LLM Automated Evaluation Results

Functional Requirement Evaluation Table

The following presents a complete evaluation of the asset management system’s requirement document, based on the user’s specifications. The evaluation process strictly adhered to the following rules:

Data Entity Extraction: Only the data entities that the system needs to manage (as derived from the functional requirements document) are counted, excluding any external data sources or formats mentioned in interface-related content. A total of 10 data entities were identified: Asset, User, Department, OperationLog, ApprovalTask, MaintenanceRecord, AssetCategory, AssetTag, ApprovalFlow, and Supplier.

Functional Requirement Consolidation: Requirements involving the same data entity and functional point were merged to avoid over-consolidation (only highly similar requirements were merged, ensuring each functional point remains distinct). Generic system setting requirements (such as preferences or multilingual support) were excluded; however, such items were not present in the document. After consolidation, the number of functional requirements was reduced from 37 to 33. The consolidation details are as follows:

FR-02 (Asset Transfer) and FR-27 (Asset Allocation Approval) were merged into “Asset Transfer/Allocation”.

FR-03 (Asset Return) and FR-36 (Asset Return Confirmation) were merged into “Asset Return”.

FR-05 (Asset Disposal Approval) and FR-29 (Asset Damage Handling) were merged into “Asset Disposal/Damage”.

FR-09 (Department Creation) and FR-10 (Department Information Maintenance) were merged into “Department Maintenance”.

All other requirements remained unmerged to maintain clarity of core functions.

Hallucination Requirement Judgment: The standard for determining hallucination was whether the operations involved data entities that could not be linked to any existing system entities, resulting in a disconnected E-R diagram (broken weak connectivity). If not judged as hallucination, further evaluations were conducted.

Evaluation Strictness: Each requirement was assessed rigorously. Hallucinated requirements were excluded from completeness and testability evaluations. As the document was of high quality, most requirements met the standards. However, a few requirements had minor issues (such as slightly mismatched input/output), causing some cases of incompleteness or untestability.

Evaluation Table

｜Functional Requirement｜Hallucination｜Evaluation Justification｜

｜Asset Registration｜No｜Involves the Asset and OperationLog entities; input/output definitions are clear, business path is complete, and implementation is feasible.｜

｜Asset Transfer/Allocation｜No｜Related to Asset and ApprovalTask entities; input/output logic is clear, behavior aligns with system processes.｜

｜Asset Return｜No｜Involves approval and asset status updates; the description is realistic and the workflow is reasonable with no redundancy.｜

｜General Asset Status Change｜No｜Asset status change logic is clearly defined, all participating entities are valid, and the structure stays within system core.｜

｜Asset Disposal/Damage｜No｜Involves approval, logging, and asset updates; workflow is feasible and introduces no abnormal paths.｜

｜User Registration｜No｜Registration process aligns with the User entity, and operational steps follow standard system design.｜

｜User Permission Assignment｜No｜Permission granting logic is tightly connected with the relevant entities, operation is valid and contains no redundant structures.｜

｜User Information Update｜No｜Modification actions match user management logic without introducing fabricated fields or behavior.｜

｜Department Maintenance｜No｜Creation and editing processes are reasonable, entities are well-associated, and no isolated elements exist.｜

｜Asset Query｜No｜Query structure is based on defined entities, filter conditions and result logic are coherent.｜

｜Asset Report Generation｜No｜Report structure is clear, fields are integrated with system modules, and the generation process is realistic and feasible.｜

｜Operation Log Query｜No｜Query is based on the OperationLog entity, conditions are clear, and behavior stays within the actual system.｜

｜Approval Process Initiation｜No｜Approval structure aligns with the ApprovalTask entity, triggering mechanism is logical and the process is complete.｜

｜Approval Task Handling｜No｜Involves approval, asset, and log entities; description is clear and behavior is controllable.｜

｜Asset Import/Export｜No｜File processing is clearly linked to the system structure, and data interaction logic is complete.｜

｜Permission Request Review｜No｜Review process is well-defined, approval and permissions are properly associated, and no fake operations are involved.｜

｜Department Asset Statistics｜No｜Query dimensions are reasonable, statistical logic coordinates with department structure, and no information is missing.｜

｜Asset Usage Record Viewing｜No｜Usage records are derived from logs, action chains are complete, and the structure is grounded.｜

｜Asset Maintenance Registration｜No｜Involves asset, log, and approval entities, and the process structure is rigorous.｜

｜Asset Depreciation Calculation｜No｜Depreciation logic is clearly defined, field updates align with asset properties, and no operation exceeds system scope.｜

｜Asset Usage Confirmation｜No｜Usage behavior is well-defined, related status transitions are compliant, and no fabricated paths are introduced.｜

｜Asset Inventory Management｜No｜Inventory process is standardized, data paths are regulated, and module coordination is proper.｜

｜Asset History Traceability｜No｜Traceability is based on the log entity, paths are closed, and the behavior has a retrievable basis.｜

｜Asset Category Maintenance｜No｜Category structure is compliant, and related operations do not detach from the core asset entity.｜

｜Asset Tag Management｜No｜Tag management is logically bound to assets, structure is clear, and no broken links are present.｜

｜Asset Valuation｜No｜Valuation logic is reasonable, field updates are compliant, and no detachment from operation flow occurs.｜

｜System Permission Auditing｜No｜Audit behavior originates clearly, user and permission structures are matched, and the process is feasible.｜

｜Multi-dimensional Report Export｜No｜Export conditions and logic are standardized, data generation is coherent, and no exaggeration exists.｜

｜Custom Approval Flow Configuration｜No｜Configuration actions are based on approval structure, system supports its flexibility, and no boundary overreach occurs.｜

｜Department Merging and Splitting｜No｜Operation involves Asset and Department entities, responsibility division is clear, and paths are realistic.｜

｜Batch Permission Granting｜No｜Batch operations are structurally compliant, user selection logic is defined, and no logical risks are introduced.｜

｜Asset Borrowing Approval｜No｜Borrowing and approval behaviors are tightly integrated, involved entities are valid, and no fictitious paths are introduced.｜

｜Supplier Management｜No｜Management objects are clearly defined, maintenance path is standard, and no structural illusion exists.｜

Summary

The document involves 10 data entities: Asset, User, Department, OperationLog, ApprovalTask, MaintenanceRecord, AssetCategory, AssetTag, ApprovalFlow, and Supplier.

After consolidation based on functional similarity, the number of functional requirements is 33 (originally 37, reduced by 4 through merging).

Number of hallucinated requirements: 0 — All functional requirements are associated with existing data entities, and the E-R diagram remains weakly connected (no isolated entities).

Number of non-hallucinated requirements: 33 — No hallucinated requirements were identified.

Note:

The hallucinated requirement count is 0. All operations in the requirements involve defined and interrelated data entities (e.g., MaintenanceRecord is defined in FR-20 and linked to Asset; Supplier is defined in FR-37), ensuring a connected E-R diagram.

Evaluation was strict: Despite the high quality of the document, each requirement was scrutinized. No major issues (e.g., missing inputs/outputs) were found. A few requirements (e.g., FR-19: Asset Usage Record Viewing) relied on implicit entities (OperationLog), but their associations were explicitly stated in the document and thus not hallucinated; they are complete and testable.