DMR – Project Issues & Architecture Review

This document provides a detailed technical review of the current state of the DMR-New-learning-public project. The objective is to identify architectural, security, session-handling, and user-management issues that may affect reliability, security, scalability, or maintainability. This review is written to support incremental, non-breaking fixes.

1. User Management – Critical Issues

The project currently contains multiple overlapping implementations of user management logic. Files such as user_management.py, modules/auth/auth.py, modules/auth_manager.py, and landing_page.py independently read and validate user data. This results in multiple sources of truth for users, roles, and status flags. In production, this can cause inconsistent behavior where a user may be authenticated in one flow but rejected or misclassified in another.

Impact: Incorrect role assignment, admin privileges leaking to non-admin users, inactive users being able to log in, and unpredictable authentication behavior.

2. User Status Enforcement Gaps

User status (active / inactive) is not consistently enforced during login or persistent session restoration. Once a user has logged in and a persistent session exists, the application trusts the stored session without revalidating the user's current status in the database. This allows inactive or administratively disabled users to continue accessing the system after logout, refresh, or app restart.

Impact: Security risk, compliance violations, and inability for administrators to effectively disable user access.

3. Persistent Session Restore Bypasses Authentication Checks

Persistent session restoration logic restores username and role directly from session storage without validating that the user still exists or that their role and subscription state are still valid. The restore flow effectively bypasses standard authentication safeguards.

Impact: Deleted users, downgraded users, or expired subscribers may regain access without reauthentication.

4. Subscription Gating Occurs Too Late in the UI Flow

Subscription checks are performed after portions of the UI have already rendered. In some cases, subscriber-only tabs or data-loading logic executes before subscription validation completes. This creates a risk of feature leakage and unnecessary backend processing for unauthorized users.

Impact: Paid features may be partially visible or accessible to non-paying users.

5. Password Reset and Session Invalidation Weaknesses

Password reset functionality does not invalidate existing sessions or persistent logins. Additionally, password complexity rules are minimal, and there is no audit logging for credential changes. This increases the risk of account takeover scenarios.

Impact: Security exposure and reduced auditability.

6. Data Files Mixed with Source Code

Runtime data files such as users_database.json, subscriptions_database.json, and persistent_sessions.json are currently present alongside source code. While functional for local development, this approach complicates deployment, increases risk of data leakage, and makes scaling or migration to a proper database more difficult.

Impact: Operational risk and poor separation of concerns.

7. Summary & Recommended Direction

The project is functionally rich and well-structured from a feature standpoint; however, the authentication, user management, and session layers require consolidation and hardening. All identified issues can be resolved through targeted, incremental patches without redesigning the system. Priority should be given to establishing a single source of truth for users, enforcing status and subscription checks at all authentication boundaries, and tightening persistent session validation.