LLM Automated Evaluation Results

Functional Requirement Evaluation Table

According to the system overview, functional requirements, and external interface documents provided by the user, I conducted a detailed evaluation as a requirements review expert. The evaluation process strictly followed the steps below:

1. Data Entity Extraction: Core data entities to be managed within the system were extracted from the functional requirement documents (excluding external data sources or formats mentioned in interface descriptions). Based on the functional requirement descriptions, a total of 12 data entities were identified: MailRecord, Folder, Group, Contact, EmailAccount, ArchivePolicy, MailStream, BackupFile, BackupLog, RestoreLog, OperationLog, and Reminder.

2. Functional Requirement Consolidation: Based on the user-specified rules, similar functional points were consolidated:

• Rule 1: Functional points on the same data entity with similar operations were merged into one requirement (e.g., add, edit, delete Contact were merged into “Manage Contact”).

• Rule 2: Generic system configuration requirements (such as preference settings and multilingual support) were removed; however, no such requirements were found in the document, so none were removed.

• After merging, the original 17 functional requirements were consolidated into 15 (FR-08, FR-09, FR-10 were merged into one under contact management).

• The merged list is as follows (original FR numbers retained for traceability):

• Register Mail (FR-01)

• Send Mail (FR-02)

• Receive Mail (FR-03)

• Format Mail (FR-04)

• Search Mail (FR-05)

• Organize Mail Folders (FR-06)

• Manage Group (FR-07)

• Manage Contact (merged FR-08, FR-09, FR-10)

• Manage Email Account (FR-11)

• Configure Archive Policy (FR-12)

• Execute Mail Archiving (FR-13)

• Capture Mail Stream (FR-14)

• Manage Backup (FR-15)

• View Backup Log (FR-16)

• Manage Restore Log (FR-17)

3. Hallucination Requirement Judgment:

• Criterion: Functional operations referencing data entities that cannot be associated with existing entities result in isolated nodes on the E-R diagram, and are thus considered hallucinated requirements. Only the 15 merged requirements were evaluated.

• Current Entity Network: MailRecord is the core, connected to Folder, Group, Contact, and EmailAccount. ArchivePolicy links Mail, MailStream, and Backup. Log entities (OperationLog, BackupLog, RestoreLog) log all operations. Reminder is connected to Contact. All entities are path-connected (e.g., MailStream is connected to Mail via ArchivePolicy, Backup is linked via EmailAccount).

• Evaluation Result: No hallucinated requirements. All requirements involve entities that are connected to the existing system structure.

4. Evaluation Table: The table below provides detailed evaluations for each merged functional requirement. Evaluations are based on the functional requirement and external interface documents and strictly comply with the user’s rules.

Functional Requirement Evaluation Table

｜Functional Requirement｜Hallucination｜Rationale｜

｜Register Mail (FR-01)｜No｜Involves MailRecord, Group, and Reminder entities, all associated with Contact. The system structure is complete, data flow is closed, and behavioral paths are verifiable.｜

｜Send Mail (FR-02)｜No｜Operation based on MailRecord entity combined with SMTP protocol, part of core system flow with clear behavioral paths.｜

｜Receive Mail (FR-03)｜No｜Involves MailRecord, Group, Contact, and Reminder. Core entities form a complete business loop, consistent with real system behavior.｜

｜Format Mail (FR-04)｜No｜Business depends on UI interaction but lacks a clear definition of formatting methods. Behavioral goals are vague and effects hard to determine.｜

｜Search Mail (FR-05)｜No｜Uses MailRecord fields for condition-based queries. Behavioral goals are specific and logic paths are clear.｜

｜Organize Mail Folders (FR-06)｜No｜Operation involves Mail and Folder entities, supports mail classification functions, logic is complete and structure is clear.｜

｜Manage Group (FR-07)｜No｜CRUD operations on Group entity. Behavioral loop is explicit and consistent with structure and data logic.｜

｜Manage Contact (merged FR-08, FR-09, FR-10)｜No｜Contact operations are related to Group and Reminder entities. Data structure is complete and logical loop supports system implementation.｜

｜Manage Email Account (FR-11)｜No｜Involves management behaviors such as creating, deleting, and configuring EmailAccount. System structure is reasonable.｜

｜Configure Archive Policy (FR-12)｜No｜Behavior centers around mail archiving flow. Triggering mechanism of policies is clearly defined and structure forms a logical loop.｜

｜Execute Mail Archiving (FR-13)｜No｜Mail archiving based on policies. Entity relationships are clear and behavior aligns with actual archiving logic.｜

｜Capture Mail Stream (FR-14)｜No｜Although the MailStream format is undefined, the capture behavior is linked to ArchivePolicy and Mail. Logical path is acceptable.｜

｜Manage Backup (FR-15)｜No｜Involves configuration of backup operations and generation of logs. Entities and behavioral logic are complete with defined operational paths.｜

｜View Backup Log (FR-16)｜No｜Behavior involves querying logs. Path is complete, data structure is clear, and entity relationships are logical.｜

｜Manage Restore Log (FR-17)｜No｜Behavior includes querying and exporting. Structure connects properly with Log entity, and logic paths are clear and feasible.｜

Summary

• The document involves 12 data entities: MailRecord, Folder, Group, Contact, EmailAccount, ArchivePolicy, MailStream, BackupFile, BackupLog, RestoreLog, OperationLog, and Reminder.

• After merging similar functional points, there are 15 functional requirements (originally 17, with FR-08/09/10 merged).

• There are 0 hallucinated requirements: All involved entities are associated with the system core (weakly connected in the E-R diagram).

• All 15 requirements are non-hallucinated.