

# WorkOrders Page Runtime Issue - Investigation Report

---

**Date:** December 17, 2025

**Project:** mechanics-app-final

**Issue:** WorkOrders page renders layout but not content

---

## Executive Summary

---

The WorkOrders page fails to render content due to **multiple Supabase client instances** being initialized across the application. This causes the warning “Multiple GoTrueClient instances detected” and creates unpredictable behavior in Vue components relying on Supabase authentication state.

---

## Issues Identified

---



### CRITICAL ISSUE #1: Multiple Supabase Client Instances

**Location:**

- `src/supabaseClient.js`
- `src/newSupabaseClient.js`
- `SupabaseClient.js` (root - compiled file, not used)

**Problem:**

The application has TWO active Supabase client files creating separate instances:

**Files importing from `supabaseClient.js` (MAJORITY):**

- `src/layouts/MainLayout.vue`
- `src/views/JobsView.vue`
- `src/views/CalendarView.vue`
- `src/views/CustomersView.vue`
- `src/views/VehiclesView.vue`
- `src/views/SuppliersView.vue`
- `src/views/InvoicesListView.vue`
- `src/views/InvoiceEditorView.vue`
- `src/views/JobArchiveView.vue`
- `src/views/AdminView.vue`
- `src/views/AdminSettingsView.vue`
- `src/views/LoginView.vue`
- `src/components/JobEditor.vue`
- `src/components/JobCard.vue`
- `src/components/CustomerEditor.vue`
- `src/components/SupplierEditor.vue`
- `src/components/VehicleEditor.vue`
- `src/components/PartsUsageEditor.vue`
- `src/composables/useVehicles.js`

**Files importing from `newSupabaseClient.js` (MINORITY):**

- `src/main.js` ⚠️
- `src/router/index.js` ⚠️
- `src/components/WorkOrderList.vue` ⚠️
- `src/views/WorkOrderDetailView.vue` ⚠️
- `src/views/WorkOrderCreate.vue` ⚠️

**Impact:**

- Multiple GoTrueClient instances compete for auth state
- Authentication state becomes inconsistent
- Components may fail silently due to auth mismatch
- Router guards may not work correctly
- User session management becomes unpredictable

**Root Cause:**

The WorkOrders feature was added later and used `newSupabaseClient.js` instead of the existing `supabaseClient.js`, creating a split in the codebase.

**🟡 ISSUE #2: No Error Handling in WorkOrderList Component**

**Location:** `src/components/WorkOrderList.vue` (line 48-60)

**Current Code:**

```
const fetchWorkOrders = async () => {
  const { data, error } = await supabase
    .from('work_orders')
    .select('*')
    .order('created_at', { ascending: false })

  if (error) {
    console.error('Error fetching work orders:', error)
  } else {
    workOrders.value = data
  }
  loading.value = false
}
```

**Problem:**

- Error is only logged to console, no user feedback
- If the component fails to mount due to client mismatch, error is silent
- No try-catch wrapper for unexpected errors
- No empty state handling for zero work orders

**🟡 ISSUE #3: Missing Error Boundaries and Debugging****Locations:**

- `src/views/WorkOrdersView.vue`
- `src/components/WorkOrderList.vue`
- `src/views/WorkOrderDetailView.vue`

**Problem:**

- No error boundary to catch component rendering errors
  - No console logging for component lifecycle (mount/unmount)
  - Silent failures make debugging difficult
  - No loading or error states in WorkOrdersView.vue wrapper
- 

## ● ISSUE #4: Router Configuration (WORKING)

**Location:** `src/router/index.js` (line 34-35)

**Current Code:**

```
{ path: 'work-orders', name: 'work-orders', component: WorkOrdersView },  
{ path: 'work-orders/:id', name: 'work-order-detail', component: WorkOrderDetailView,  
  props: true },
```

**Status:** ☒ Routes are configured correctly with proper params

---

## ● ISSUE #5: WorkOrderDetailView (MOSTLY WORKING)

**Location:** `src/views/WorkOrderDetailView.vue`

**Status:** This component is properly structured with:

- ☒ Proper `onMounted` wrapper (line 299-301)
- ☒ Error handling in async functions
- ☒ Loading state with spinner
- ☒ Validation for form inputs
- ☒ Try-catch blocks in `addPart` function

**Minor Issue:** Uses `newSupabaseClient.js` instead of standard client

---

## Root Cause Analysis

### Why WorkOrders Page Shows Blank Content:

1. **Auth State Mismatch:** The router guard uses `newSupabaseClient.js`, authenticates the user, and allows navigation to WorkOrders
  2. **Component Auth Failure:** `WorkOrderList` component tries to use `newSupabaseClient.js` which may not have the same auth state as the main app (which uses `supabaseClient.js`)
  3. **Silent Failure:** Without proper error handling, the component fails to fetch data but doesn't show any error to the user
  4. **Layout Still Renders:** `MainLayout` uses `supabaseClient.js` and works fine, so sidebar and header appear
  5. **Content Area Blank:** `WorkOrderList` component fails silently, leaving the content area empty
-

## Recommended Fixes

---

### FIX #1: Consolidate Supabase Clients (CRITICAL - Must Do First)

**Action:** Standardize on ONE Supabase client file

**Option A: Use `supabaseClient.js` (Recommended)**

```
# Delete the duplicate client
rm src/newSupabaseClient.js

# Update all files importing from newSupabaseClient
# Replace: from '@newSupabaseClient'
# With:    from '@supabaseClient'
```

**Files to Update:**

1. `src/main.js` - line 7
2. `src/router/index.js` - line 2
3. `src/components/WorkOrderList.vue` - line 41
4. `src/views/WorkOrderDetailView.vue` - line 175
5. `src/views/WorkOrderCreate.vue` - (check line importing supabase)

**Option B: Use `newSupabaseClient.js`**

```
# Delete the old client
rm src/supabaseClient.js

# Update ALL other files to use newSupabaseClient
# (NOT RECOMMENDED - too many files to change)
```

---

### FIX #2: Add Error Handling to WorkOrderList

**File:** `src/components/WorkOrderList.vue`

**Replace lines 48-60 with:**

```

const fetchWorkOrders = async () => {
  try {
    console.log('🔄 Fetching work orders...')
    const { data, error } = await supabase
      .from('work_orders')
      .select('*')
      .order('created_at', { ascending: false })

    if (error) {
      console.error('❌ Error fetching work orders:', error)
      throw error
    }

    console.log('✅ Work orders fetched:', data?.length || 0, 'items')
    workOrders.value = data || []
  } catch (error) {
    console.error('❌ Unexpected error in fetchWorkOrders:', error)
    // Show error to user
    workOrders.value = []
  } finally {
    loading.value = false
  }
}

```

**Add error state to template** (after line 5):

```

<!-- Error state -->
<div v-else-if="error" class="error-message">
  <p>Failed to load work orders. Please try refreshing the page.</p>
  <button @click="fetchWorkOrders">Retry</button>
</div>

<!-- Empty state -->
<div v-else-if="workOrders.length === 0" class="empty-state">
  <p>No work orders found.</p>
</div>

```

**Add error ref** (after line 45):

```

const error = ref(null)

```

## FIX #3: Add Lifecycle Logging

**File:** src/components/WorkOrderList.vue

**Add after onMounted** (after line 64):

```
onMounted(() => {  
  console.log('🚀 WorkOrderList component mounted')  
  fetchWorkOrders()  
})  
  
// Add this new hook  
onUnmounted(() => {  
  console.log('👋 WorkOrderList component unmounted')  
})
```

**Don't forget to import:**

```
import { ref, onMounted, onUnmounted } from 'vue'
```

---

## **FIX #4: Add Error Boundary to WorkOrdersView**

**File:** `src/views/WorkOrdersView.vue`

**Replace entire file with:**

```

<template>
  <div class="work-orders-view">
    <Suspense>
      <WorkOrderList />
      <template #fallback>
        <div class="loading">Loading work orders...</div>
      </template>
    </Suspense>
  </div>
</template>

<script setup>
import { onMounted, onErrorCaptured } from 'vue'
import WorkOrderList from '@/components/WorkOrderList.vue'

onMounted(() => {
  console.log('📄 WorkOrdersView mounted')
})

onErrorCaptured((err, instance, info) => {
  console.error('❌ Error captured in WorkOrdersView:', err)
  console.error('Component info:', info)
  console.error('Instance:', instance)
  return false // Propagate error to parent
})
</script>

<style scoped>
.work-orders-view {
  padding: 1rem;
}

.loading {
  text-align: center;
  padding: 2rem;
  font-size: 1.2rem;
  color: #666;
}
</style>

```

## FIX #5: Update WorkOrderDetailView Import

File: `src/views/WorkOrderDetailView.vue` (line 175)

Change:

```

// Old
import { supabase } from '@/newSupabaseClient.js'

// New
import { supabase } from '@/supabaseClient.js'

```

## Implementation Priority

---

### Phase 1: Critical Fixes (Must Do Immediately)

1. ☒ Consolidate Supabase clients - Fix #1
2. ☒ Update all imports to use single client

### Phase 2: Error Handling (High Priority)

1. ☒ Add error handling to WorkOrderList - Fix #2
2. ☒ Add lifecycle logging - Fix #3
3. ☒ Add error boundary to WorkOrdersView - Fix #4

### Phase 3: Testing (After Fixes)

1. ☒ Test WorkOrders list page
  2. ☒ Test WorkOrder detail page
  3. ☒ Test navigation and auth flow
  4. ☒ Verify console warnings are gone
- 

## Testing Checklist

---

After implementing fixes:

- ☐ No “Multiple GoTrueClient instances” warning in console
  - ☐ WorkOrders page shows list of work orders
  - ☐ Loading state appears while fetching
  - ☐ Error messages show if fetch fails
  - ☐ Empty state shows if no work orders exist
  - ☐ Clicking work order ID navigates to detail page
  - ☐ Detail page loads work order data
  - ☐ Can add parts to work order
  - ☐ Can start/complete jobs
  - ☐ All lifecycle logs appear in console
  - ☐ No silent errors in console
- 

## Files Requiring Changes

---

### High Priority:

1. `src/components/WorkOrderList.vue` - Add error handling and logging
2. `src/views/WorkOrdersView.vue` - Add error boundary
3. `src/views/WorkOrderDetailView.vue` - Update import
4. `src/views/WorkOrderCreate.vue` - Update import
5. `src/main.js` - Update import
6. `src/router/index.js` - Update import
7. `src/newSupabaseClient.js` - DELETE this file



## Low Priority (Optional Cleanup):

1. `SupabaseClient.js` (root) - DELETE if not used elsewhere

## Expected Behavior After Fixes

### Before Fixes:

- ❌ WorkOrders page shows blank content
- ❌ "Multiple GoTrueClient instances" warning
- ❌ Silent errors in console
- ❌ No user feedback on errors

### After Fixes:

- ✅ WorkOrders page shows list of work orders
- ✅ No Supabase warnings
- ✅ Clear console logging for debugging
- ✅ User-friendly error messages
- ✅ Loading and empty states work correctly
- ✅ Consistent auth state across all components

## Additional Recommendations

### 1. Add Global Error Handler

File: `src/main.js`

```
app.config.errorHandler = (err, instance, info) => {
  console.error('Global error:', err)
  console.error('Component:', instance)
  console.error('Info:', info)
}
```

### 2. Add Request Logging

Create `src/utils/supabaseLogger.js`:

```
export function logSupabaseRequest(table, operation, result) {
  if (result.error) {
    console.error(`❌ ${operation} on ${table}:`, result.error)
  } else {
    console.log(`✅ ${operation} on ${table}:`, result.data)
  }
}
```

### 3. Create Composable for Work Orders

Create `src/composables/useWorkOrders.js`:

```

import { ref } from 'vue'
import { supabase } from '@/supabaseClient'

export function useWorkOrders() {
  const workOrders = ref([])
  const loading = ref(false)
  const error = ref(null)

  const fetchWorkOrders = async () => {
    loading.value = true
    error.value = null

    try {
      const { data, error: fetchError } = await supabase
        .from('work_orders')
        .select('*')
        .order('created_at', { ascending: false })

      if (fetchError) throw fetchError

      workOrders.value = data || []
    } catch (e) {
      error.value = e
      console.error('Error fetching work orders:', e)
    } finally {
      loading.value = false
    }
  }

  return {
    workOrders,
    loading,
    error,
    fetchWorkOrders
  }
}

```

---

## Conclusion

The root cause of the blank WorkOrders page is the **multiple Supabase client instances** creating auth state inconsistency. The fix is straightforward: consolidate to a single client and add proper error handling.

**Estimated Time to Fix:** 15-30 minutes

**Risk Level:** Low (changes are straightforward and well-documented)

**Impact:** High (fixes critical user-facing issue)

---

## Next Steps

1. Review this report
2. Backup current code: `git commit -am "Before fixing WorkOrders issue"`
3. Implement Fix #1 (consolidate clients)

4. Test the page
  5. Implement Fixes #2-4 (error handling)
  6. Test thoroughly
  7. Commit changes: `git commit -am "Fix WorkOrders blank page issue"`
- 

**Report Generated:** December 17, 2025

**Investigator:** DeepAgent AI

**Status:** Ready for Implementation