LLM Automated Evaluation Results

Functional Requirement Evaluation Table

Based on your requirements, I conducted a detailed review of the entire document. First, I tallied the data entities involved in the document. Data entities refer to core data objects that need to be managed within the system (excluding external interfaces such as data sources or external formats). Based on the functional requirements document, I identified the following data entities:

1. Email: Manages email content, status, and attributes (such as sender, recipient, subject, etc.).

2. Folder: Organizational unit for storing emails (emails can be moved or copied to folders).

3. Contact Group: Manages contact groupings (e.g., create, edit, delete contact groups).

4. Contact: Manages contact information (such as name, email address, etc.).

5. Reminder Task: Manages reminders related to emails or schedules.

6. Schedule: Manages scheduling information (such as time, location, participants, etc.).

7. Policy: Manages system policies (e.g., sending, archiving, backup strategies).

8. Backup File: Manages backup data files (such as create, view, delete).

9. Backup Log: Manages backup operation logs (such as view, export, filter, delete).

10. Email Backup Record: Manages records of email backups (such as view, export).

Total number of data entities involved: 10.

Next, I performed merging and counting of the functional requirements to ensure consistent granularity. The merging rules are as follows:

• Combine requirements targeting the same functionality point for the same data entity (e.g., adding, updating, deleting contacts are merged as “Manage Contact”).

• Exclude generic system setting requirements (such as preferences or multilingual support), but none exist in the document, so no removal is necessary.

• Avoid over-merging; merge only when the requirements are highly similar or target the same operation on the same data entity.

There were originally 21 functional requirements. After merging:

• FR-08 (Add Contact), FR-09 (Update Contact), and FR-10 (Delete Contact) are merged into one requirement: Manage Contact.

• No other requirements were merged, as each targets different operations or entities.

• Total functional requirements after merging: 19.

List of merged functional requirements:

1. Register Email (FR-01)

2. Send Email (FR-02)

3. Receive Email (FR-03)

4. Format Email (FR-04)

5. Search Email (FR-05)

6. Organize Email Folders (FR-06)

7. Manage Contact Groups (FR-07)

8. Manage Contacts (Merged FR-08,09,10)

9. Set Reminder Tasks (FR-11)

10. Manage Schedules (FR-12)

11. Define Policies (FR-13)

12. Archive Email (FR-14)

13. View Archived Emails (FR-15)

14. Delete Archived Emails (FR-16)

15. Manage Backups (FR-17)

16. Restore Database (FR-18)

17. Manage Backup Logs (FR-19)

18. Delete Expired Emails (FR-20)

19. View Email Backup Records (FR-21)

Now, I proceed with the evaluations in the following order:

1. Hallucinated Requirement Judgment: The criterion is “a requirement introduces a functional operation but involves a data entity that cannot be connected to any existing entity in the system, resulting in a weakly connected E-R diagram (i.e., isolated node).” The E-R diagram is constructed based on the relationship between data entities and functional requirements; all entities should be connected through paths (e.g., the email entity connects to folders, reminder tasks, etc.). If the requirement causes entity isolation, it is considered a hallucinated requirement.

Evaluation results are shown in the table below. I conducted a slightly stricter evaluation to avoid overlooking any requirement. Note:

• All judgments are based on the merged requirement list.

• Judgment reasons briefly summarize key points (Hallucination: entity connectivity; Completeness: input/output definition; Testability: feasibility of transformation from input to output).

• Summary totals are given after the table.

Functional Requirement Evaluation Table

｜Functional Requirement｜Hallucinated Requirement｜Evaluation Reason｜

｜————————｜————————｜——————｜

｜Register Email｜No｜Involves the Email entity, connected to the core system, input-output closed loop, no isolated structure.｜

｜Send Email｜No｜Involves the Email entity, sending behavior depends on policy, complete structure and clear connection.｜

｜Receive Email｜No｜Involves the Email entity, receiving path is clear, input and output maintain weak connectivity.｜

｜Format Email｜No｜Involves the Email entity, formatting behavior is related to email structure, no isolated node.｜

｜Search Email｜No｜Involves the Email entity, search conditions derived from core fields, E-R diagram is connected.｜

｜Organize Email Folders｜No｜Involves Email and Folder entities, operational behavior closed loop, entity relationship clear.｜

｜Manage Contact Groups｜No｜Involves Contact Group and Contact entities, operation path clear, structure complete.｜

｜Manage Contacts｜No｜Involves the Contact entity, behavior closed loop, entity can be linked in the system.｜

｜Set Reminder Tasks｜No｜Involves Reminder Task and Email entities, task depends on email, logic is connected.｜

｜Manage Schedules｜No｜Involves Schedule and Reminder Task entities, closed loop path, no structural isolation.｜

｜Define Policies｜No｜Involves Policy entity, referenced in multiple functions, centralized structure, no isolation.｜

｜Archive Email｜No｜Involves Email entity, archiving path is clear, system structure is weakly connected.｜

｜View Archived Emails｜No｜Involves Email entity, clear connection to archived content, E-R diagram complete.｜

｜Delete Archived Emails｜No｜Involves Email entity, archiving deletion behavior does not cause isolation, structure accessible.｜

｜Manage Backups｜No｜Involves Email and Backup entities, backup logic forms a closed loop, effective entity connection.｜

｜Restore Database｜No｜Involves Backup File entity, restore operation connects to core behavior, no isolation.｜

｜Manage Backup Logs｜No｜Involves Backup Log and File entities, log behavior has defined structure.｜

｜Delete Expired Emails｜No｜Involves Email entity, expired deletion operation forms structural loop.｜

｜View Email Backup Records｜No｜Involves Email Backup Record and File entities, behavior path complete and traceable.｜

Summary

• The entire document involves 10 data entities (Email, Folder, Contact Group, Contact, Reminder Task, Schedule, Policy, Backup File, Backup Log, Email Backup Record).

• After merging functional points, there are 19 functional requirements (originally 21, contact management merged into 1 item).

• Number of hallucinated requirements: 0 (All requirements involve data entities connected to the core system architecture, E-R diagram is weakly connected: the Email entity is associated with Folder, Reminder Task, etc., the Policy entity is referenced by Send Email, no isolated nodes).

• Number of non-hallucinated requirements: 19 (no hallucinated requirements).

• Among non-hallucinated requirements, 18 have complete references (only “Receive Email” is incomplete because the “User Receive Policy” input is not defined in the document).

• Among non-hallucinated requirements, 19 are testable (All requirements have controllable inputs and verifiable outputs, even if some rely on user interaction or external events such as email reception).

Evaluation Notes:

• Hallucination judgment is strict: All entities are linked through functional operations (e.g., Reminder Task is associated with Email, Policy is linked to Send Email), no disconnection.