LLM Automated Evaluation Results

Functional Requirements Evaluation Table

According to the user’s request, I first compiled the data entities involved in the document. Data entities refer to the core data objects that need to be managed within the system, excluding external data sources or formats mentioned in interface descriptions. The data entities extracted from the functional requirement document are as follows:

Mail (including content, attachments, etc., core entity), Contact (contact information in the user’s address book), Group (contact groupings such as mailing lists), Calendar Event (scheduled events in a calendar), Task (to-do items and reminders), Folder (structure for mail categorization), Mail Account (individual mail account info), Shared Account (team shared mailbox), Archiving Policy (rules for mail archiving), Backup Policy (rules for data backup), Log (system activity logs, including mail sending, categorization, etc.).

Total number of data entities: 11.

Next, I performed a merge evaluation on the functional requirements. Based on the rules: only merge identical functional points involving the same data entity (e.g., merging similar operations under different conditions for the same entity); system general settings (e.g., preferences, multilingual support) are excluded, but such requirements were not found in the document and thus none were removed. Excessive merging is avoided (e.g., CRUD operations are not merged into a single “manage” operation unless functionally identical).

Among the 42 functional requirements in the document, I examined the possibility of merging:

For instance, FR-09 (Add Contact) and FR-12 (Manage Contact Info) both involve the Contact entity but represent different functional points (add vs. edit), so they are not merged. Similarly, FR-16 (Create Event) and FR-19 (Manage Event) differ in functional points (create vs. edit) and are not merged. No requirements meet the criteria for merging; therefore, all 42 functional requirements remain distinct.

Final count of functional requirements after merging: 42.

Now, I assess each functional requirement for hallucinated requirements. Evaluation criteria:

Hallucinated requirements introduce operations involving data entities that cannot be associated with any existing entity in the system, resulting in a disconnected E-R diagram (i.e., isolated nodes). If the entity is on the predefined list of 11, it is not hallucinated.

The evaluation result is shown in the following table. Each requirement has been strictly assessed.

Hallucinated requirements: All requirements involve defined data entities; thus, none are hallucinated (the E-R diagram is weakly connected: mail account is the central entity, linked to mail, contacts, etc., with no isolated nodes).

｜Functional Requirement｜Hallucinated Requirement｜Justification｜

｜FR-01 Compose Mail｜No｜Involves the Mail entity, clearly defined functionality, logically structured input/output, describable behavior.｜

｜FR-02 Send Mail｜No｜Associated with Mail entity, workflow clearly defined, operational logic is consistent.｜

｜FR-03 Receive Mail｜No｜Function is reasonable, involves server communication, receiving behavior is system-supported.｜

｜FR-04 Format Mail｜No｜Adds formatting to existing Mail, action is clear, aligns with extension expectations.｜

｜FR-05 Search Mail｜No｜Search logic based on Mail fields, consistent entity reference, conforms to standard search behavior.｜

｜FR-06 Categorize Mail｜No｜Mail sorted into folders based on rules, matches existing entities, aligns with expected operations.｜

｜FR-07 Delete Mail｜No｜Functionally valid, but verification code details are not described in the document.｜

｜FR-08 Archive Mail｜No｜Function based on archiving policy, target definition is clear, operation is valid.｜

｜FR-09 Add Contact｜No｜Action is intuitive, input and saving of contact info match, logically consistent.｜

｜FR-10 Delete Contact｜No｜Deletion logic is reasonable, but verification mechanism lacks documentation support.｜

｜FR-11 Search Contact｜No｜Searching contact info is standard behavior, requirement objective is clear.｜

｜FR-12 Manage Contact Info｜No｜Contact editing process is standard, field operations are valid.｜

｜FR-13 Create Group｜No｜Group creation conforms to common system design, logic is unambiguous.｜

｜FR-14 Manage Group｜No｜Member update logic is valid, behavioral goals are clear.｜

｜FR-15 Delete Group｜No｜Deletion is reasonable, but “retention policy” details are missing in the description.｜

｜FR-16 Create Event｜No｜Event creation is common in office systems, goal is clear and feasible.｜

｜FR-17 Delete Event｜No｜Function is valid, but security verification logic is not detailed in original text.｜

｜FR-18 View Event｜No｜Viewing events is a standard function, process aligns with associated entities.｜

｜FR-19 Manage Event｜No｜Management and notification mechanisms are reasonable, functionality forms a logical cycle.｜

｜FR-20 Create Task｜No｜Task is clearly defined, input/output behavior follows standard logic.｜

｜FR-21 Manage Task｜No｜Task status adjustment and notification follow common system logic, behavior is interpretable.｜

｜FR-22 Delete Task｜No｜Deletion goal is valid, but no definition of verification code logic.｜

｜FR-23 Create Folder｜No｜Folder creation is a basic function in mail management, behavior is clear.｜

｜FR-24 Delete Folder｜No｜Operationally valid, but security validation logic is not explicitly described.｜

｜FR-25 Manage Folder Permissions｜No｜Permission control is a valid extension, input configuration has clear purpose.｜

｜FR-26 Create Mail Account｜No｜Mail account info is clearly defined, creation logic is standard and reusable.｜

｜FR-27 Delete Mail Account｜No｜Deletion request is valid, but lacks detail on approval and retention rules.｜

｜FR-28 Manage Mail Account Permissions｜No｜Permission control logic matches account management needs, goal is clear.｜

｜FR-29 Create Shared Account｜No｜Shared account is an extension type, creation logic continues existing framework.｜

｜FR-30 Delete Shared Account｜No｜Deletion is behaviorally valid, but strategy-related conditions are undefined.｜

｜FR-31 Manage Shared Account Permissions｜No｜Permission assignment is valid, rule-setting logic is clear.｜

｜FR-32 Create Archiving Policy｜No｜Policy definition is behaviorally valid, aligns with archiving process requirements.｜

｜FR-33 Delete Archiving Policy｜No｜Deletion has a basis, but lacks sufficient description of approval parameters.｜

｜FR-34 Modify Archiving Policy｜No｜Modification process is reasonably needed, but approval dependencies are not specified.｜

｜FR-35 Archive Mail｜No｜Operation follows defined policies, behavior aligns with associated entities.｜

｜FR-36 Search Archived Mail｜No｜Search behavior is clear, path corresponds to defined archive logic.｜

｜FR-37 Create Backup Policy｜No｜Backup logic is clear, policy parameters are behaviorally bound.｜

｜FR-38 Execute Backup｜No｜Execution is logical, output matches input configuration.｜

｜FR-39 Restore Mail｜No｜Restore action matches entity state changes, workflow has logical linkage.｜

｜FR-40 Manage Logs｜No｜Management action has operational objective, data handling is clear.｜

｜FR-41 Export Logs｜No｜Export is standard, includes format setup and file generation, logic is coherent.｜

｜FR-42 View Operation Logs｜No｜Viewing/filtering behavior is routine, purpose is clear, action is interpretable.｜

Summary

The entire document involves 11 data entities (Mail, Contact, Group, Calendar Event, Task, Folder, Mail Account, Shared Account, Archiving Policy, Backup Policy, Log).

After merging by identical functional points: none were merged; total functional requirements remain 42.

Hallucinated Requirements: 0 (all requirements involve defined entities; the E-R diagram is weakly connected).

Non-Hallucinated Requirements: 42 (since no hallucinated items, all are non-hallucinated).