项目文档

# Functional Requirement

# 1. Functional Requirements  
  
## 1.1 Product Management Function   
\*\*Function ID\*\*: FR-01   
\*\*Description\*\*: Administrators can add, modify, delete, or view products in the system.   
\*\*Input\*\*: Product details such as name, description, price, stock quantity, category, and image URL.   
\*\*Output\*\*: Updated product list or confirmation message for success or failure.  
  
## 1.2 Add Product Function   
\*\*Function ID\*\*: FR-02   
\*\*Description\*\*: Enables the administrator to add a new product to the system, including validation and saving to the database.   
\*\*Input\*\*: Product details including name, description, price, stock quantity, and category.   
\*\*Output\*\*: New product added to the database, with confirmation message and updated product list.  
  
## 1.3 Modify Product Information Function   
\*\*Function ID\*\*: FR-03   
\*\*Description\*\*: Allows the administrator to modify existing product details in the system.   
\*\*Input\*\*: Updated product information such as name, description, price, stock quantity, and category.   
\*\*Output\*\*: Modified product information saved in the database, with confirmation message and updated product list.  
  
## 1.4 Delete Product Function   
\*\*Function ID\*\*: FR-04   
\*\*Description\*\*: Enables the administrator to delete a product from the system after confirmation.   
\*\*Input\*\*: Product ID or product name selected for deletion.   
\*\*Output\*\*: Product removed from the database, with confirmation message and updated product list.  
  
## 1.5 View Product Details Function   
\*\*Function ID\*\*: FR-05   
\*\*Description\*\*: Displays detailed information of a specific product to the customer or administrator.   
\*\*Input\*\*: Product ID or product name selected for viewing.   
\*\*Output\*\*: Display of product details such as name, description, price, stock quantity, and associated images.  
  
## 1.6 Customer Registration Function   
\*\*Function ID\*\*: FR-06   
\*\*Description\*\*: Enables new customers to register in the system and receive a confirmation email.   
\*\*Input\*\*: Customer information including first name, last name, email, password, and contact number.   
\*\*Output\*\*: Registered customer account in the database, confirmation email sent, and success message displayed.  
  
## 1.7 Customer Login Function   
\*\*Function ID\*\*: FR-07   
\*\*Description\*\*: Allows a registered customer to log in using their verified email and password.   
\*\*Input\*\*: Customer's registered email and password.   
\*\*Output\*\*: Successful login and display of the account dashboard; or error message if login fails.  
  
## 1.8 View Product Catalog Function   
\*\*Function ID\*\*: FR-08   
\*\*Description\*\*: Displays the product catalog to customers and administrators, with options for filtering and searching.   
\*\*Input\*\*: Filter criteria such as category, price range, and keywords.   
\*\*Output\*\*: Displayed product catalog with relevant items based on the input criteria.  
  
## 1.9 Place Order Function   
\*\*Function ID\*\*: FR-09   
\*\*Description\*\*: Enables the customer to place an order after selecting products and completing the payment.   
\*\*Input\*\*: Selected product IDs and quantities, customer's chosen payment method, and validated customer session.   
\*\*Output\*\*: A new order created in the database, inventory updated, and a confirmation email sent to the customer.  
  
## 1.10 View Order History Function   
\*\*Function ID\*\*: FR-10   
\*\*Description\*\*: Displays a list of past orders for the customer or administrator, including detailed order information.   
\*\*Input\*\*: User's login session.   
\*\*Output\*\*: Displayed order history with order IDs, dates, total amounts, and statuses; optional detailed view of selected orders.  
  
## 1.11 Order Confirmation Function   
\*\*Function ID\*\*: FR-11   
\*\*Description\*\*: Sends a confirmation email to the customer after a successful order is placed.   
\*\*Input\*\*: Order details including product list, total amount, and estimated delivery time.   
\*\*Output\*\*: Confirmation email sent to the customer and order status updated to "confirmed".  
  
## 1.12 Payment Processing Function   
\*\*Function ID\*\*: FR-12   
\*\*Description\*\*: Handles the customer's payment transaction using the selected payment method.   
\*\*Input\*\*: Customer's payment details such as card number, expiration date, and CVV.   
\*\*Output\*\*: Updated order status (e.g., "paid") and confirmation message to the customer.  
  
## 1.13 Payment Verification Function   
\*\*Function ID\*\*: FR-13   
\*\*Description\*\*: Verifies the validity and success of a payment transaction through the associated payment plugin.   
\*\*Input\*\*: Transaction details from the payment plugin or gateway.   
\*\*Output\*\*: Updated order and payment statuses (e.g., "verified" or "unverified") and notification to the customer and administrator.  
  
## 1.14 Email Notification Configuration Function   
\*\*Function ID\*\*: FR-14   
\*\*Description\*\*: Allows the administrator to configure email notification settings, such as enabled/disabled status and recipient lists.   
\*\*Input\*\*: Email notification settings including notification types and recipient details.   
\*\*Output\*\*: Updated email notification configuration in the database and confirmation message.  
  
## 1.15 Send Order Confirmation Email Function   
\*\*Function ID\*\*: FR-15   
\*\*Description\*\*: Automatically sends a confirmation email to the customer after an order is successfully placed.   
\*\*Input\*\*: Valid order ID and customer email address.   
\*\*Output\*\*: Confirmation email sent to the customer and email sending status logged in the system.  
  
## 1.16 Plugin Installation Function   
\*\*Function ID\*\*: FR-16   
\*\*Description\*\*: Enables the administrator to install new plugins to extend system functionality.   
\*\*Input\*\*: Plugin file or plugin metadata (name, version, description).   
\*\*Output\*\*: Plugin installed and added to the plugin list in the database.  
  
## 1.17 Plugin Activation Function   
\*\*Function ID\*\*: FR-17   
\*\*Description\*\*: Activates a plugin after it has been installed, ensuring it is functional within the system.   
\*\*Input\*\*: Plugin ID and administrator confirmation for activation.   
\*\*Output\*\*: Plugin status updated to "active" in the database and displayed in the plugin list.  
  
## 1.18 Plugin Deactivation Function   
\*\*Function ID\*\*: FR-18   
\*\*Description\*\*: Deactivates a plugin without uninstalling it, making it no longer functional.   
\*\*Input\*\*: Plugin ID and administrator confirmation for deactivation.   
\*\*Output\*\*: Plugin status updated to "inactive" in the database and displayed in the plugin list.  
  
## 1.19 Plugin Uninstallation Function   
\*\*Function ID\*\*: FR-19   
\*\*Description\*\*: Removes a plugin from the system after deactivation and cleans up related configurations.   
\*\*Input\*\*: Plugin ID and administrator confirmation for uninstallation.   
\*\*Output\*\*: Plugin removed from the system and configuration data deleted from the database.  
  
## 1.20 Administrator Login Function   
\*\*Function ID\*\*: FR-20   
\*\*Description\*\*: Authenticates the administrator by validating their login credentials and role.   
\*\*Input\*\*: Administrator's username (email) and password.   
\*\*Output\*\*: Administrator redirected to the admin dashboard or shown an error message if login fails.  
  
## 1.21 Administrator Dashboard Access Function   
\*\*Function ID\*\*: FR-21   
\*\*Description\*\*: Provides the administrator with access to the dashboard containing system metrics and plugin statuses.   
\*\*Input\*\*: Administrator login session.   
\*\*Output\*\*: Displayed dashboard with system status, plugin statuses, and access to various management functions.  
  
## 1.22 Manage Customers Function   
\*\*Function ID\*\*: FR-22   
\*\*Description\*\*: Enables the administrator to add, modify, or delete customer records in the system.   
\*\*Input\*\*: Customer details such as name, email, contact number, and account status.   
\*\*Output\*\*: Updated customer list in the system with success or failure notification.  
  
## 1.23 Manage Orders Function   
\*\*Function ID\*\*: FR-23   
\*\*Description\*\*: Allows the administrator to add, modify, or delete orders, and update inventory and payment status accordingly.   
\*\*Input\*\*: Order details such as customer, product list, quantities, and payment method.   
\*\*Output\*\*: Updated order list in the system, inventory and payment status adjusted, and success or failure notification.  
  
## 1.24 Manage Payments Function   
\*\*Function ID\*\*: FR-24   
\*\*Description\*\*: Enables the administrator to manage payment records, including adding, modifying, or deleting payments.   
\*\*Input\*\*: Payment details such as order ID, amount, method, and transaction ID.   
\*\*Output\*\*: Updated payment list and corresponding order status, with success or failure notification.  
  
## 1.25 Manage Product Category Function   
\*\*Function ID\*\*: FR-25   
\*\*Description\*\*: Allows the administrator to add, modify, or delete product categories and reassign products if needed.   
\*\*Input\*\*: Category details such as name, description, and parent category.   
\*\*Output\*\*: Updated product category list in the database and success or failure notification.  
  
## 1.26 Manage Email Template Function   
\*\*Function ID\*\*: FR-26   
\*\*Description\*\*: Enables the administrator to create, modify, or delete email templates used for notifications.   
\*\*Input\*\*: Email template details such as name, subject, content, and placeholders for dynamic data.   
\*\*Output\*\*: Updated email template list in the database and success or failure notification.  
  
## 1.27 Manage Payment Method Function   
\*\*Function ID\*\*: FR-27   
\*\*Description\*\*: Allows the administrator to add, modify, or delete supported payment methods and their configurations.   
\*\*Input\*\*: Payment method configuration such as method name, description, and plugin-specific parameters.   
\*\*Output\*\*: Updated payment method list and configuration saved in the database, with success or failure notification.  
  
## 1.28 Manage Administrators Function   
\*\*Function ID\*\*: FR-28   
\*\*Description\*\*: Enables the administrator to add, modify, or delete other administrator accounts and update access control.   
\*\*Input\*\*: Administrator details such as first name, last name, email, password, and role.   
\*\*Output\*\*: Updated administrator list in the database, with success or failure notification.

# External Description

# 2. External Interfaces  
  
This chapter describes the external interfaces of the system, including user interfaces, hardware interfaces, software interfaces, and communication interfaces. These interfaces are critical to the system's functionality and interaction with external components.  
  
## 2.1 User Interface Output  
  
The system interacts with users through well-defined graphical user interfaces (GUIs) that support the following user types: \*\*administrators\*\* and \*\*customers\*\*. Each interface is designed to provide a seamless and intuitive experience for performing specific functions.  
  
- \*\*Product Management Interface\*\*   
 - \*\*Description\*\*: A GUI where administrators can perform actions such as adding, modifying, deleting, or viewing products.   
 - \*\*Input/Output\*\*: Input includes product details such as name, description, price, stock quantity, category, and image URL. Output is an updated product list or a success/failure message.  
  
- \*\*Customer Registration and Login Interface\*\*   
 - \*\*Description\*\*: A GUI that allows new customers to register and existing customers to log in.   
 - \*\*Input/Output\*\*: Input includes personal information like first name, last name, email, password, and contact number. Output includes confirmation of successful registration, login to the account dashboard, or error messages for failed attempts.  
  
- \*\*Product Catalog Interface\*\*   
 - \*\*Description\*\*: A GUI that displays the product catalog to users with options for filtering and searching.   
 - \*\*Input/Output\*\*: Input includes filter criteria like category, price range, and keywords. Output includes the displayed catalog with relevant items based on the filters.  
  
- \*\*Order Placement Interface\*\*   
 - \*\*Description\*\*: A GUI where customers can select products, specify quantities, and proceed with payment.   
 - \*\*Input/Output\*\*: Input includes selected product IDs and quantities, as well as a validated customer session. Output includes creation of a new order, updated inventory, and a confirmation message.  
  
- \*\*Order History Interface\*\*   
 - \*\*Description\*\*: A GUI that allows users to view their past orders.   
 - \*\*Input/Output\*\*: Input includes a valid user session. Output includes a list of orders with order IDs, dates, total amounts, and statuses, and an optional detailed view.  
  
- \*\*Administrator Login and Dashboard Interface\*\*   
 - \*\*Description\*\*: A GUI for administrator authentication and access to the system's administrative dashboard.   
 - \*\*Input/Output\*\*: Input includes the administrator's username and password. Output includes access to the admin dashboard or an error message if the login fails.  
  
- \*\*Customer and Administrator Product View Interface\*\*   
 - \*\*Description\*\*: A GUI that displays detailed product information.   
 - \*\*Input/Output\*\*: Input includes a selected product ID or name. Output includes the product's details like name, description, price, stock quantity, and images.  
  
- \*\*Plugin Management Interface\*\*   
 - \*\*Description\*\*: A GUI where administrators can install, activate, deactivate, or uninstall plugins.   
 - \*\*Input/Output\*\*: Input includes plugin metadata such as name, version, and description. Output includes updates to the plugin list and status changes (active/inactive/uninstalled).  
  
- \*\*Email Notification and Template Management Interface\*\*   
 - \*\*Description\*\*: A GUI for administrators to configure email notification settings and manage email templates.   
 - \*\*Input/Output\*\*: Input includes notification settings and template details (name, subject, content, placeholders). Output includes updated configurations and templates with success/failure messages.  
  
- \*\*User Management Interface (Customer and Administrator)\*\*   
 - \*\*Description\*\*: A GUI for administrators to manage customer and administrator accounts.   
 - \*\*Input/Output\*\*: Input includes user details such as name, email, contact number, password, and role. Output includes updated user lists and access control changes with confirmation.  
  
- \*\*Order and Payment Management Interface\*\*   
 - \*\*Description\*\*: A GUI where administrators can manage orders and payments, including status updates and inventory adjustments.   
 - \*\*Input/Output\*\*: Input includes order and payment details such as customer, product list, quantities, payment method, and transaction ID. Output includes updated order and payment lists with success/failure messages.  
  
- \*\*Product Category Management Interface\*\*   
 - \*\*Description\*\*: A GUI for administrators to manage product categories, including adding, modifying, deleting, and reassigning products.   
 - \*\*Input/Output\*\*: Input includes category details like name, description, and parent category. Output includes an updated category list with confirmation.  
  
## 2.2 Hardware Interface Output  
  
The system does not directly interact with any external hardware devices. Therefore, no hardware interfaces are defined for this system.  
  
## 2.3 Software Interface Output  
  
The system interacts with various external software components, including \*\*databases\*\*, \*\*payment plugins\*\*, and \*\*third-party email services\*\*.  
  
- \*\*Database Interface\*\*   
 - \*\*Description\*\*: The system uses a database to store and retrieve data, including products, customers, orders, payments, email templates, plugin configurations, and administrator accounts.   
 - \*\*Input/Output\*\*: Input includes data for insertion, modification, or deletion (e.g., product details, customer information, order records). Output includes confirmation of data updates or retrieval results.  
  
- \*\*Payment Plugin Interface\*\*   
 - \*\*Description\*\*: The system supports integration with third-party payment plugins (e.g., PayPal, Stripe) to handle and verify payment transactions.   
 - \*\*Input/Output\*\*: Input includes customer payment details (e.g., card number, expiration date, CVV) and transaction data from the payment gateway. Output includes updated order and payment statuses (e.g., "paid", "verified") and confirmation messages.  
  
- \*\*Email Notification Service Interface\*\*   
 - \*\*Description\*\*: The system communicates with an external email service to send confirmation emails to customers and administrators.   
 - \*\*Input/Output\*\*: Input includes the customer's email address and order details (e.g., product list, total amount, delivery time). Output includes the delivery status of the email and a log of the notification in the system.  
  
- \*\*Plugin Management Interface (Software-Level)\*\*   
 - \*\*Description\*\*: The system provides an API or software interface to handle plugin installation, activation, deactivation, and uninstallation.   
 - \*\*Input/Output\*\*: Input includes plugin metadata (e.g., name, version, description) and commands for managing plugin status. Output includes plugin status updates and logs of plugin operations.  
  
## 2.4 Communication Interface Output  
  
The system communicates with external components using network-based protocols to ensure data exchange and notification delivery.  
  
- \*\*Email Communication Interface\*\*   
 - \*\*Description\*\*: The system sends and receives email notifications using an SMTP or similar email communication protocol.   
 - \*\*Input/Output\*\*: Input includes the customer's email address and message content (e.g., confirmation message, order details). Output includes a confirmation that the email was sent successfully or an error message if the communication fails.  
  
- \*\*Web-Based Communication Interface\*\*   
 - \*\*Description\*\*: The system supports web-based interaction through HTTP/HTTPS protocols, allowing users to access the system via web browsers.   
 - \*\*Input/Output\*\*: Input includes user requests and session data transmitted over the web. Output includes HTML or JSON responses that render the GUI or provide data for system operations.  
  
- \*\*Payment Gateway Communication Interface\*\*   
 - \*\*Description\*\*: The system communicates with external payment gateways using secure APIs over HTTPS to process and verify transactions.   
 - \*\*Input/Output\*\*: Input includes customer payment data and transaction identifiers. Output includes transaction status and payment confirmation data returned from the gateway.  
  
- \*\*Plugin Communication Interface\*\*   
 - \*\*Description\*\*: The system communicates with installed plugins through predefined APIs or hooks to extend functionality dynamically.   
 - \*\*Input/Output\*\*: Input includes plugin commands and configuration data. Output includes plugin execution results and status updates.  
  
## 2.5 Summary of External Data Sources  
  
The following external data sources are used across the functional requirements and are covered in the interfaces described above:  
  
- \*\*Database\*\*: Used for storing and retrieving product, customer, order, payment, plugin, and administrator data.   
- \*\*Payment Plugin\*\*: Used for processing and verifying customer payments.   
- \*\*Email Service\*\*: Used for sending confirmation emails and managing notification templates.   
- \*\*Web Browser\*\*: Used for user access and interaction with the system.   
- \*\*Plugin APIs\*\*: Used for managing and extending system functionality.  
  
All of these data sources are clearly defined in the external interfaces section, ensuring consistency and clarity for developers and system integrators.

# Use Case

Use Case Name: Product Management   
Use Case ID: UC-01   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage products.   
Postconditions: The product information is updated or added in the system, and the administrator is notified of the success or failure of the operation.   
  
Main Flow:   
1. The administrator selects the "Product Management" option from the system menu.   
2. The system displays a list of existing products.   
3. The administrator chooses to either add a new product or modify an existing one.   
4. The administrator enters the product details (e.g., name, description, price, stock quantity).   
5. The system validates the input data.   
6. The system saves the product information.   
7. The system confirms the successful update or addition of the product.   
  
Alternative Flow:   
1. If the input data is invalid, the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save the product information, the system displays an error message and logs the issue for review.  
  
Use Case Name: Add Product   
Use Case ID: UC-02   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to add products.   
Postconditions: The new product is successfully added to the system, and the administrator is notified of the success or failure of the operation.   
  
Main Flow:   
1. The administrator selects the "Add Product" option from the product management interface.   
2. The system displays an input form for product details.   
3. The administrator fills in the required product information (e.g., name, description, price, stock quantity, and category).   
4. The system validates the input fields for completeness and correctness.   
5. The system saves the new product information to the database.   
6. The system confirms the successful addition of the product and updates the product list.   
  
Alternative Flow:   
1. If any required field is missing or contains invalid data, the system displays an error message and highlights the incorrect fields.   
2. If the system fails to save the product information (e.g., due to database issues), the system displays an error message and logs the failure for troubleshooting.  
  
Use Case Name: Modify Product Information   
Use Case ID: UC-03   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to modify product information. The product to be modified must already exist in the system.   
Postconditions: The selected product information is updated in the system, and the administrator is notified of the success or failure of the modification.   
  
Main Flow:   
1. The administrator selects the "Modify Product" option from the product management interface.   
2. The system displays a list of products with their current information.   
3. The administrator selects the product to be modified.   
4. The system opens an editable form with the selected product's details (e.g., name, description, price, stock quantity, and category).   
5. The administrator updates the necessary product information.   
6. The system validates the updated fields for completeness and correctness.   
7. The system saves the modified product information to the database.   
8. The system confirms the successful modification of the product and updates the product list accordingly.   
  
Alternative Flow:   
1. If any updated field is missing or contains invalid data, the system displays an error message and highlights the incorrect fields.   
2. If the system fails to save the modified product information (e.g., due to database issues), the system displays an error message and logs the failure for troubleshooting.  
  
Use Case Name: Delete Product   
Use Case ID: UC-04   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to delete products. The product to be deleted must already exist in the system.   
Postconditions: The selected product is removed from the system, and the administrator is notified of the success or failure of the deletion.   
  
Main Flow:   
1. The administrator selects the "Delete Product" option from the product management interface.   
2. The system displays a list of existing products.   
3. The administrator selects the product to be deleted.   
4. The system prompts the administrator to confirm the deletion.   
5. The administrator confirms the deletion request.   
6. The system deletes the selected product from the database.   
7. The system updates the product list and notifies the administrator of the successful deletion.   
  
Alternative Flow:   
1. If the administrator cancels the deletion request, the system returns to the product list without making any changes.   
2. If the system fails to delete the product (e.g., due to database issues or if the product is associated with active orders), the system displays an error message and logs the issue for review.  
  
Use Case Name: View Product Details   
Use Case ID: UC-05   
Actors: Customer, Administrator, Plugin   
Preconditions: The customer or administrator is logged into the system. The product to be viewed must exist in the system.   
Postconditions: The product details are displayed to the user, and no changes are made to the product information in the system.   
  
Main Flow:   
1. The customer or administrator navigates to the product listing or search interface.   
2. The user selects a specific product to view its details.   
3. The system retrieves the product information from the database.   
4. The system displays the product details (e.g., name, description, price, stock quantity, and images).   
5. The user reviews the displayed information.   
  
Alternative Flow:   
1. If the product does not exist or cannot be retrieved, the system displays an error message indicating the issue.   
2. If the user requests to view a product without sufficient permissions (e.g., restricted access), the system denies the request and displays an appropriate message.  
  
Use Case Name: Customer Registration   
Use Case ID: UC-06   
Actors: Customer, Plugin, Administrator   
Preconditions: The customer is not yet registered in the system. The system is accessible to the customer via the registration interface.   
Postconditions: The customer is successfully registered in the system, and an email confirmation is sent to the customer. The administrator is notified if there are any issues during registration.   
  
Main Flow:   
1. The customer navigates to the registration page.   
2. The system displays a registration form with required fields (e.g., name, email, password, contact number).   
3. The customer fills in the registration form with their personal information.   
4. The system validates the input data for completeness and correctness.   
5. The system saves the customer's information to the database.   
6. The system sends a confirmation email to the customer's provided email address.   
7. The customer receives the confirmation email and clicks on the verification link.   
8. The system verifies the customer's email address and updates their status to "verified."   
9. The system displays a success message and redirects the customer to the login page.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete, the system displays an error message and prompts the customer to correct the information.   
2. If the system fails to save the customer's information (e.g., due to database issues), an error message is shown, and the administrator is notified for troubleshooting.   
3. If the confirmation email cannot be sent (e.g., due to email service failure), the system displays an error message and logs the issue for review.   
4. If the customer does not click the verification link within a specified time, the system may require them to request a new confirmation email.  
  
Use Case Name: Customer Login   
Use Case ID: UC-07   
Actors: Customer, Plugin   
Preconditions: The customer is not currently logged into the system. The customer has already registered and received a verified email. The login interface is accessible.   
Postconditions: The customer is successfully logged into the system and can access their account. If the login is unsuccessful, the customer is prompted with an appropriate message.   
  
Main Flow:   
1. The customer navigates to the login page.   
2. The system displays the login form with fields for email and password.   
3. The customer enters their registered email and password.   
4. The system validates the email and password against the database.   
5. If the credentials are correct, the system logs the customer in and displays their account dashboard.   
6. The system records the login activity for audit purposes.   
  
Alternative Flow:   
1. If the email is not found in the database, the system displays an error message indicating that the email is invalid.   
2. If the password is incorrect, the system displays an error message and prompts the customer to re-enter the password.   
3. If the system fails to validate the credentials (e.g., due to database issues), an error message is shown, and the administrator is notified for troubleshooting.   
4. If the customer's account is locked or disabled, the system denies access and displays a message explaining the reason.  
  
Use Case Name: View Product Catalog   
Use Case ID: UC-08   
Actors: Customer, Administrator, Plugin   
Preconditions: The customer or administrator is logged into the system. The product catalog must exist in the system.   
Postconditions: The product catalog is displayed to the user, and no changes are made to the product information in the system.   
  
Main Flow:   
1. The customer or administrator navigates to the product catalog section from the homepage or menu.   
2. The system displays the product catalog, including categories, product thumbnails, names, and prices.   
3. The user can filter or search for specific products using available options (e.g., category, price range, keyword).   
4. The system updates the displayed catalog based on the user’s filtering or search criteria.   
5. The user reviews the catalog and may select individual products for more details.   
  
Alternative Flow:   
1. If no products match the search or filter criteria, the system displays a message stating that no products were found.   
2. If the system cannot load the product catalog (e.g., due to database issues), the system displays an error message and logs the issue for review.  
  
Use Case Name: Place Order   
Use Case ID: UC-09   
Actors: Customer, Plugin, Order, Product, Payment, Email   
Preconditions: The customer is logged into the system. The customer has selected at least one product to purchase. The system has valid inventory information for the selected products.   
Postconditions: An order is created and stored in the system. Inventory is updated to reflect the purchased items. The customer receives an order confirmation via email. The payment is processed if applicable.   
  
Main Flow:   
1. The customer selects the "Place Order" option from their shopping cart or product details page.   
2. The system displays the selected products with their quantities and prices.   
3. The customer reviews the order summary and confirms the items to be purchased.   
4. The system checks the inventory for the selected products.   
5. If inventory is sufficient, the system proceeds to the payment interface.   
6. The customer selects a payment method and completes the payment transaction.   
7. The system verifies the payment and creates a new order in the database.   
8. The system updates the inventory for the purchased products.   
9. The system sends a confirmation email to the customer with the order details.   
10. The customer receives the confirmation email and the system displays a success message.   
  
Alternative Flow:   
1. If the inventory is insufficient for any of the selected products, the system displays an error message and suggests alternatives or prompts the customer to adjust the order.   
2. If the payment fails, the system displays an error message and allows the customer to try another payment method or cancel the order.   
3. If the system fails to create the order (e.g., due to database issues), the system displays an error message and logs the issue for review.   
4. If the confirmation email cannot be sent, the system logs the issue and displays a message to the customer indicating the order was placed but the email is pending.  
  
Use Case Name: View Order History   
Use Case ID: UC-10   
Actors: Customer, Administrator, Plugin   
Preconditions: The customer or administrator is logged into the system. There must be existing orders in the system associated with the customer or administrator.   
Postconditions: The order history is displayed to the user, and no changes are made to the order data in the system.   
  
Main Flow:   
1. The customer or administrator navigates to the "Order History" section from the account dashboard.   
2. The system retrieves the list of orders associated with the user's account.   
3. The system displays the order history, including order ID, date, total amount, and order status.   
4. The user can select an individual order to view detailed information such as product list, quantities, and shipping address.   
5. The system retrieves and displays the detailed information for the selected order.   
6. The user reviews the order details and returns to the order history view.   
  
Alternative Flow:   
1. If no orders are found for the user, the system displays a message indicating that there is no order history available.   
2. If the system fails to retrieve the order data (e.g., due to database issues), an error message is shown, and the administrator is notified for troubleshooting.   
3. If the user attempts to view order details without selecting an order, the system prompts them to choose an order from the list.  
  
Use Case Name: Order Confirmation   
Use Case ID: UC-11   
Actors: Customer, Email, Order, Plugin   
Preconditions: The customer has placed an order successfully. The system has created the order and updated the inventory. The customer's email address is valid and verified.   
Postconditions: The customer receives an email confirmation of the order. The order status is updated in the system to "confirmed."   
  
Main Flow:   
1. After a successful order placement, the system generates an order confirmation message.   
2. The system prepares the confirmation email with the order details, including product list, total amount, and expected delivery time.   
3. The system sends the confirmation email to the customer's registered email address.   
4. The customer receives the email and verifies the order information.   
5. The system updates the order status to "confirmed" in the database.   
  
Alternative Flow:   
1. If the system fails to send the confirmation email (e.g., due to email service issues), the system logs the error and displays a message to the customer indicating that the order was placed but the email is pending.   
2. If the customer's email address is invalid or unverified, the system displays an error message and prevents the confirmation email from being sent.   
3. If the system fails to update the order status, an error message is displayed, and the issue is logged for troubleshooting.  
  
Use Case Name: Payment Processing   
Use Case ID: UC-12   
Actors: Customer, Payment, Plugin, Administrator   
Preconditions: The customer has placed an order and selected a payment method. The system has valid order and product information. The payment gateway is accessible.   
Postconditions: The payment is processed successfully, and the order status is updated. The customer is notified of the payment result, and the administrator is alerted in case of failure.   
  
Main Flow:   
1. The system redirects the customer to the payment processing interface.   
2. The customer enters payment details (e.g., card number, expiration date, CVV).   
3. The system sends the payment request to the selected payment plugin.   
4. The payment plugin processes the transaction with the payment gateway.   
5. The payment plugin returns the transaction result to the system.   
6. If the payment is successful, the system updates the order status to "paid."   
7. The system logs the payment and sends a success message to the customer.   
8. The system triggers the order confirmation process.   
  
Alternative Flow:   
1. If the payment details are invalid, the system displays an error message and prompts the customer to re-enter the information.   
2. If the payment plugin returns a failure (e.g., insufficient funds or declined transaction), the system updates the order status to "unpaid" and displays an error message to the customer.   
3. If the system fails to update the order status after a successful payment (e.g., due to database error), the system logs the issue and notifies the administrator.   
4. If the payment gateway is unavailable, the system displays an error message and suggests trying again later or selecting an alternative payment method.  
  
Use Case Name: Payment Verification   
Use Case ID: UC-13   
Actors: Payment, Plugin, Customer, Administrator   
Preconditions: The customer has initiated a payment for an order. The system has received the payment transaction details. The payment plugin is properly configured and accessible.   
Postconditions: The payment is verified as successful or failed. The order status is updated accordingly. The customer is notified of the verification result. The administrator is alerted in case of payment failure.   
  
Main Flow:   
1. The system receives a payment transaction from the customer or payment gateway.   
2. The system forwards the transaction details to the payment plugin for verification.   
3. The payment plugin processes the transaction and verifies its validity (e.g., checks for fraud, confirms payment status).   
4. The plugin returns the verification result (success or failure) to the system.   
5. If the verification is successful, the system updates the order status to "verified" and logs the payment.   
6. The system notifies the customer of the successful verification.   
7. The system may also trigger additional processes, such as order fulfillment or shipment preparation.   
  
Alternative Flow:   
1. If the verification fails (e.g., invalid transaction, suspected fraud), the system updates the order status to "unverified" and logs the issue.   
2. The system displays an error message to the customer and suggests re-initiating the payment or contacting support.   
3. The administrator is alerted to review the failed verification and take necessary actions.   
4. If the payment plugin is unavailable or returns an error, the system displays a message to the customer and logs the issue for troubleshooting.  
  
Use Case Name: Email Notification Configuration   
Use Case ID: UC-14   
Actors: Administrator, Email, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to configure email notifications. The email service and plugins must be properly integrated with the system.   
Postconditions: The email notification settings are updated in the system, and the administrator is notified of the success or failure of the configuration.   
  
Main Flow:   
1. The administrator selects the "Email Notification Configuration" option from the system settings menu.   
2. The system displays the current email notification settings (e.g., enabled/disabled, notification types, recipient lists).   
3. The administrator modifies the settings as needed (e.g., enables notifications, selects notification types, adds or removes recipients).   
4. The system validates the updated configuration for correctness and compatibility with the email service.   
5. The system saves the updated email notification settings.   
6. The system confirms the successful update and displays the new configuration.   
  
Alternative Flow:   
1. If the updated configuration is invalid (e.g., invalid recipient email format), the system displays an error message and prompts the administrator to correct the settings.   
2. If the system fails to save the configuration (e.g., due to database or plugin integration issues), an error message is shown, and the administrator is notified for review.   
3. If the plugin responsible for email notification is not available, the system displays a warning message and prevents saving the configuration until the plugin is restored.  
  
Use Case Name: Send Order Confirmation Email   
Use Case ID: UC-15   
Actors: System, Email, Order, Customer   
Preconditions: The order has been successfully placed and saved in the system. The customer's email address is valid and verified. The email service is active and configured correctly.   
Postconditions: The order confirmation email is sent to the customer. The system logs the email sending status.   
  
Main Flow:   
1. The system triggers the "Send Order Confirmation Email" process after the order is successfully placed.   
2. The system retrieves the customer's registered email address and the order details.   
3. The system generates the confirmation email content, including the order summary, product list, and estimated delivery time.   
4. The system sends the email to the customer through the configured email service.   
5. The system logs the successful email delivery and updates the order status accordingly.   
  
Alternative Flow:   
1. If the customer's email address is invalid or unverified, the system logs the issue and displays a message to the customer indicating the email could not be sent.   
2. If the email service is unavailable or fails to send the email, the system logs the error and displays a message to the customer that the email is pending.   
3. If the system fails to retrieve the order or customer data, an error message is displayed, and the issue is logged for review.  
  
Use Case Name: Plugin Installation   
Use Case ID: UC-16   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to install plugins. The plugin to be installed is available for integration.   
Postconditions: The plugin is successfully installed and integrated into the system. The administrator is notified of the success or failure of the installation.   
  
Main Flow:   
1. The administrator navigates to the "Plugin Installation" section from the system settings or administration menu.   
2. The system displays a list of available plugins, including their names, descriptions, and versions.   
3. The administrator selects the plugin they wish to install.   
4. The system retrieves the plugin configuration and compatibility information.   
5. The system verifies that the plugin is compatible with the current system version.   
6. The administrator confirms the installation request.   
7. The system installs the plugin and updates the plugin configuration in the database.   
8. The system confirms the successful installation and displays the updated plugin list.   
  
Alternative Flow:   
1. If the selected plugin is not compatible with the system, the system displays an error message and prevents installation.   
2. If the plugin is already installed, the system displays a message indicating the plugin is already active.   
3. If the system fails to install the plugin (e.g., due to file access issues or database errors), the system displays an error message and logs the issue for troubleshooting.   
4. If the administrator cancels the installation request, the system returns to the plugin list without making any changes.  
  
Use Case Name: Plugin Activation   
Use Case ID: UC-17   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to activate plugins. The plugin must be installed and available in the system.   
Postconditions: The selected plugin is activated and functional within the system. The administrator is notified of the success or failure of the activation.   
  
Main Flow:   
1. The administrator navigates to the "Plugin Management" section from the administration menu.   
2. The system displays a list of installed plugins with their current status (e.g., active, inactive).   
3. The administrator selects the plugin they wish to activate.   
4. The system verifies the plugin's compatibility and configuration readiness.   
5. The administrator confirms the activation request.   
6. The system activates the plugin and updates its status in the database.   
7. The system confirms the successful activation and displays the updated plugin list.   
  
Alternative Flow:   
1. If the plugin is not compatible with the current system version, the system displays an error message and prevents activation.   
2. If the plugin is already active, the system displays a message indicating the plugin is already activated.   
3. If the system fails to activate the plugin (e.g., due to configuration errors or file access issues), the system displays an error message and logs the issue for troubleshooting.   
4. If the administrator cancels the activation request, the system returns to the plugin list without making any changes.  
  
Use Case Name: Plugin Deactivation   
Use Case ID: UC-18   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to deactivate plugins. The plugin to be deactivated must already be installed and active.   
Postconditions: The selected plugin is deactivated and no longer functional in the system. The administrator is notified of the success or failure of the deactivation.   
  
Main Flow:   
1. The administrator navigates to the "Plugin Management" section from the administration menu.   
2. The system displays a list of installed plugins with their current status (e.g., active, inactive).   
3. The administrator selects the plugin they wish to deactivate.   
4. The system verifies that the plugin is currently active and ready for deactivation.   
5. The administrator confirms the deactivation request.   
6. The system deactivates the plugin and updates its status in the database.   
7. The system confirms the successful deactivation and displays the updated plugin list.   
  
Alternative Flow:   
1. If the selected plugin is already inactive, the system displays a message indicating that the plugin is not currently active.   
2. If the system fails to deactivate the plugin (e.g., due to dependency issues or configuration errors), the system displays an error message and logs the issue for troubleshooting.   
3. If the administrator cancels the deactivation request, the system returns to the plugin list without making any changes.  
  
Use Case Name: Plugin Uninstallation   
Use Case ID: UC-19   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to uninstall plugins. The plugin to be uninstalled must already be installed in the system.   
Postconditions: The selected plugin is removed from the system, and the administrator is notified of the success or failure of the uninstallation.   
  
Main Flow:   
1. The administrator navigates to the "Plugin Management" section from the administration menu.   
2. The system displays a list of installed plugins with their current status (e.g., active, inactive).   
3. The administrator selects the plugin they wish to uninstall.   
4. The system verifies that the plugin is installed and can be uninstalled.   
5. The administrator confirms the uninstallation request.   
6. The system uninstalls the plugin and removes its configuration from the database.   
7. The system confirms the successful uninstallation and updates the plugin list.   
  
Alternative Flow:   
1. If the selected plugin is currently active and cannot be uninstalled without deactivation, the system prompts the administrator to deactivate the plugin first.   
2. If the plugin is not installed, the system displays a message indicating the plugin is not found.   
3. If the system fails to uninstall the plugin (e.g., due to dependency issues or file access errors), the system displays an error message and logs the issue for review.   
4. If the administrator cancels the uninstallation request, the system returns to the plugin list without making any changes.  
  
Use Case Name: Administrator Login   
Use Case ID: UC-01   
Actors: Administrator, Plugin   
Preconditions: The administrator has access to the system login interface. The administrator has a valid account with the correct credentials. The system and associated plugins are operational.   
Postconditions: The administrator is successfully authenticated and logged into the system. If the login fails, the administrator is notified with an appropriate error message.   
  
Main Flow:   
1. The administrator navigates to the login page of the system.   
2. The system displays the login form, requiring the administrator to enter a username and password.   
3. The administrator enters their valid username and password.   
4. The system validates the entered credentials against the stored user data.   
5. If the credentials are correct, the system authenticates the administrator and redirects them to the admin dashboard.   
6. The system logs the login activity for audit and security purposes.   
  
Alternative Flow:   
1. If the username is not found, the system displays an error message indicating an invalid username.   
2. If the password is incorrect, the system displays an error message and prompts the administrator to re-enter the password.   
3. If the system fails to validate the credentials (e.g., due to database issues), an error message is shown, and the administrator is notified for troubleshooting.   
4. If the administrator's account is locked or disabled, the system denies access and displays a message explaining the reason.  
  
Use Case Name: Administrator Dashboard Access   
Use Case ID: UC-20   
Actors: Administrator, Plugin   
Preconditions: The administrator has a valid account and is logged into the system. The system must be operational with active plugins.   
Postconditions: The administrator is granted access to the dashboard, and system status is displayed. If access is denied, an appropriate message is shown.   
  
Main Flow:   
1. The administrator successfully logs into the system.   
2. The system verifies the administrator's role and permissions.   
3. The system loads the dashboard interface, displaying key metrics and system status.   
4. The system lists available plugins and their statuses.   
5. The administrator can navigate to various management sections from the dashboard.   
  
Alternative Flow:   
1. If the administrator does not have sufficient permissions, the system denies access and displays an error message.   
2. If the system fails to load the dashboard (e.g., due to plugin errors or database issues), an error message is shown, and the administrator is notified for troubleshooting.   
3. If the plugin status cannot be retrieved, the system displays a warning and continues with other dashboard functions.  
  
Use Case Name: Manage Customers   
Use Case ID: UC-21   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage customer data. The system must have existing customer records or allow for new customer creation.   
Postconditions: The customer information is updated, added, or deleted in the system. The administrator is notified of the success or failure of the operation.   
  
Main Flow:   
1. The administrator selects the "Manage Customers" option from the administration menu.   
2. The system displays a list of existing customers, including their names, email addresses, and account status.   
3. The administrator chooses to add a new customer, modify an existing one, or delete a customer.   
4. If adding a new customer, the administrator enters the customer's details (e.g., name, email, contact number).   
5. If modifying a customer, the administrator selects the customer and updates the necessary information.   
6. If deleting a customer, the administrator selects the customer and confirms the deletion.   
7. The system validates the input data for completeness and correctness.   
8. The system saves or updates the customer information in the database.   
9. The system confirms the successful operation and updates the customer list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete, the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save or update the customer information (e.g., due to database issues), the system displays an error message and logs the issue for review.   
3. If the customer to be deleted is associated with active orders, the system displays a warning and prevents deletion until the issue is resolved.   
4. If the administrator cancels any operation, the system returns to the customer list without making any changes.  
  
Use Case Name: Manage Orders   
Use Case ID: UC-22   
Actors: Administrator, Plugin, Order, Product, Payment, Customer   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage orders. The system must have existing orders or allow for new order creation.   
Postconditions: The order is updated, added, or deleted in the system. The administrator is notified of the success or failure of the operation. Inventory and payment statuses are updated accordingly.   
  
Main Flow:   
1. The administrator selects the "Manage Orders" option from the administration menu.   
2. The system displays a list of existing orders, including order ID, date, total amount, status, and customer information.   
3. The administrator chooses to add a new order, modify an existing order, or delete an order.   
4. If adding a new order, the administrator enters the order details (e.g., customer, product list, quantities, and payment method).   
5. If modifying an order, the administrator selects the order and updates the necessary information.   
6. If deleting an order, the administrator selects the order and confirms the deletion.   
7. The system validates the input data for completeness and correctness.   
8. The system updates the order information in the database, adjusts inventory levels, and processes payment if applicable.   
9. The system confirms the successful operation and updates the order list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete, the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to update or save the order information (e.g., due to database issues), the system displays an error message and logs the issue for troubleshooting.   
3. If the administrator attempts to delete an order that is already shipped or partially fulfilled, the system displays a warning and prevents deletion.   
4. If the system fails to update inventory or payment status, an error message is displayed, and the issue is logged for review.   
5. If the administrator cancels any operation, the system returns to the order list without making any changes.  
  
Use Case Name: Manage Payments   
Use Case ID: UC-23   
Actors: Administrator, Plugin, Payment, Customer, Order   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage payments. The system must have existing orders with associated payment records or allow for new payment creation.   
Postconditions: The payment records are updated, added, or deleted in the system. The administrator is notified of the success or failure of the operation. The order and payment statuses are updated accordingly.   
  
Main Flow:   
1. The administrator selects the "Manage Payments" option from the administration menu.   
2. The system displays a list of existing payments, including order ID, payment amount, payment status, and method.   
3. The administrator chooses to add a new payment, modify an existing one, or delete a payment.   
4. If adding a new payment, the administrator selects an order and enters the payment details (e.g., amount, method, transaction ID).   
5. If modifying a payment, the administrator selects the payment and updates the necessary information.   
6. If deleting a payment, the administrator selects the payment and confirms the deletion.   
7. The system validates the input data for correctness and consistency with the associated order.   
8. The system updates or adds the payment record in the database and adjusts the order status accordingly.   
9. The system confirms the successful operation and updates the payment list.   
  
Alternative Flow:   
1. If the input data is invalid or inconsistent (e.g., incorrect order ID or payment amount), the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save or update the payment information (e.g., due to database issues), an error message is shown, and the issue is logged for troubleshooting.   
3. If the administrator attempts to delete a payment for a completed or shipped order, the system displays a warning and prevents deletion.   
4. If the system fails to update the order status after a payment change, an error message is displayed, and the issue is logged for review.   
5. If the administrator cancels any operation, the system returns to the payment list without making any changes.  
  
Use Case Name: Manage Plugins   
Use Case ID: UC-24   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage plugins. The system must support plugin installation, activation, deactivation, and uninstallation.   
Postconditions: The plugin is installed, activated, deactivated, or uninstalled in the system. The administrator is notified of the success or failure of the operation.   
  
Main Flow:   
1. The administrator navigates to the "Manage Plugins" section from the administration menu.   
2. The system displays a list of plugins, including their names, statuses, and descriptions.   
3. The administrator selects an action for the plugin (e.g., install, activate, deactivate, uninstall).   
4. The system validates the selected action and checks for any prerequisites (e.g., compatibility, current status).   
5. The administrator confirms the action.   
6. The system performs the selected action (install, activate, deactivate, or uninstall).   
7. The system updates the plugin status in the database and displays the updated plugin list.   
8. The system confirms the success of the operation to the administrator.   
  
Alternative Flow:   
1. If the selected action is not valid for the current plugin status (e.g., attempting to activate an already active plugin), the system displays an error message and prevents the action.   
2. If the plugin is not compatible with the system version, the system displays a warning and prevents installation or activation.   
3. If the system fails to perform the selected action (e.g., due to file access issues, configuration errors, or database problems), the system displays an error message and logs the issue for troubleshooting.   
4. If the administrator cancels the action, the system returns to the plugin list without making any changes.  
  
Use Case Name: Manage Product Category   
Use Case ID: UC-25   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage product categories. The system must have existing product categories or allow for new category creation.   
  
Postconditions: The product category is updated, added, or deleted in the system. The administrator is notified of the success or failure of the operation. Related product data is adjusted accordingly if needed.   
  
Main Flow:   
1. The administrator selects the "Manage Product Category" option from the administration menu.   
2. The system displays a list of existing product categories, including their names, descriptions, and associated products.   
3. The administrator chooses to add a new category, modify an existing one, or delete a category.   
4. If adding a new category, the administrator enters the category details (e.g., name, description, and parent category if applicable).   
5. If modifying a category, the administrator selects the category and updates the necessary information.   
6. If deleting a category, the administrator selects the category and confirms the deletion.   
7. The system validates the input data for completeness and correctness.   
8. The system saves or updates the category information in the database. If a category is deleted, the system ensures no products are associated with it or reassigns the products to another category.   
9. The system confirms the successful operation and updates the category list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete, the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save or update the category information (e.g., due to database issues), an error message is shown, and the issue is logged for troubleshooting.   
3. If the administrator attempts to delete a category that is associated with active products, the system displays a warning and prompts the administrator to reassign the products before deletion.   
4. If the system fails to reassign products during category deletion, an error message is displayed, and the issue is logged for review.   
5. If the administrator cancels any operation, the system returns to the category list without making any changes.  
  
Use Case Name: Manage Email Template   
Use Case ID: UC-26   
Actors: Administrator, Email, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage email templates. The email service and plugins must be properly integrated with the system.   
  
Postconditions: The email template is added, modified, or deleted in the system. The administrator is notified of the success or failure of the operation. The system's email functionality remains unaffected or updated accordingly.   
  
Main Flow:   
1. The administrator selects the "Manage Email Template" option from the administration menu.   
2. The system displays a list of existing email templates, including their names, purposes, and last modified date.   
3. The administrator chooses to add a new email template, modify an existing one, or delete a template.   
4. If adding a new template, the administrator selects the template type (e.g., order confirmation, payment reminder, registration confirmation) and enters the template content (e.g., subject, body, placeholders for dynamic data).   
5. If modifying a template, the administrator selects the template and edits the necessary fields.   
6. If deleting a template, the administrator selects the template and confirms the deletion.   
7. The system validates the template data for correctness and completeness.   
8. The system saves or updates the email template in the database. If a template is deleted, the system ensures no active orders or processes are dependent on it.   
9. The system confirms the successful operation and updates the email template list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete (e.g., missing subject or placeholders), the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save or update the template (e.g., due to database issues), an error message is shown, and the issue is logged for troubleshooting.   
3. If the administrator attempts to delete an email template that is currently in use (e.g., for order confirmations), the system displays a warning and prevents deletion until the template is no longer required.   
4. If the system fails to verify template dependencies, an error message is displayed, and the administrator is prompted to review the template usage.   
5. If the administrator cancels any operation, the system returns to the email template list without making any changes.  
  
Use Case Name: Manage Payment Method   
Use Case ID: UC-27   
Actors: Administrator, Plugin, Payment   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage payment methods. The system must support the integration of payment plugins and have access to the payment configuration database.   
  
Postconditions: The payment method configuration is updated, added, or deleted in the system. The administrator is notified of the success or failure of the operation. The system's payment functionality remains consistent with the updated configuration.   
  
Main Flow:   
1. The administrator selects the "Manage Payment Method" option from the administration menu.   
2. The system displays a list of currently supported payment methods, including their names, statuses, and associated plugins.   
3. The administrator chooses to add a new payment method, modify an existing one, or delete a payment method.   
4. If adding a new payment method, the administrator selects the appropriate plugin and configures the method (e.g., method name, description, and plugin-specific parameters).   
5. If modifying a payment method, the administrator selects the method and updates the necessary configuration details.   
6. If deleting a payment method, the administrator selects the method and confirms the deletion.   
7. The system validates the configuration data for correctness and compatibility with the selected plugin.   
8. The system saves or updates the payment method configuration in the database. If a payment method is deleted, the system ensures no active orders depend on it or prompts the administrator to handle the transition.   
9. The system confirms the successful operation and updates the payment method list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incompatible with the selected plugin (e.g., missing API keys or incorrect parameters), the system displays an error message and prompts the administrator to correct the configuration.   
2. If the system fails to save or update the payment method configuration (e.g., due to database issues), an error message is shown, and the issue is logged for troubleshooting.   
3. If the administrator attempts to delete a payment method that is currently in use for active or pending orders, the system displays a warning and prevents deletion until the method is no longer required.   
4. If the system fails to verify plugin compatibility or dependencies, an error message is displayed, and the administrator is prompted to review the payment method setup.   
5. If the administrator cancels any operation, the system returns to the payment method list without making any changes.  
  
Use Case Name: Manage Administrators   
Use Case ID: UC-28   
Actors: Administrator, Plugin   
Preconditions: The administrator is logged into the system and has appropriate permissions to manage administrator accounts. The system must have existing administrator records or allow for new administrator creation. The plugin responsible for user authentication and authorization must be operational.   
  
Postconditions: The administrator account is added, modified, or deleted in the system. The administrator is notified of the success or failure of the operation. System access control and permissions are updated accordingly.   
  
Main Flow:   
1. The administrator selects the "Manage Administrators" option from the administration menu.   
2. The system displays a list of existing administrators, including their usernames, roles, and account statuses.   
3. The administrator chooses to add a new administrator, modify an existing one, or delete an administrator.   
4. If adding a new administrator, the administrator enters the new user’s details (e.g., username, email, password, and role).   
5. If modifying an existing administrator, the administrator selects the user and updates the necessary information (e.g., role, password, or status).   
6. If deleting an administrator, the administrator selects the user and confirms the deletion.   
7. The system validates the input data for completeness, correctness, and compliance with access control policies.   
8. The system saves or updates the administrator information in the database. If an administrator is deleted, the system ensures no critical functions depend on the deleted user.   
9. The system confirms the successful operation and updates the administrator list accordingly.   
  
Alternative Flow:   
1. If the input data is invalid or incomplete (e.g., duplicate username, missing email, or invalid role), the system displays an error message and prompts the administrator to correct the information.   
2. If the system fails to save or update the administrator information (e.g., due to database issues), an error message is shown, and the issue is logged for troubleshooting.   
3. If the administrator attempts to delete an account that is currently logged in or is the last active administrator, the system displays a warning and prevents the deletion.   
4. If the system fails to verify role-based access control or plugin dependencies, an error message is displayed, and the administrator is prompted to review the configuration.   
5. If the administrator cancels any operation, the system returns to the administrator list without making any changes.