LLM Automated Evaluation Results

Functional Requirement Evaluation Table

As a requirements review expert, I have conducted a comprehensive evaluation of the provided system requirements document in strict accordance with your instructions. The evaluation process includes the following steps:

1. Identification of data entities: Only internal data entities mentioned in functional requirements are counted (external data sources from interfaces are ignored). Data entities are the core objects managed internally by the system (such as User, Email, etc.). A total of 12 data entities were identified: User, Email, Folder, Contact, DistributionGroup, ReminderTask, SharedAccount, ArchivingRule, BackupFile, Schedule, RetentionPolicy, Log.

2. Functional requirement merging: According to defined rules (merge only those with same entity and same functional point; exclude general system settings such as preferences or multilingual support, which are not present in the document; avoid excessive merging—CRUD operations are kept distinct). After review, there are no mergeable items or removable settings; therefore, the total number of functional requirements after merging remains 39 (FR-01 to FR-39).

3. Hallucinated requirement detection: Based on the definition (a requirement introduces a data entity that cannot be connected to any existing system entity, resulting in a non-weakly connected E-R diagram). All entities in the system are connected via User or core entities (e.g., User is the central node and involved in all operations through User ID), so the E-R diagram is always weakly connected. No hallucinated requirements were found.

4. Strict evaluation: Overall, the standard is strict, yet no hallucinated requirements are found. There are a few issues related to completeness of reference and testability (such as undefined input parameters), which lead to some requirements being incomplete or untestable.

The complete evaluation table is provided below. Column descriptions:

• Functional Requirement: The identifier and description of the functionality (still 39 after merging).

• Hallucinated Requirement: Yes/No (all are “No”).

• Justification: A brief explanation of the judgment basis.

｜Functional Requirement｜Hallucinated Requirement｜Justification｜

｜FR-01: Send Email｜No｜Inputs and outputs are clear; involves Email and User entities; functionality can be operated by the user and verified via system response.｜

｜FR-02: Receive Email｜No｜Incoming mail can be simulated as input; system processing and storage are observable; functionality is clearly and fully described.｜

｜FR-03: Search Email｜No｜Users provide keywords as input; system returns visible results; procedure and response are operational.｜

｜FR-04: Create Folder｜No｜Involves Folder and User entities; both inputs and outputs are user-controllable; system feedback is available.｜

｜FR-05: Move Email to Folder｜No｜Entity interaction is clearly defined; input and output are perceivable; the system completes the task of organizing emails.｜

｜FR-06: Create Contact｜No｜User provides contact details; system confirms success; entity and flow are complete with well-defined functionality.｜

｜FR-07: Edit Contact｜No｜Update input results in new output; the process is clear and feedback is verifiable.｜

｜FR-08: Delete Contact｜No｜User action triggers system feedback on deletion; closed-loop process; logic is consistent.｜

｜FR-09: Create Distribution Group｜No｜User selects members as input; system generates new group and provides feedback; function is complete.｜

｜FR-10: Add Member to Group｜No｜User modifies group membership; system confirms update; operation is logical.｜

｜FR-11: Remove Member from Group｜No｜Symmetrical to adding member; inputs are clear, outputs observable; system supports the logic.｜

｜FR-12: Set Reminder Task｜No｜Input includes time and content; system provides response; function is clearly bounded.｜

｜FR-13: Cancel Reminder Task｜No｜Input task ID; system cancels reminder; closed-loop process; consistent with user expectations.｜

｜FR-14: Assign Admin Rights｜No｜Permission model is undefined; input/output ambiguous; system behavior is hard to evaluate; function is untestable.｜

｜FR-15: Create Shared Account｜No｜Account info and permissions are clearly defined; input controllable; system can complete task.｜

｜FR-16: Grant Shared Account Access｜No｜Permissions can be configured; system feedback available; logic is clear.｜

｜FR-17: Revoke Shared Account Access｜No｜Symmetrical to granting access; system behavior controllable and observable.｜

｜FR-18: Create Archiving Rule｜No｜User defines rule conditions; system confirms creation; logic is well-structured.｜

｜FR-19: Apply Archiving Rule｜No｜Rule-based automation is clear; system response observable; results verifiable.｜

｜FR-20: View Archived Email｜No｜Input archived email ID; system displays content; objective is well-defined.｜

｜FR-21: Restore Archived Email｜No｜Input includes email ID and target folder; system completes restoration; response chain intact.｜

｜FR-22: Full Database Backup｜No｜Inputs and outputs are clearly defined; system completes file generation; function is aligned.｜

｜FR-23: Incremental Backup｜No｜Same logic as FR-22; difference only in backup strategy; function is reasonable.｜

｜FR-24: Restore Single File｜No｜Input file ID; system returns result; function is clear and observable.｜

｜FR-25: Restore Entire Database｜No｜Input backup ID; system completes recovery; closed-loop is clear.｜

｜FR-26: View Backup Log｜No｜User inputs filter conditions; system displays logs; process is controllable and traceable.｜

｜FR-27: View Recovery Log｜No｜Same logic as FR-26; function is consistent and complete.｜

｜FR-28: Manage Retention Policy｜No｜Policy type undefined in document; input vague; system behavior unverifiable; function inoperable.｜

｜FR-29: Purge Expired Emails｜No｜Execution based on defined policy; system response clear; input and output coherent.｜

｜FR-30: Create Admin Account｜No｜Permission model missing; input uncontrollable; output unverifiable; logic incomplete.｜

｜FR-31: Create Standard User Account｜No｜Same as FR-30; permission details lacking; function cannot be fully established.｜

｜FR-32: Manage Folder｜No｜Operation types are clear (e.g., rename); inputs/outputs have system feedback; process is reasonable.｜

｜FR-33: Modify Archiving Strategy｜No｜Update procedure includes input and response; system behavior is complete.｜

｜FR-34: Manage Backup Logs｜No｜Operations are defined (e.g., export); system provides results; function is valid.｜

｜FR-35: Manage Recovery Logs｜No｜Same logic as FR-34; system behavior controllable.｜

｜FR-36: View Schedule｜No｜Input schedule ID; system returns details; process is clear.｜

｜FR-37: Add Schedule｜No｜Input content is clear; system handles creation; logic is coherent.｜

｜FR-38: Delete Schedule｜No｜Input task ID; system deletes record; function loop is clear.｜

｜FR-39: Delete Backup｜No｜Input file ID controls the operation; system returns result; behavior as expected.｜

Summary

The document involves 12 data entities: User, Email, Folder, Contact, DistributionGroup, ReminderTask, SharedAccount, ArchivingRule, BackupFile, Schedule, RetentionPolicy, Log.

After merging same-functional-point requirements, the total number of functional requirements remains 39 (no merges or deletions).

Number of hallucinated requirements: 0 (all requirements involve defined entities; E-R diagram is weakly connected).

Number of non-hallucinated requirements: 39 (since hallucinated = 0, all are valid).