

UI/UX Enhancement: Implement Collapsible Filter Panel Across All Session Management Tabs and Move "Show Filters" Button to Header Row (Before Download/Refresh Icons) on All Session Management Tabs

10/08/2026 02:11 AM - Shreya Agarwal

Status:	New	Start date:	10/08/2026
Priority:	Normal	Due date:	
Assignee:	Shreya Agarwal	% Done:	0%
Category:		Estimated time:	0.00 hour
Target version:		Spent time:	9.00 hours
Description			
The "Show Filters" / "Hide Filters" toggle button is currently implemented only on the "Active Sessions" tab. The other tabs (Session History, Failed, Rejected, Faulted, Abandoned, Excluded, Manual) still display the filter panel expanded by default, taking full width and pushing the data table down. and Move "Show Filters" Button to Header Row (Before Download/Refresh Icons) on All Session Management Tabs			

History

#1 - 10/08/2026 09:03 AM - Shreya Agarwal

Today, I worked on enhancing the UI and optimizing network performance across all tabs within the SessionManagement.tsx page. First, I standardized and implemented the header-level "Show Filter" / "Hide Filter" toggle buttons across all seven tabs (Active Sessions, Session History, Failed, Faulted, Abandoned, Excluded, and Manual). The filter panels are set to remain collapsed by default on initial render, ensuring a clean and uncluttered interface. I also removed redundant nested "Hide filters" header links from within the individual filter panels while preserving all Advanced Filter controls, filter chips, search inputs, and event handlers.

Second, in the Failed sessions tab table, I commented out the Status column in both the table header (TableHead) and body rows (TableBody), as it was no longer required on the UI for that specific view. I preserved the original column code for seamless future re-enabling and adjusted the empty and loading state column spans (colSpan) accordingly to prevent any visual misalignment or table layout issues.

Third, I optimized network performance by implementing lazy-loading for the three shared filter dropdown APIs: Filter by Station (getAllChargingStations), Filter by Charger (loadAvailableMachines), and Charger Tag (companyChargerTagApi.list). Previously, these three endpoints were being triggered eagerly on initial page mount. I removed their initial mount calls from useEffect and initializeData(), and consolidated them into a unified loader (loadSharedFilterDropdownOptions). These APIs now execute on-demand only when the user clicks the "Show Filter" button on any tab.

To ensure optimal performance, I implemented a shared guard reference (filterOptionsLoadedRef) that guarantees these three APIs are called strictly once across the entire application session. Toggling between "Show Filter" and "Hide Filter", or switching across tabs (such as navigating between Active, Failed, History, Faulted, etc.), directly reuses the dropdown data stored in state without re-fetching or triggering redundant network requests.

On the new branch, I focused on cleaning up redundant network calls, standardizing table column headers, and ensuring consistent count displays across all tabs. First, I eliminate unnecessary network overhead, commented out the await loadAvailableLocations() call inside initializeData(), effectively stopping the redundant request on page mount while preserving the underlying loader function for future use if ever required. Second, standardized the energy column header across the application to achieve full visual consistency. Across four tabs—Active Sessions, Session History, Excluded, and Manual—we updated the table column label from Energy Consumed (kWh) to KWH consumed. This aligns the header text directly with the naming convention already present in the Failed, Rejected, Faulted, and Abandoned tabs. Third, enhanced the Session History tab header by introducing the dynamic record count display. Previously, the section title only displayed Session History without a count. updated the heading typography to dynamically bind the existing state value {state.totalHistoryCount}, rendering it as Session History ({state.totalHistoryCount}).