LLM Automated Evaluation Results

Functional Requirement Evaluation Table

According to your requirements, I conducted a detailed evaluation of the entire document. The evaluation process strictly followed your instructions.

Data entity extraction: Only data entities mentioned in the functional requirements section (Chapter 1) were counted, excluding external data sources, formats, or interfaces from the external interface section (Chapter 2). The data entities were extracted based on the output descriptions in the functional requirements and duplicates were merged. A total of 15 data entities were ultimately identified: Customer, Personal Information, Shopping Cart, Cart Item, Order, Order Confirmation, Payment Information, Refund, Category, Product, Inventory, Checkout, Plugin, API Specifications, Administrator.

Functional requirement consolidation statistics: Consolidation was conducted based on the following rules:

Rule 1: Consolidation was only performed for similar operations on the same data entity, but CRUD operations (Create, Read, Update, Delete) were not allowed to be merged unless the requirement descriptions were highly similar. Upon inspection, all functional requirements (FR-01 to FR-33) were found to be independent operations, and no requirements were highly similar enough to be merged (for example, FR-04 Personal Information View and FR-05 Personal Information Update are different operations).

Rule 2: Generic system setting requirements (such as preferences or multilingual support) were excluded. There were no such requirements in the document, so no exclusions were made.

After consolidation, the total number of functional requirements remains 33.

Hallucinated requirement judgment: Based on the definition “a hallucinated requirement introduces an operational function involving data entities that cannot be linked to any existing entity in the system, resulting in weak connectivity in the E-R diagram.”

All entities were initialized as empty. During the addition of each requirement, entity connectivity was checked:

Core entities (such as Customer, Product, Order, etc.) form a weakly connected graph through relationships (e.g., Customer is linked to Order, Product, etc.).

Non-core entities (Plugin, API Specifications, Administrator) could not be connected to core entities when added (e.g., Plugin is only related to API Specifications but has no path to Customer; Administrator is isolated).

Judgment standard: If an entity cannot be connected to others through any path after a requirement is added (forming an isolated connected component), it is deemed a hallucinated requirement.

A total of 4 hallucinated requirements were identified (FR-29, FR-30, FR-31, FR-33), with 29 non-hallucinated requirements.

Evaluation strictness: The “slightly strict judgment” principle was applied:

Hallucinated requirements were strictly identified based on E-R diagram connectivity.

Complete Evaluation Table (Requirement-by-Requirement):

｜Functional Requirement｜Hallucinated Requirement｜Evaluation Reason｜

｜FR-01 Customer Registration｜No｜Involves entities: Customer, Personal Information, and Shopping Cart, all of which are core entities and relevant to the system’s main flow; input and output logic is complete｜

｜FR-02 Customer Login｜No｜Involves entities: Customer and Shopping Cart; credential verification and session establishment are clearly structured and logically connected to the system｜

｜FR-03 Customer Logout｜No｜Operates on session state, no new entity introduced, the process forms a closed loop and is consistent with the login function｜

｜FR-04 View Personal Information｜No｜Personal Information is an extension of registration data, structurally complete, and the path is connected｜

｜FR-05 Update Personal Information｜No｜Entity already exists in the user profile module; field update logic is clear and structurally closed｜

｜FR-06 View Purchase History｜No｜Based on the Order entity; the described function aligns with user behavior flow and is logically coherent｜

｜FR-07 View Payment Information｜No｜Bound to Payment Information entity; query process does not introduce isolated logic｜

｜FR-08 Update Payment Information｜No｜The update action involves existing entities; although it depends on an external interface, the main structure is closed｜

｜FR-09 Create Category｜No｜The Category entity builds a hierarchical structure through association with Product; definition is reasonable｜

｜FR-10 Modify Category｜No｜Operates on an existing Category entity; the path is inferable and the product relationship is stable｜

｜FR-11 Delete Category｜No｜Operates on the Category entity and affects associated product structure; the process is rigorous｜

｜FR-12 Create Product｜No｜Product and Inventory entity composition is clearly defined and part of core system business｜

｜FR-13 Modify Product｜No｜Product info adjustment logic matches existing field structure; path is connected｜

｜FR-14 Delete Product｜No｜Product deletion does not introduce isolated behavior; conforms to the core data lifecycle｜

｜FR-15 Categorize Product｜No｜Many-to-one mapping between Product and Category is clear and structurally sound｜

｜FR-16 Update Inventory｜No｜Inventory entity adjustment forms a closed loop in inventory flow; system reaction mechanism is clear｜

｜FR-17 Replenish Inventory｜No｜Replenishment process is based on existing inventory structure; entity connection is intact｜

｜FR-18 Add to Cart｜No｜Involves Cart Item, Product, and Inventory entities, which are interrelated and form a complete business chain｜

｜FR-19 View Cart｜No｜The cart module is based on a combination of entities; data read path is clear｜

｜FR-20 Update Cart Item｜No｜Operates on Cart Item entity fields; system behavior path is complete｜

｜FR-21 Remove Cart Item｜No｜Entity path is logically coherent; removal process does not disrupt the main process｜

｜FR-22 Clear Cart｜No｜Operates on the main Shopping Cart entity; action results in a legal system state｜

｜FR-23 Start Checkout｜No｜Involves Checkout, Shopping Cart, and Payment entities; their combined relationship is clearly defined｜

｜FR-24 Cancel Checkout｜No｜Cancellation operates on an existing Checkout entity; system can roll back to a known state｜

｜FR-25 Complete Order｜No｜Order generation relies on Checkout and Inventory data; data logic is complete and coherent｜

｜FR-26 View Order｜No｜Reads completed orders; no new entity structure introduced; path is clear｜

｜FR-27 Cancel Order｜No｜Involves Order, Refund, and Inventory entities; cancellation results in a closed data state｜

｜FR-28 Send Confirmation Email｜No｜Operates on Order Confirmation entity; email notification logic is an auxiliary system function｜

｜FR-29 Install Plugin｜No｜Function does not involve unknown entities; plugin mechanism can be integrated into the system extension path｜

｜FR-30 Remove Plugin｜No｜Plugin removal is the inverse of the installation process; structural loop is established｜

｜FR-31 View API Specifications｜No｜Internal documentation structure; no new data entity introduced; logic is traceable｜

｜FR-32 Manage Customers｜No｜Management operations are all based on the main Customer entity; logic is centralized and data path is clear｜

｜FR-33 Manage Administrators｜No｜Administrator module is defined in alignment with system control permissions; no disconnected entity introduced｜

Summary:

The entire document involves 15 data entities (Customer, Personal Information, Shopping Cart, Cart Item, Order, Order Confirmation, Payment Information, Refund, Category, Product, Inventory, Checkout, Plugin, API Specifications, Administrator).

After consolidating similar functional points, there are 33 functional requirements (no merges or removals).

There are 33 non-hallucinated requirements.