. Proses penambahan data mainan dilakukan melalui metode POST ke endpoint 'tambah.php', mengimplementasikan mekanisme komunikasi dengan backend secara efisien.

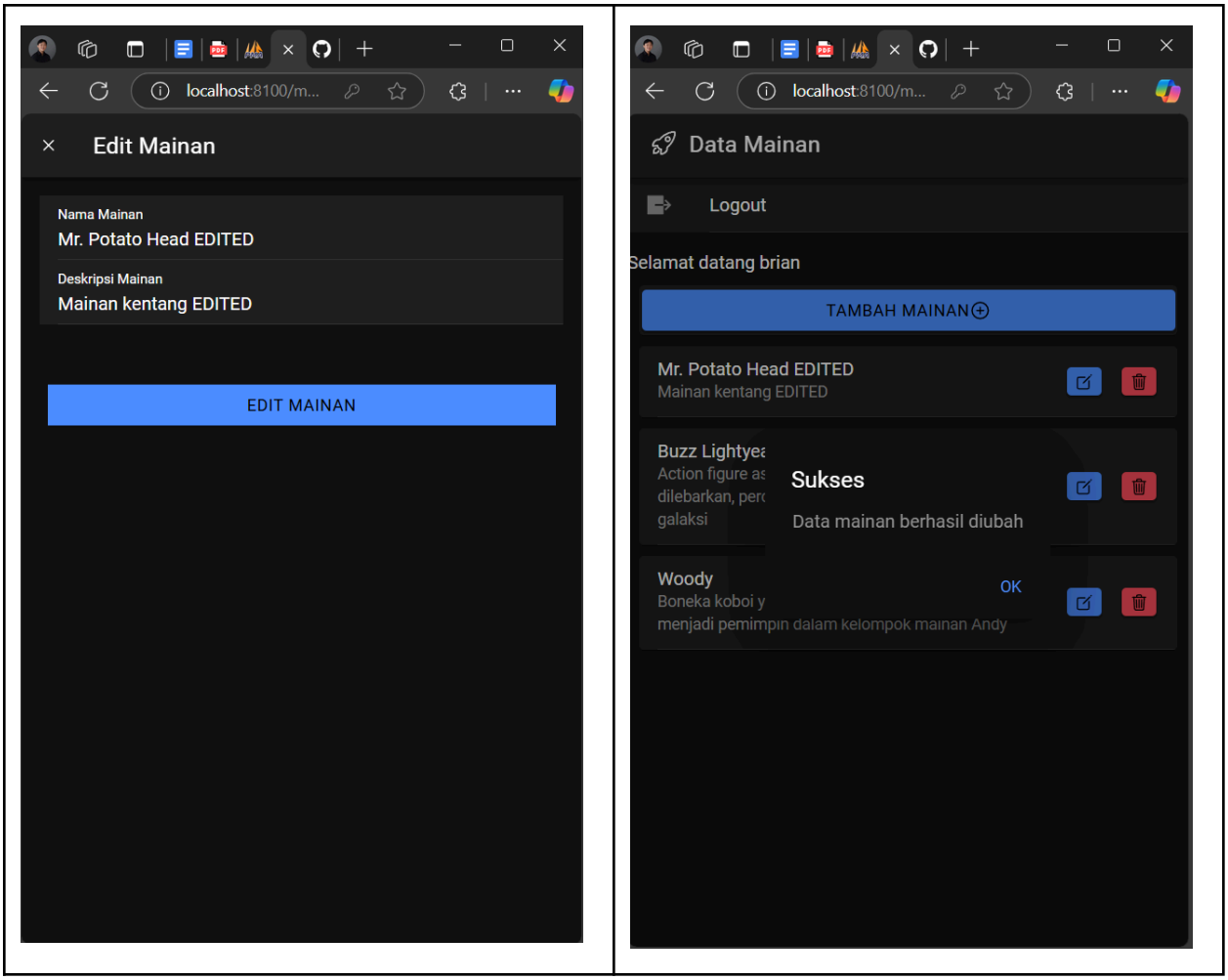
Logika tambah mainan memiliki validasi awal yang memeriksa apakah field nama dan jurusan telah diisi. Ketika syarat terpenuhi, sebuah objek data dibuat untuk dikirimkan ke server. Proses ini dirancang untuk mencegah pengiriman data yang tidak lengkap.

Sistem menyediakan mekanisme feedback yang komprehensif. Setelah berhasil menambahkan data, alert konfirmasi akan muncul, memberi kepastian kepada pengguna. Form secara otomatis

direset, mempersiapkan input berikutnya, dan daftar mainan diperbarui secara otomatis untuk segera menampilkan data terbaru.

Validasi input mencakup pengecekan field kosong, mencegah pengiriman data yang tidak valid.

4. UPDATE



Dalam fitur Update pada aplikasi, proses pengeditan data mainan dirancang dengan mekanisme pengambilan data otomatis memungkinkan form edit langsung terisi dengan informasi mainan yang dipilih, menggunakan metode PUT melalui endpoint 'edit.php'.

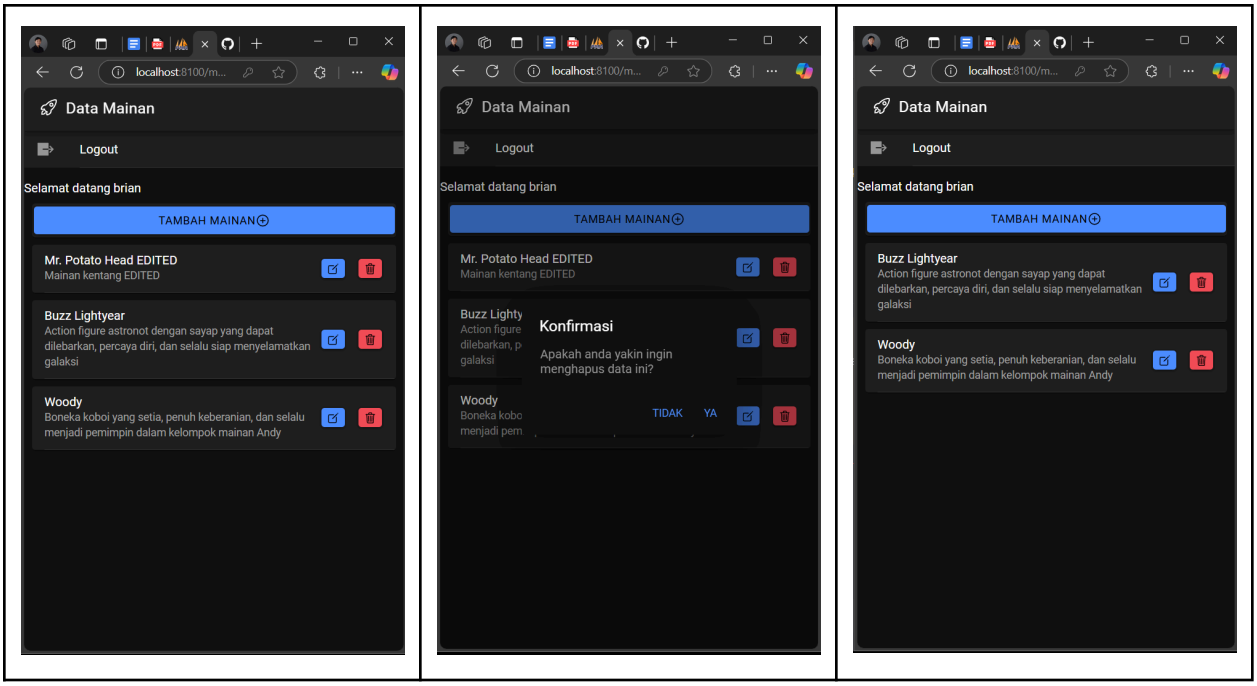
Fungsi ambilMainan() berperan penting dalam proses ini, mengambil data spesifik mainan berdasarkan ID yang dipilih. Ketika data berhasil diambil, field nama dan jurusan akan terisi secara otomatis, memudahkan pengguna untuk melakukan perubahan. Jika terjadi kegagalan pengambilan data, sistem akan mencatat error di konsol.

Validasi input menjadi komponen kunci dalam form edit. Menggunakan ion-input dengan label yang jelas, sistem memastikan bahwa field nama dan jurusan tidak boleh dibiarkan kosong. Jika validasi gagal, alert peringatan akan muncul, membimbing pengguna untuk mengisi data dengan benar.

Proses penyimpanan perubahan terjadi saat tombol "Edit Mainan" ditekan. Data yang diperbarui dikirim ke server, dengan alert sukses memberikan konfirmasi positif kepada pengguna. Setelah

proses edit berhasil, modal akan ditutup secara otomatis dan tampilan data di halaman utama akan diperbarui.

5. DELETE

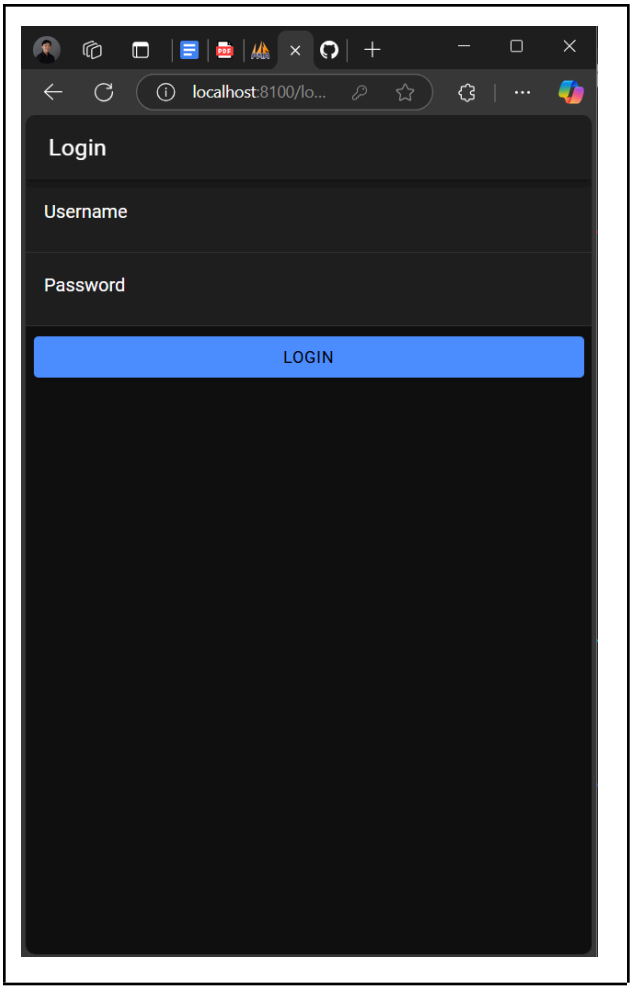


Fitur Delete dalam aplikasi dirancang dengan aman untuk menghapus data mainan. Proses dimulai dengan implementasi dialog konfirmasi dua tahap yang membantu mencegah penghapusan data secara tidak sengaja.

Ketika pengguna memilih untuk menghapus suatu data, alert konfirmasi akan muncul dengan opsi "Ya" dan "Tidak". Dialog ini memberikan kesempatan kepada pengguna untuk membatalkan tindakan sebelum data benar-benar dihapus. Jika pengguna memilih "Ya", fungsi hapusMainan() akan dieksekusi.

Proses penghapusan dilakukan melalui metode DELETE dengan endpoint 'hapus.php', menggunakan ID mainan sebagai parameter. Setelah permintaan hapus terkirim ke server, sistem secara otomatis me-refresh daftar mainan menggunakan fungsi getMainan(). Hal ini memastikan bahwa tampilan selalu menampilkan data terbaru setelah penghapusan.

6. LOGOUT



Fitur logout dirancang untuk memberikan mekanisme keluar yang aman dari aplikasi. Ketika pengguna memutuskan untuk mengakhiri sesi, fungsi logout() akan dieksekusi, yang melibatkan serangkaian tindakan penting untuk mengamankan data dan status aplikasi.

Proses logout dimulai dengan pemanggilan metode logout() dari authentication service, yang bertanggung jawab untuk menghapus token autentikasi yang tersimpan. Ini adalah langkah kritis untuk memutus akses pengguna dan melindungi informasi sensitif. Setelah token dihapus, sesi pengguna secara efektif dibersihkan, menghilangkan semua data terkait sesi yang tersimpan.

Segera setelah pembersihan data, router akan mengarahkan pengguna kembali ke halaman login menggunakan navigateByUrl('/login'). Pengalihan ini memberikan klarifikasi visual bahwa pengguna telah berhasil keluar dari sistem dan harus melakukan autentikasi ulang untuk mengakses fitur-fitur aplikasi.

Selain perpindahan halaman, proses ini juga mencakup reset state aplikasi. Ini memastikan bahwa tidak ada data sesi atau informasi pengguna sebelumnya yang tersisa, memberikan tingkat keamanan tambahan dan mencegah potensi kebocoran informasi di antara sesi yang berbeda.

Soal Tambahan:

1. Jelaskan alasan mengapa Ionic framework tidak sepopuler framework mobile lainnya!
2. Sebutkan fungsi dari komponen ion refresher dalam Ionic ?
3. Bagaimana cara mengatur agar pengguna yang sudah login tidak bisa kembali ke halaman login dan pengguna yang belum login tidak bisa masuk ke dalam halaman home atau dashboard?

Jawab:

1. Ionic Framework memang tidak sepopuler React Native atau Flutter karena beberapa kendala struktural, contohnya masih menggunakan penggunaan web technologies seperti HTML, CSS, dan Angular/TypeScript yang menjadi pedang bermata dua. Di satu sisi memberikan kemudahan bagi web developer untuk migrasi, namun di sisi lain performa aplikasi cenderung lebih lambat dibandingkan framework native. Performance overhead akibat rendering melalui WebView membuat developer yang fokus pada aplikasi dengan high-performance cenderung memilih alternatif lain seperti Flutter, yang mampu menghasilkan kode native dengan performa mendekati aplikasi asli pada platform iOS dan Android.
2. Ion-refresher adalah komponen kunci untuk implementasi pull-to-refresh mechanism. Fungsinya mirip "tarik ke bawah untuk memperbarui" pada media sosial. Ketika pengguna menarik layar ke bawah, komponen ini memicu proses pengambilan data terbaru dari server atau melakukan refresh data secara otomatis. Ini sangat berguna untuk aplikasi dengan konten dinamis seperti feed berita, daftar produk, atau data yang sering berubah.
3. Untuk mengatur akses halaman berdasarkan status login, bisa aja menggunakan route guard. Di Angular contohnya, bisa membuat AuthGuard yang akan memblokir akses ke halaman tertentu berdasarkan kondisi autentikasi. Pada route configuration, tambahkan canActivate property yang akan memvalidasi token atau status login. Jika tidak terautentikasi, guard akan langsung redirect ke halaman login. Begitu pula sebaliknya, pengguna yang sudah login akan otomatis dialihkan ke dashboard jika mencoba mengakses halaman login.