项目文档

# Functional Requirement

1.1 Product Creation Function   
Function ID: FR-01   
Description: Administrators can create new products in the system by providing necessary details such as name, description, price, and inventory level.   
Input: Product details including Name, Description, Price, Inventory, and Status.   
Output: A new Product entry in the database with the provided details.  
  
1.2 Product Update Function   
Function ID: FR-02   
Description: Administrators can modify the details of an existing product, such as name, description, price, or inventory level.   
Input: ProductID and updated details including Name, Description, Price, Inventory, and Status.   
Output: Updated Product entry in the database with the modified details.  
  
1.3 Product Deletion Function   
Function ID: FR-03   
Description: Administrators can delete a product from the system. Deletion is only allowed if the product is not associated with any active order or payment.   
Input: ProductID.   
Output: The selected Product is removed from the database.  
  
1.4 Product View Function   
Function ID: FR-04   
Description: Users (Administrators and Customers) can view the details of a specific product, such as name, description, price, and inventory level.   
Input: ProductID.   
Output: Displayed Product information in a non-editable format.  
  
1.5 Inventory Management Function   
Function ID: FR-05   
Description: Administrators can adjust the inventory level of a product by adding or subtracting the quantity.   
Input: ProductID and quantity change (positive or negative).   
Output: Updated inventory level of the selected Product in the database.  
  
1.6 Product Search Function   
Function ID: FR-06   
Description: Users (Administrators and Customers) can search for products based on criteria such as name, category, or price range.   
Input: Search criteria including Name, Category, or Price Range.   
Output: A list of Products matching the search criteria.  
  
1.7 Customer Registration Function   
Function ID: FR-07   
Description: Administrators can create new customer accounts by providing personal information such as name, email, password, address, and phone number.   
Input: Customer details including Name, Email, Password, Address, and PhoneNumber.   
Output: A new Customer entry in the database with the provided details.  
  
1.8 Customer Information Update Function   
Function ID: FR-08   
Description: Administrators can update the information of an existing customer, such as name, email, address, or phone number.   
Input: CustomerID and updated details including Name, Email, Address, and PhoneNumber.   
Output: Updated Customer entry in the database with the modified details.  
  
1.9 Customer Account Deletion Function   
Function ID: FR-09   
Description: Administrators can delete a customer account from the system. Deletion is only allowed if the customer is not associated with any active order or payment.   
Input: CustomerID.   
Output: The selected Customer account is removed from the database.  
  
1.10 Customer Profile View Function   
Function ID: FR-10   
Description: Users (Administrators and Customers) can view the profile information of a specific customer, such as name, email, address, and phone number.   
Input: CustomerID.   
Output: Displayed Customer profile in a non-editable format.  
  
1.11 Order Placement Function   
Function ID: FR-11   
Description: Customers can place an order for selected products after confirming the order details and completing the payment process.   
Input: CustomerID, ProductID list, and quantity for each product.   
Output: A new Order entry in the database with associated OrderItems and updated inventory levels.  
  
1.12 Order Cancellation Function   
Function ID: FR-12   
Description: Customers or Administrators can cancel an order if it is in an eligible state (e.g., not yet shipped or processed).   
Input: OrderID.   
Output: The selected Order is updated to "Canceled" in the database, and inventory levels are adjusted accordingly.  
  
1.13 Order Details View Function   
Function ID: FR-13   
Description: Users (Administrators and Customers) can view the details of a specific order, including products, quantities, total price, and order status.   
Input: OrderID.   
Output: Displayed Order details in a non-editable format.  
  
1.14 Order Status Tracking Function   
Function ID: FR-14   
Description: Users (Administrators and Customers) can track the current status of an order, including shipping and payment status.   
Input: OrderID.   
Output: Displayed Order status and related tracking information in a non-editable format.  
  
1.15 Payment Processing Function   
Function ID: FR-15   
Description: The system processes a payment for a selected order through a configured Plugin.   
Input: OrderID and Payment details including PaymentMethod and Amount.   
Output: A new Payment entry in the database and an updated Order status to "Paid".  
  
1.16 Payment Confirmation Function   
Function ID: FR-16   
Description: The system confirms a payment after receiving a confirmation from a third-party payment gateway via the Plugin.   
Input: PaymentID and confirmation data from the Plugin.   
Output: Updated Payment status to "Confirmed" in the database and Order status to "Paid".  
  
1.17 Order Confirmation Email Sending Function   
Function ID: FR-17   
Description: The system sends an order confirmation email to the customer using the configured Email template and Plugin.   
Input: OrderID and Customer email address.   
Output: Email sent to the customer with order details, and the Order status updated to "Confirmation Email Sent".  
  
1.18 Email Template Management Function   
Function ID: FR-18   
Description: Administrators can create, update, or delete email templates used for system notifications (e.g., order confirmation).   
Input: EmailTemplate details including TemplateName, Subject, and Content.   
Output: Updated EmailTemplate entry in the database or deletion of the selected template.  
  
1.19 Administrator Registration Function   
Function ID: FR-19   
Description: Administrators can register a new administrator account with the system by providing personal and role-based information.   
Input: Administrator details including Name, Email, Password, and Role.   
Output: A new Admin entry in the database with the provided details.  
  
1.20 Administrator Information Update Function   
Function ID: FR-20   
Description: Administrators can update their own or other administrator accounts with new information such as name, email, or role.   
Input: AdminID and updated details including Name, Email, and Role.   
Output: Updated Admin entry in the database with the modified details.  
  
1.21 Administrator Account Deletion Function   
Function ID: FR-21   
Description: Administrators can delete an administrator account from the system if it is not associated with any active orders or payments.   
Input: AdminID.   
Output: The selected Admin account is removed from the database.  
  
1.22 Plugin Management Function   
Function ID: FR-22   
Description: Administrators can install, update, or uninstall plugins to extend system functionality.   
Input: Plugin details including Name, Version, Description, and Source (for installation).   
Output: Updated Plugin entry in the database or removal of the selected plugin.  
  
1.23 Plugin Configuration Function   
Function ID: FR-23   
Description: Administrators can configure settings for an installed plugin, such as API keys or endpoints.   
Input: PluginID and configuration details including API keys and endpoints.   
Output: Updated Plugin configuration in the database and the Plugin is reloaded to apply changes.  
  
1.24 Plugin Information View Function   
Function ID: FR-24   
Description: Administrators can view the details of an installed plugin, such as name, version, and status.   
Input: PluginID.   
Output: Displayed Plugin information in a non-editable format.  
  
1.25 Order Item Management Function   
Function ID: FR-25   
Description: Users (Administrators and Customers) can manage order items by adding, updating, or removing them from an order.   
Input: OrderID and OrderItem details including ProductID, Quantity, and Subtotal.   
Output: Updated OrderItem entry in the database and adjusted inventory levels for affected products.  
  
1.26 Email Management Function   
Function ID: FR-26   
Description: Administrators can update or delete email records associated with customer communications or system notifications.   
Input: EmailID and updated details including Email content or recipient information.   
Output: Updated or deleted Email entry in the database.

# External Description

# 2. External Interfaces   
  
## 2.1 User Interface Output   
The system provides a set of user interfaces for administrators and customers to interact with the system functionalities. These interfaces are designed to allow data input, modification, and viewing of system information in a non-editable format when appropriate.   
  
\*\*Role and Interaction Method:\*\*   
- \*\*Administrators:\*\*   
 - \*\*Product Management Interface:\*\* Administrators can input and update product details such as Name, Description, Price, Inventory, and Status. This interface supports creation, modification, and deletion of product records.   
 - \*\*Customer Management Interface:\*\* Administrators can input and update customer details such as Name, Email, Address, and PhoneNumber. This interface supports creation, modification, and deletion of customer records.   
 - \*\*Order Management Interface:\*\* Administrators can view Order details including products, quantities, total price, and order status. This interface supports cancellation and tracking of orders.   
 - \*\*Administrator Management Interface:\*\* Administrators can register, update, or delete other admin accounts by providing Name, Email, Password, and Role.   
 - \*\*Plugin Management Interface:\*\* Administrators can install, update, or uninstall plugins by providing Name, Version, Description, and Source. They can also configure plugins by specifying API keys and endpoints.   
 - \*\*Email Template Management Interface:\*\* Administrators can create, update, or delete email templates by providing TemplateName, Subject, and Content.   
  
- \*\*Customers:\*\*   
 - \*\*Product View Interface:\*\* Customers can view product details such as Name, Description, Price, and Inventory by providing ProductID.   
 - \*\*Order Placement Interface:\*\* Customers can select products and specify quantities to place an order. This interface requires inputs such as CustomerID, ProductID list, and quantity for each product.   
 - \*\*Order Cancellation Interface:\*\* Customers can cancel an order by providing OrderID.   
 - \*\*Order Details Interface:\*\* Customers can view Order details including products, quantities, total price, and order status by providing OrderID.   
 - \*\*Order Status Tracking Interface:\*\* Customers can track the current status of an order, including shipping and payment status, by providing OrderID.   
  
\*\*Output Description:\*\*   
- All interfaces output data in a user-friendly format, typically as web-based forms or data tables.   
- For viewing functionalities (e.g., Product View, Order Details), data is displayed in a non-editable format.   
- For management functionalities (e.g., Product Creation, Order Cancellation), the system updates the database and provides confirmation or status messages to the user.   
  
## 2.2 Hardware Interface Output   
No direct interaction with hardware devices is required for the functionalities described in the system. The system does not rely on specialized hardware for processing, storage, or communication.   
  
\*\*Role and Interaction Method:\*\*   
- The system is expected to run on standard server hardware and client devices (e.g., desktops, mobile phones) and does not require any specific hardware components for its operation.   
- The Plugin Management and Configuration functionalities may indirectly interact with hardware via external systems or APIs, but this is abstracted at the software level and does not require direct hardware interface definitions.   
  
## 2.3 Software Interface Output   
The system interacts with several external software components, including a database, third-party payment gateway, and email service plugin. These interfaces are essential for storing, retrieving, and processing data, as well as extending the system’s functionality.   
  
### 2.3.1 Database Interface   
\*\*Description:\*\*   
The system stores and retrieves data from a database. This includes Product, Customer, Order, OrderItem, Payment, EmailTemplate, and Plugin records.   
  
\*\*Interaction Method:\*\*   
- \*\*Product Creation/Update/Deletion:\*\* The system inserts, updates, or deletes a Product record in the database using the provided inputs (e.g., Name, Description, Price, Inventory, Status, ProductID).   
- \*\*Customer Registration/Update/Deletion:\*\* The system inserts, updates, or deletes a Customer record in the database using the provided inputs (e.g., Name, Email, Password, Address, PhoneNumber, CustomerID).   
- \*\*Order Placement/Item Management/Cancellation:\*\* The system inserts, updates, or deletes Order, OrderItem, and Inventory records in the database using inputs such as CustomerID, ProductID, Quantity, OrderID, and Subtotal.   
- \*\*Payment Processing/Confirmation:\*\* The system inserts a Payment record and updates the Order status to "Paid" when a payment is processed or confirmed.   
- \*\*Email Template Management:\*\* The system inserts, updates, or deletes EmailTemplate records using TemplateName, Subject, and Content.   
- \*\*Plugin Management/Configuration:\*\* The system inserts, updates, or deletes Plugin records using Name, Version, Description, Source, and configuration details such as API keys and endpoints.   
  
\*\*Data Flow:\*\*   
- The system reads and writes records in the database as part of its core functionality.   
- Inventory levels are updated in the database when a product is created, updated, or when an order is placed or canceled.   
- Payment status is updated in the database after successful processing or confirmation.   
- Email templates are stored in the database and used when sending system notifications.   
  
### 2.3.2 Payment Gateway Interface (via Plugin)   
\*\*Description:\*\*   
The system processes payments through a configured plugin that interacts with an external payment gateway. This plugin is responsible for handling payment transactions and confirming their success or failure.   
  
\*\*Interaction Method:\*\*   
- The system sends \*\*Payment details\*\* (OrderID, PaymentMethod, Amount) to the plugin for processing.   
- The plugin communicates with the external payment gateway to execute the transaction.   
- The system receives \*\*confirmation data\*\* (e.g., PaymentID, transaction status) from the plugin after the payment is processed.   
  
\*\*Data Flow:\*\*   
- Payment data is sent to the plugin for initiation.   
- The plugin sends a confirmation back to the system, which is then used to update the Payment and Order records in the database.   
  
### 2.3.3 Email Service Interface (via Plugin)   
\*\*Description:\*\*   
The system sends emails to customers using a configured plugin that connects to an email service. This includes order confirmation emails and system notifications.   
  
\*\*Interaction Method:\*\*   
- The system sends \*\*OrderID and Customer email address\*\* to the plugin to trigger the sending of an email.   
- The plugin uses the configured email template and external email service to send the message.   
- The system receives a confirmation from the plugin that the email was sent successfully.   
  
\*\*Data Flow:\*\*   
- Email content is generated based on the configured template and the associated Order data.   
- The plugin sends the email to the customer and updates the Order status to "Confirmation Email Sent" in the database.   
  
### 2.3.4 Plugin Management Interface   
\*\*Description:\*\*   
The system allows administrators to manage plugins, including installation, configuration, and uninstallation. Plugins extend the system's functionality, such as payment and email processing.   
  
\*\*Interaction Method:\*\*   
- \*\*Installation:\*\* The system accepts Plugin details (Name, Version, Description, Source) and installs the plugin.   
- \*\*Configuration:\*\* The system accepts PluginID and configuration details (e.g., API keys, endpoints) and updates the plugin configuration in the database.   
- \*\*Uninstallation:\*\* The system removes the Plugin record from the database and uninstalls the plugin.   
  
\*\*Data Flow:\*\*   
- Plugin information is stored in the database.   
- Configuration changes are saved and the plugin is reloaded to apply new settings.   
  
## 2.4 Communication Interface Output   
The system communicates with external services and users via various communication channels, primarily through email notifications and plugin-based interactions.   
  
### 2.4.1 Email Notification Interface   
\*\*Description:\*\*   
The system sends email notifications to customers using the configured email plugin. This includes order confirmation emails and other system-generated messages.   
  
\*\*Interaction Method:\*\*   
- The system triggers an email notification by providing \*\*OrderID and Customer email address\*\* to the email plugin.   
- The plugin generates the email using the configured template and sends it to the recipient.   
  
\*\*Data Flow:\*\*   
- The email is sent to the customer's email address.   
- The system updates the Order status to "Confirmation Email Sent" once the email is successfully sent.   
  
### 2.4.2 Web-Based Communication Interface   
\*\*Description:\*\*   
The system is accessed via a web-based interface, allowing users to interact with it through web browsers.   
  
\*\*Interaction Method:\*\*   
- Users (Administrators and Customers) interact with the system using HTTP/HTTPS protocols via web browsers.   
- The system processes incoming requests and returns appropriate web pages or API responses.   
  
\*\*Data Flow:\*\*   
- Requests are received via the web interface and processed by the system.   
- Responses (e.g., web pages, JSON data) are sent back to the user's browser for display.   
  
---  
  
\*\*Cross-Check Summary:\*\*   
All external data sources and interfaces referenced in the functional requirements are clearly defined in this section:   
- \*\*Database:\*\* Covers all entity records (Product, Customer, Order, OrderItem, Payment, EmailTemplate, Plugin).   
- \*\*Payment Gateway Plugin:\*\* Processes and confirms payments using OrderID, PaymentMethod, and Amount.   
- \*\*Email Service Plugin:\*\* Sends order confirmation emails using OrderID and Customer email address.   
- \*\*Web Interface:\*\* Supports user interaction for all functionalities.   
- \*\*Plugin Management and Configuration:\*\* Allows for dynamic extension of system capabilities.   
  
Each external interface is described with its role, inputs, outputs, and interaction methods to ensure clarity and ease of implementation.

# Use Case

Use Case Name: Create Product   
Use Case ID: UC-01   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Product data entity.   
  
Postconditions:   
1. A new Product is successfully added to the system.   
2. The system updates the Product database with the new entry.   
  
Main Flow:   
1. The Administrator selects the "Create Product" option from the system menu.   
2. The system displays a Product creation form with required fields (e.g., product name, description, price, inventory level, category).   
3. The Administrator fills in the required information for the new Product.   
4. The Administrator submits the form.   
5. The system validates the input data.   
6. If validation is successful, the system creates a new Product entry in the database.   
7. The system provides a confirmation message that the Product was created successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect data format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the system detects a duplicate Product (e.g., same product name and category), it alerts the Administrator and cancels the creation process.  
  
Use Case Name: Update Product   
Use Case ID: UC-02   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Product data entity.   
3. A valid Product already exists in the database.   
  
Postconditions:   
1. The selected Product is successfully updated with new information.   
2. The system updates the Product database with the modified entry.   
  
Main Flow:   
1. The Administrator selects the "Update Product" option from the system menu.   
2. The system displays a list of existing Products.   
3. The Administrator selects a Product to update.   
4. The system displays the Product details in an editable form.   
5. The Administrator modifies the necessary information (e.g., product name, description, price, inventory level, category).   
6. The Administrator submits the updated form.   
7. The system validates the updated input data.   
8. If validation is successful, the system updates the selected Product in the database.   
9. The system provides a confirmation message that the Product was updated successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect data format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the system detects a duplicate Product after the update (e.g., same product name and category), it alerts the Administrator and cancels the update process.  
  
Use Case Name: Delete Product   
Use Case ID: UC-03   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Product data entity.   
3. A valid Product already exists in the database.   
  
Postconditions:   
1. The selected Product is successfully removed from the system.   
2. The system updates the Product database by deleting the entry.   
  
Main Flow:   
1. The Administrator selects the "Delete Product" option from the system menu.   
2. The system displays a list of existing Products.   
3. The Administrator selects a Product to delete.   
4. The system prompts the Administrator to confirm the deletion.   
5. The Administrator confirms the deletion.   
6. The system removes the selected Product from the database.   
7. The system provides a confirmation message that the Product was deleted successfully.   
  
Alternative Flow:   
1. If the Administrator cancels the deletion, the system returns to the Product list without making any changes.   
2. If the selected Product is associated with an existing Order or Payment, the system alerts the Administrator and prevents deletion to maintain data integrity.  
  
Use Case Name: View Product   
Use Case ID: UC-04   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Product data entity.   
3. At least one Product exists in the database.   
  
Postconditions:   
1. The requested Product information is displayed to the user.   
2. The system does not modify the Product database during this process.   
  
Main Flow:   
1. The user selects the "View Product" option from the system menu or product list.   
2. The system displays a list of available Products.   
3. The user selects a specific Product to view.   
4. The system retrieves the selected Product's details from the database.   
5. The system displays the Product information (e.g., product name, description, price, inventory level, category) in a non-editable format.   
  
Alternative Flow:   
1. If no Products exist in the database, the system displays a message indicating that there are no products to view.   
2. If the selected Product does not exist or has been deleted, the system displays an error message and returns to the product list.  
  
Use Case Name: Manage Inventory   
Use Case ID: UC-05   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Product data entity.   
3. A valid Product already exists in the database.   
  
Postconditions:   
1. The inventory level of the selected Product is updated in the system.   
2. The system maintains data consistency by updating the Product database.   
  
Main Flow:   
1. The Administrator selects the "Manage Inventory" option from the system menu.   
2. The system displays a list of existing Products with their current inventory levels.   
3. The Administrator selects a Product to adjust its inventory.   
4. The system displays an inventory adjustment form with the current inventory level.   
5. The Administrator enters the quantity to add or subtract from the inventory.   
6. The Administrator submits the form.   
7. The system validates the input quantity (e.g., correct format, non-negative values).   
8. If validation is successful, the system updates the inventory level of the selected Product.   
9. The system provides a confirmation message that the inventory was updated successfully.   
  
Alternative Flow:   
1. If the input quantity is invalid (e.g., missing, incorrect format, or negative value), the system displays an error message and prompts the Administrator to correct the input.   
2. If the selected Product does not exist or has been deleted, the system displays an error message and returns to the inventory management screen.  
  
Use Case Name: Search Product   
Use Case ID: UC-06   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Product data entity.   
3. At least one Product exists in the database.   
  
Postconditions:   
1. The system displays a list of Products matching the search criteria.   
2. The Product database remains unchanged.   
  
Main Flow:   
1. The user selects the "Search Product" option from the system menu or product list.   
2. The system displays a search interface with fields such as product name, category, price range, or keyword.   
3. The user enters search criteria and submits the request.   
4. The system queries the Product database for matching entries.   
5. The system displays the list of Products that match the search criteria.   
  
Alternative Flow:   
1. If no Products match the search criteria, the system displays a message indicating that no results were found.   
2. If the search query is invalid or incomplete, the system displays an error message and prompts the user to refine the search.  
  
Use Case Name: Register Customer   
Use Case ID: UC-07   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Customer data entity.   
  
Postconditions:   
1. A new Customer is successfully added to the system.   
2. The system updates the Customer database with the new entry.   
  
Main Flow:   
1. The Administrator selects the "Register Customer" option from the system menu.   
2. The system displays a customer registration form with required fields (e.g., name, email, password, address, contact information).   
3. The Administrator fills in the required information for the new Customer.   
4. The Administrator submits the form.   
5. The system validates the input data (e.g., valid email format, unique email).   
6. If validation is successful, the system creates a new Customer entry in the database.   
7. The system provides a confirmation message that the Customer was registered successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect email format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the system detects a duplicate Customer (e.g., same email address), it alerts the Administrator and cancels the registration process.  
  
Use Case Name: Update Customer Information   
Use Case ID: UC-08   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Customer data entity.   
3. A valid Customer already exists in the database.   
  
Postconditions:   
1. The selected Customer is successfully updated with new information.   
2. The system updates the Customer database with the modified entry.   
  
Main Flow:   
1. The Administrator selects the "Update Customer Information" option from the system menu.   
2. The system displays a list of existing Customers.   
3. The Administrator selects a Customer to update.   
4. The system displays the Customer details in an editable form.   
5. The Administrator modifies the necessary information (e.g., name, email, address, contact information).   
6. The Administrator submits the updated form.   
7. The system validates the updated input data.   
8. If validation is successful, the system updates the selected Customer in the database.   
9. The system provides a confirmation message that the Customer information was updated successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect email format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the system detects a duplicate Customer (e.g., same email address with another existing customer), it alerts the Administrator and cancels the update process.  
  
Use Case Name: Delete Customer Account   
Use Case ID: UC-09   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Customer data entity.   
3. A valid Customer account already exists in the database.   
  
Postconditions:   
1. The selected Customer account is successfully removed from the system.   
2. The system updates the Customer database by deleting the entry.   
  
Main Flow:   
1. The Administrator selects the "Delete Customer Account" option from the system menu.   
2. The system displays a list of existing Customer accounts.   
3. The Administrator selects a Customer account to delete.   
4. The system prompts the Administrator to confirm the deletion.   
5. The Administrator confirms the deletion.   
6. The system removes the selected Customer account from the database.   
7. The system provides a confirmation message that the Customer account was deleted successfully.   
  
Alternative Flow:   
1. If the Administrator cancels the deletion, the system returns to the Customer list without making any changes.   
2. If the selected Customer is associated with an existing Order or Payment, the system alerts the Administrator and prevents deletion to maintain data integrity.  
  
Use Case Name: View Customer Profile   
Use Case ID: UC-10   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Customer data entity.   
3. A valid Customer account already exists in the database.   
  
Postconditions:   
1. The requested Customer profile information is displayed to the user.   
2. The system does not modify the Customer database during this process.   
  
Main Flow:   
1. The user selects the "View Customer Profile" option from the system menu or user account section.   
2. The system retrieves the Customer profile data from the database based on the logged-in user or selected Customer.   
3. The system displays the Customer profile information (e.g., name, email, address, contact information, account status).   
4. The user reviews the displayed profile information.   
  
Alternative Flow:   
1. If the selected Customer does not exist or has been deleted, the system displays an error message and returns to the previous screen.   
2. If the user does not have permission to view the Customer profile (e.g., a Customer attempting to view another Customer's profile), the system displays an access denied message and terminates the process.  
  
Use Case Name: Place Order   
Use Case ID: UC-11   
Actors: Customer, Plugin   
Preconditions:   
1. The Customer is logged into the system.   
2. The system has access to the Product and Order data entities.   
3. The Customer has selected at least one Product to purchase.   
  
Postconditions:   
1. A new Order is successfully created in the system.   
2. The system updates the Order database with the new entry.   
3. The inventory levels of the ordered Products are adjusted accordingly.   
  
Main Flow:   
1. The Customer selects the "Place Order" option from the shopping cart or checkout page.   
2. The system displays a confirmation screen with the list of selected Products, quantities, and total price.   
3. The Customer confirms the order details and proceeds to payment.   
4. The system initiates the Payment process.   
5. The system creates a new Order entry in the database with the Customer and Product details.   
6. The system reduces the inventory levels of the ordered Products.   
7. The system sends an Email confirmation to the Customer.   
8. The system provides a confirmation message that the Order was placed successfully.   
  
Alternative Flow:   
1. If the Customer cancels the order before submission, the system returns to the shopping cart without creating an Order.   
2. If the system detects insufficient inventory for any of the selected Products, it displays an error message and prevents the Order from being placed.   
3. If the Payment process fails, the system displays an error message and does not create the Order.   
4. If the system cannot send the Email confirmation, it logs an error but still proceeds with the Order placement.  
  
Use Case Name: Cancel Order   
Use Case ID: UC-12   
Actors: Customer, Administrator, Plugin   
Preconditions:   
1. The Customer or Administrator is logged into the system.   
2. The system has access to the Order data entity.   
3. A valid Order exists in the database for the Customer or is accessible to the Administrator.   
  
Postconditions:   
1. The selected Order is successfully canceled in the system.   
2. The system updates the Order database to reflect the canceled status.   
3. The inventory levels of the ordered Products are adjusted accordingly.   
  
Main Flow:   
1. The Customer or Administrator selects the "Cancel Order" option from the order list or account section.   
2. The system displays a list of eligible Orders that can be canceled.   
3. The user selects an Order to cancel.   
4. The system prompts the user to confirm the cancellation.   
5. The user confirms the cancellation.   
6. The system updates the selected Order status to "Canceled".   
7. The system adjusts the inventory levels by returning the quantity of canceled Products.   
8. The system sends an Email notification to the Customer regarding the cancellation.   
9. The system provides a confirmation message that the Order was canceled successfully.   
  
Alternative Flow:   
1. If the user cancels the cancellation process, the system returns to the order list without making any changes.   
2. If the selected Order is already processed or shipped, the system alerts the user and prevents cancellation.   
3. If the system cannot send the Email notification, it logs an error but still proceeds with the cancellation.   
4. If the system fails to update the inventory levels due to an error, it alerts the user and rolls back the Order status.  
  
Use Case Name: View Order Details   
Use Case ID: UC-13   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Order data entity.   
3. A valid Order exists in the database for the logged-in user or is accessible to the Administrator.   
  
Postconditions:   
1. The requested Order details are displayed to the user.   
2. The system does not modify the Order or Product databases during this process.   
  
Main Flow:   
1. The user selects the "View Order Details" option from the order history or account section.   
2. The system displays a list of the user's Orders.   
3. The user selects a specific Order to view.   
4. The system retrieves the selected Order's details from the database, including Products, quantities, total price, and order status.   
5. The system displays the Order information in a non-editable format.   
  
Alternative Flow:   
1. If the selected Order does not exist or has been deleted, the system displays an error message and returns to the order list.   
2. If the user does not have permission to view the Order (e.g., a Customer attempting to view another Customer’s Order), the system displays an access denied message and terminates the process.  
  
Use Case Name: Track Order Status   
Use Case ID: UC-14   
Actors: Customer, Administrator, Plugin   
Preconditions:   
1. The Customer or Administrator is logged into the system.   
2. The system has access to the Order data entity.   
3. A valid Order exists in the database for the logged-in Customer or is accessible to the Administrator.   
  
Postconditions:   
1. The current status and details of the selected Order are displayed to the user.   
2. The system does not modify the Order or Product databases during this process.   
  
Main Flow:   
1. The Customer or Administrator selects the "Track Order Status" option from the account section or order list.   
2. The system displays a list of the user's Orders or all Orders accessible to the Administrator.   
3. The user selects a specific Order to track.   
4. The system retrieves the selected Order's status and related details (e.g., shipping information, payment status, estimated delivery date) from the database.   
5. The system displays the Order status and information in a non-editable format.   
  
Alternative Flow:   
1. If the selected Order does not exist or has been deleted, the system displays an error message and returns to the order list.   
2. If the user does not have permission to track the Order (e.g., a Customer attempting to track another Customer’s Order), the system displays an access denied message and terminates the process.  
  
Use Case Name: Process Payment   
Use Case ID: UC-15   
Actors: Customer, Administrator, Plugin   
Preconditions:   
1. The Customer is logged into the system.   
2. The system has access to the Order and Payment data entities.   
3. A valid Order exists in the database and is ready for payment.   
  
Postconditions:   
1. The Payment is successfully processed and recorded in the system.   
2. The selected Order status is updated to reflect the completed payment.   
3. The system sends an Email confirmation to the Customer.   
  
Main Flow:   
1. The Customer selects the "Process Payment" option from the checkout page after confirming the Order.   
2. The system displays the payment interface with available payment methods.   
3. The Customer selects a payment method and enters the required payment details.   
4. The system validates the payment information (e.g., card details, billing address).   
5. The system initiates the payment transaction through the Plugin.   
6. The Plugin processes the payment and returns a success confirmation.   
7. The system records the Payment in the database and updates the Order status to "Paid".   
8. The system sends an Email confirmation to the Customer.   
9. The system provides a confirmation message that the Payment was completed successfully.   
  
Alternative Flow:   
1. If the payment information is invalid (e.g., incorrect card number, expired card), the system displays an error message and prompts the Customer to correct the details.   
2. If the Plugin returns a failed transaction, the system alerts the Customer and keeps the Order in a "Pending" status.   
3. If the system cannot send the Email confirmation, it logs an error but still proceeds with the Payment and Order status update.   
4. If the Order has already been paid, the system displays an error message and prevents duplicate payment processing.  
  
Use Case Name: Confirm Payment   
Use Case ID: UC-16   
Actors: Customer, Administrator, Plugin   
Preconditions:   
1. The Customer is logged into the system.   
2. The system has access to the Payment and Order data entities.   
3. A Payment has been initiated and is in a "Pending" status.   
  
Postconditions:   
1. The Payment status is updated to "Confirmed" in the system.   
2. The associated Order is marked as "Paid".   
3. The system sends an Email confirmation to the Customer.   
  
Main Flow:   
1. The system receives a payment confirmation request (e.g., from a third-party payment gateway via the Plugin).   
2. The system verifies the Order associated with the Payment.   
3. The system checks that the Payment is in a "Pending" status.   
4. The system updates the Payment status to "Confirmed" in the database.   
5. The system updates the associated Order status to "Paid".   
6. The system sends an Email to the Customer confirming the payment and order status.   
7. The system provides a confirmation message to the user that the payment has been confirmed.   
  
Alternative Flow:   
1. If the Payment is already confirmed, the system displays an error message and does not update the status.   
2. If the associated Order is invalid or does not exist, the system logs an error and does not confirm the Payment.   
3. If the system cannot send the Email notification, it logs an error but still proceeds with the Payment and Order status update.   
4. If the Plugin returns an invalid confirmation, the system alerts the Administrator and keeps the Payment in "Pending" status.  
  
Use Case Name: Send Order Confirmation Email   
Use Case ID: UC-17   
Actors: System, Email, Customer, Plugin   
Preconditions:   
1. The system has access to the Order and Email data entities.   
2. A valid Order has been placed and processed.   
3. The associated Customer has provided a valid email address.   
4. The system is integrated with an Email service via the Plugin.   
  
Postconditions:   
1. The Customer receives an email confirming the Order.   
2. The system logs the successful email delivery.   
3. The Order status is updated to reflect that the confirmation has been sent.   
  
Main Flow:   
1. The system triggers the "Send Order Confirmation Email" process after a successful Order placement or Payment confirmation.   
2. The system retrieves the Order details and the Customer's email address from the database.   
3. The system generates an email template containing Order information (e.g., order number, product list, total amount, and delivery status).   
4. The system uses the Plugin to send the email to the Customer.   
5. The system updates the Order status to "Confirmation Email Sent".   
6. The system logs the email delivery for tracking and auditing purposes.   
  
Alternative Flow:   
1. If the Customer's email address is invalid or missing, the system logs an error and does not send the confirmation email.   
2. If the Plugin fails to send the email, the system displays an error message and keeps the Order status as "Pending Confirmation Email".   
3. If the system is unable to retrieve the Order or Customer information, it logs an error and terminates the process.  
  
Use Case Name: Manage Email Templates   
Use Case ID: UC-18   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Email data entity.   
3. The system is integrated with an email service via the Plugin.   
  
Postconditions:   
1. The email template is successfully created, updated, or deleted.   
2. The system updates the Email database with the new or modified template.   
  
Main Flow:   
1. The Administrator selects the "Manage Email Templates" option from the system menu.   
2. The system displays a list of existing email templates (e.g., order confirmation, payment confirmation, etc.).   
3. The Administrator selects an option to either create, update, or delete a template.   
4. If creating, the system displays a form to enter the template name, subject, and body.   
5. If updating, the system displays the selected template in an editable form.   
6. If deleting, the system prompts the Administrator to confirm the deletion.   
7. The Administrator submits the form with the updated or new template information.   
8. The system validates the template data (e.g., required fields, valid syntax).   
9. The system updates or creates the email template in the database.   
10. The system provides a confirmation message that the template was successfully managed.   
  
Alternative Flow:   
1. If the template data is invalid (e.g., missing subject, incorrect syntax), the system displays an error message and prompts the Administrator to correct the information.   
2. If the Administrator cancels the deletion process, the system returns to the template list without making changes.   
3. If the selected template does not exist or has already been deleted, the system displays an error message and returns to the template list.   
4. If the system fails to update the database, it logs an error and informs the Administrator that the template could not be saved.  
  
Use Case Name: Register Administrator   
Use Case ID: UC-19   
Actors: Administrator, Plugin   
Preconditions:   
1. The system has access to the Administrator data entity.   
2. No Administrator is currently logged in.   
3. The system is in an administrative setup or configuration mode.   
  
Postconditions:   
1. A new Administrator account is successfully created in the system.   
2. The system updates the Administrator database with the new entry.   
  
Main Flow:   
1. The system prompts for the "Register Administrator" process.   
2. The system displays a registration form with required fields (e.g., full name, email, password, role, permissions).   
3. The user fills in the required information for the new Administrator.   
4. The user submits the form.   
5. The system validates the input data (e.g., valid email format, strong password, unique email).   
6. If validation is successful, the system creates a new Administrator entry in the database.   
7. The system provides a confirmation message that the Administrator was registered successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect email format), the system displays an error message and prompts the user to correct the information.   
2. If the system detects a duplicate Administrator (e.g., same email address already registered), it alerts the user and cancels the registration process.   
3. If the system fails to save the Administrator information due to a database error, it logs the error and displays a message to the user.  
  
Use Case Name: Update Administrator Information   
Use Case ID: UC-20   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Administrator data entity.   
3. A valid Administrator account already exists in the database.   
  
Postconditions:   
1. The selected Administrator is successfully updated with new information.   
2. The system updates the Administrator database with the modified entry.   
  
Main Flow:   
1. The Administrator selects the "Update Administrator Information" option from the account settings or system menu.   
2. The system displays a list of existing Administrator accounts.   
3. The Administrator selects an account to update.   
4. The system displays the Administrator details in an editable form.   
5. The Administrator modifies the necessary information (e.g., name, email, password, role, permissions).   
6. The Administrator submits the updated form.   
7. The system validates the updated input data (e.g., correct email format, unique email).   
8. If validation is successful, the system updates the selected Administrator in the database.   
9. The system provides a confirmation message that the Administrator information was updated successfully.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., missing required fields, incorrect email format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the system detects a duplicate Administrator (e.g., same email address already registered), it alerts the Administrator and cancels the update process.   
3. If the selected Administrator does not exist or has been deleted, the system displays an error message and returns to the administrator list.   
4. If the system fails to update the database due to an error, it logs the error and displays a message to the Administrator.  
  
Use Case Name: Delete Administrator Account   
Use Case ID: UC-21   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Administrator data entity.   
3. A valid Administrator account already exists in the database.   
  
Postconditions:   
1. The selected Administrator account is successfully removed from the system.   
2. The system updates the Administrator database by deleting the entry.   
  
Main Flow:   
1. The Administrator selects the "Delete Administrator Account" option from the account management section.   
2. The system displays a list of existing Administrator accounts.   
3. The Administrator selects an account to delete.   
4. The system prompts the Administrator to confirm the deletion.   
5. The Administrator confirms the deletion.   
6. The system removes the selected Administrator account from the database.   
7. The system provides a confirmation message that the account was deleted successfully.   
  
Alternative Flow:   
1. If the Administrator cancels the deletion, the system returns to the Administrator list without making any changes.   
2. If the selected Administrator is associated with an existing Order or Payment, the system alerts the Administrator and prevents deletion to maintain data integrity.   
3. If the selected Administrator does not exist or has been deleted, the system displays an error message and returns to the Administrator list.  
  
Use Case Name: Manage Plugins   
Use Case ID: UC-22   
Actors: Administrator   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Plugin data entity.   
3. The system is in a configuration or management mode.   
  
Postconditions:   
1. The Plugin is successfully added, updated, or removed from the system.   
2. The Plugin database is updated to reflect the changes.   
  
Main Flow:   
1. The Administrator selects the "Manage Plugins" option from the system settings or administration menu.   
2. The system displays a list of currently installed or configured Plugins.   
3. The Administrator selects an action: Install, Update, or Delete a Plugin.   
4. If installing, the system displays a Plugin installation form with required details (e.g., name, version, source URL, description).   
5. If updating, the system displays the selected Plugin in an editable form.   
6. If deleting, the system prompts the Administrator to confirm the deletion.   
7. The Administrator submits the form with the updated or new Plugin information.   
8. The system validates the Plugin data (e.g., correct URL format, unique name).   
9. The system updates or creates the Plugin entry in the database.   
10. The system provides a confirmation message that the Plugin was successfully managed.   
  
Alternative Flow:   
1. If the input Plugin data is invalid (e.g., missing required fields, incorrect format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the Administrator cancels the deletion process, the system returns to the Plugin list without making changes.   
3. If the selected Plugin does not exist or has already been deleted, the system displays an error message and returns to the Plugin list.   
4. If the system fails to update the Plugin database due to an error, it logs the error and displays a message to the Administrator.  
  
Use Case Name: Install Plugin   
Use Case ID: UC-23   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Plugin data entity.   
3. The system is in a configuration or administration mode.   
  
Postconditions:   
1. The selected Plugin is successfully installed and configured in the system.   
2. The Plugin database is updated with the new installation record.   
3. The system confirms the Plugin installation status.   
  
Main Flow:   
1. The Administrator selects the "Install Plugin" option from the system settings or administration menu.   
2. The system displays a list of available Plugins or a Plugin search interface.   
3. The Administrator selects the Plugin to install.   
4. The system retrieves the Plugin details (e.g., name, version, source, description).   
5. The system prompts the Administrator to confirm the installation.   
6. The Administrator confirms the installation.   
7. The system installs the Plugin and updates the Plugin database.   
8. The system provides a confirmation message that the Plugin was installed successfully.   
  
Alternative Flow:   
1. If the Plugin is already installed, the system displays an error message and does not proceed with installation.   
2. If the Plugin details are missing or incomplete, the system displays an error message and prompts the Administrator to select a valid Plugin.   
3. If the installation fails due to compatibility or system errors, the system logs the error and displays a message to the Administrator.   
4. If the Administrator cancels the installation, the system returns to the Plugin list without making any changes.  
  
Use Case Name: Uninstall Plugin   
Use Case ID: UC-24   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Plugin data entity.   
3. A valid Plugin is currently installed in the system.   
  
Postconditions:   
1. The selected Plugin is successfully uninstalled from the system.   
2. The Plugin database is updated by removing the Plugin entry.   
3. The system confirms the Plugin uninstallation status.   
  
Main Flow:   
1. The Administrator selects the "Uninstall Plugin" option from the system settings or administration menu.   
2. The system displays a list of currently installed Plugins.   
3. The Administrator selects the Plugin to uninstall.   
4. The system prompts the Administrator to confirm the uninstallation.   
5. The Administrator confirms the uninstallation.   
6. The system uninstalls the selected Plugin and removes its configuration.   
7. The system updates the Plugin database by deleting the Plugin entry.   
8. The system provides a confirmation message that the Plugin was uninstalled successfully.   
  
Alternative Flow:   
1. If the Plugin is in use or critical to the system, the system alerts the Administrator and prevents uninstallation.   
2. If the Administrator cancels the uninstallation, the system returns to the Plugin list without making any changes.   
3. If the selected Plugin does not exist or has already been uninstalled, the system displays an error message and returns to the Plugin list.   
4. If the system fails to update the Plugin database due to an error, it logs the error and displays a message to the Administrator.  
  
Use Case Name: Configure Plugin Settings   
Use Case ID: UC-25   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Plugin data entity.   
3. A valid Plugin is already installed in the system.   
  
Postconditions:   
1. The Plugin settings are successfully updated in the system.   
2. The Plugin database is updated with the new configuration.   
3. The Plugin is reloaded or restarted to apply the new settings.   
  
Main Flow:   
1. The Administrator selects the "Configure Plugin Settings" option from the system settings or administration menu.   
2. The system displays a list of installed Plugins.   
3. The Administrator selects a Plugin to configure.   
4. The system displays a configuration form with the Plugin's current settings.   
5. The Administrator modifies the Plugin settings as needed (e.g., API keys, endpoints, activation status).   
6. The Administrator submits the updated configuration.   
7. The system validates the configuration data (e.g., correct format, required fields).   
8. If validation is successful, the system updates the Plugin configuration in the database.   
9. The system reloads or restarts the Plugin to apply the new settings.   
10. The system provides a confirmation message that the Plugin settings were configured successfully.   
  
Alternative Flow:   
1. If the configuration data is invalid (e.g., missing required fields, incorrect format), the system displays an error message and prompts the Administrator to correct the information.   
2. If the selected Plugin does not exist or has already been uninstalled, the system displays an error message and returns to the Plugin list.   
3. If the system fails to update the Plugin database due to an error, it logs the error and displays a message to the Administrator.   
4. If the Plugin cannot be reloaded or restarted due to an error, the system displays a warning and suggests troubleshooting steps or manual intervention.  
  
Use Case Name: View Plugin Information   
Use Case ID: UC-26   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Plugin data entity.   
3. At least one Plugin is installed and available in the system.   
  
Postconditions:   
1. The Plugin information is displayed to the Administrator.   
2. The Plugin database remains unchanged.   
  
Main Flow:   
1. The Administrator selects the "View Plugin Information" option from the administration menu.   
2. The system displays a list of installed Plugins.   
3. The Administrator selects a specific Plugin to view.   
4. The system retrieves the selected Plugin's details from the database (e.g., name, version, description, status).   
5. The system displays the Plugin information in a non-editable format.   
  
Alternative Flow:   
1. If no Plugins are installed, the system displays a message indicating that there are no plugins to view.   
2. If the selected Plugin does not exist or has been uninstalled, the system displays an error message and returns to the Plugin list.   
3. If the system fails to retrieve the Plugin information due to an error, it logs the error and displays a message to the Administrator.  
  
Use Case Name: Manage OrderItem   
Use Case ID: UC-27   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Order and Product data entities.   
3. A valid Order exists in the database, containing at least one OrderItem.   
  
Postconditions:   
1. The selected OrderItem is successfully added, updated, or removed from the system.   
2. The system updates the Order database to reflect changes in the OrderItem.   
3. The inventory levels of the associated Products are adjusted accordingly when an OrderItem is removed.   
  
Main Flow:   
1. The user selects the "Manage OrderItem" option from the order details or account section.   
2. The system displays a list of OrderItems within the selected Order.   
3. The user selects an action: Add, Update, or Remove an OrderItem.   
4. If adding, the system displays a form to select a Product and specify the quantity.   
5. If updating, the system displays the selected OrderItem in an editable form (e.g., quantity, product details).   
6. If removing, the system prompts the user to confirm the removal.   
7. The user submits the form with the updated or new OrderItem information.   
8. The system validates the input data (e.g., product availability, valid quantity).   
9. If validation is successful, the system updates the OrderItem in the database.   
10. If an OrderItem is removed, the system adjusts the inventory levels of the associated Product.   
11. The system provides a confirmation message that the OrderItem was successfully managed.   
  
Alternative Flow:   
1. If the input data is invalid (e.g., product not available, quantity exceeds inventory), the system displays an error message and prompts the user to correct the information.   
2. If the user cancels the removal process, the system returns to the OrderItem list without making any changes.   
3. If the selected OrderItem does not exist or has already been removed, the system displays an error message and returns to the OrderItem list.   
4. If the system fails to update the Order or Product database due to an error, it logs the error and displays a message to the user.   
5. If the user is a Customer and attempts to modify an OrderItem in an Order that has already been processed, the system alerts the user and prevents the change.   
6. If the system fails to adjust the inventory levels due to an error, it alerts the user and rolls back the OrderItem removal.  
  
Use Case Name: Manage EmailTemplate   
Use Case ID: UC-28   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the EmailTemplate data entity.   
3. The system is integrated with an email service via the Plugin.   
  
Postconditions:   
1. The EmailTemplate is successfully created, updated, or deleted in the system.   
2. The system updates the EmailTemplate database with the new or modified entry.   
  
Main Flow:   
1. The Administrator selects the "Manage EmailTemplate" option from the administration menu.   
2. The system displays a list of existing EmailTemplates (e.g., order confirmation, payment confirmation, shipping update, etc.).   
3. The Administrator selects an action: Create, Update, or Delete an EmailTemplate.   
4. If creating, the system displays a form for entering the template name, subject, body content, and associated event.   
5. If updating, the system displays the selected EmailTemplate in an editable form.   
6. If deleting, the system prompts the Administrator to confirm the deletion.   
7. The Administrator submits the form with the new or updated EmailTemplate information.   
8. The system validates the EmailTemplate data (e.g., required fields, valid syntax, template uniqueness).   
9. If validation is successful, the system updates or creates the EmailTemplate in the database.   
10. The system provides a confirmation message that the EmailTemplate was successfully managed.   
  
Alternative Flow:   
1. If the EmailTemplate data is invalid (e.g., missing subject, incorrect syntax), the system displays an error message and prompts the Administrator to correct the information.   
2. If the Administrator cancels the deletion process, the system returns to the EmailTemplate list without making changes.   
3. If the selected EmailTemplate does not exist or has already been deleted, the system displays an error message and returns to the EmailTemplate list.   
4. If the system fails to update the EmailTemplate database due to an error, it logs the error and displays a message to the Administrator.   
5. If the EmailTemplate is associated with an automated system process (e.g., order confirmation), the system alerts the Administrator before deletion to confirm the impact.  
  
Use Case Name: Delete Order   
Use Case ID: UC-29   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Order data entity.   
3. A valid Order exists in the database that is accessible to the user.   
4. The Order is in a cancellable or deletable state (e.g., not shipped or processed).   
  
Postconditions:   
1. The selected Order is successfully removed from the system.   
2. The system updates the Order database by deleting the entry.   
3. The inventory levels of the associated Products are adjusted accordingly.   
  
Main Flow:   
1. The user selects the "Delete Order" option from the order history or account section.   
2. The system displays a list of Orders eligible for deletion.   
3. The user selects an Order to delete.   
4. The system prompts the user to confirm the deletion.   
5. The user confirms the deletion.   
6. The system removes the selected Order from the database.   
7. The system adjusts the inventory levels of the Products included in the deleted Order.   
8. The system provides a confirmation message that the Order was deleted successfully.   
  
Alternative Flow:   
1. If the user cancels the deletion, the system returns to the Order list without making any changes.   
2. If the selected Order is already shipped or processed, the system alerts the user and prevents deletion.   
3. If the system fails to adjust the inventory levels due to an error, it alerts the user and rolls back the Order deletion.   
4. If the system fails to update the Order database due to an error, it logs the error and displays a message to the user.   
5. If the user is a Customer and attempts to delete an Order they do not own, the system displays an access denied message and terminates the process.  
  
Use Case Name: Delete Payment   
Use Case ID: UC-30   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Payment data entity.   
3. A valid Payment record exists in the database for the selected Order.   
4. The associated Order is in a state that allows payment deletion (e.g., not yet shipped or processed).   
  
Postconditions:   
1. The selected Payment record is successfully removed from the system.   
2. The system updates the Payment database by deleting the entry.   
3. The associated Order status is adjusted to reflect the removal of the Payment.   
4. The system ensures that no related financial transactions are affected unless appropriate rollbacks are supported.   
  
Main Flow:   
1. The Administrator selects the "Delete Payment" option from the order management or payment administration section.   
2. The system displays a list of existing Payments, including their associated Orders and statuses.   
3. The Administrator selects a Payment to delete.   
4. The system verifies that the associated Order is eligible for payment deletion.   
5. The system prompts the Administrator to confirm the deletion.   
6. The Administrator confirms the deletion.   
7. The system removes the selected Payment from the database.   
8. The system updates the associated Order status to reflect that the Payment has been removed (e.g., "Unpaid").   
9. The system provides a confirmation message that the Payment was deleted successfully.   
  
Alternative Flow:   
1. If the Administrator cancels the deletion, the system returns to the Payment list without making any changes.   
2. If the selected Payment does not exist or has already been deleted, the system displays an error message and returns to the Payment list.   
3. If the associated Order has already been processed or shipped, the system alerts the Administrator and prevents deletion to maintain data integrity.   
4. If the system fails to update the Order status due to an error, it alerts the Administrator and rolls back the Payment deletion.   
5. If the system fails to update the Payment database due to an error, it logs the error and displays a message to the Administrator.   
6. If the Administrator attempts to delete a Payment that is not associated with a valid Order, the system displays an error message and terminates the process.  
  
Use Case Name: Update Email   
Use Case ID: UC-31   
Actors: Administrator, Customer, Plugin   
Preconditions:   
1. The user (Administrator or Customer) is logged into the system.   
2. The system has access to the Email data entity.   
3. A valid Email record exists in the database for the associated Customer or system notification.   
  
Postconditions:   
1. The selected Email record is successfully updated with new information.   
2. The system updates the Email database with the modified entry.   
3. If the Email is associated with a Customer, the system ensures the new email address is valid and unique.   
  
Main Flow:   
1. The user selects the "Update Email" option from the account settings or email management section.   
2. The system displays the current Email record for the user to modify.   
3. The user enters the new email address or modifies other related information (e.g., subject, message content, for system notifications).   
4. The user submits the updated email information.   
5. The system validates the updated email data (e.g., valid email format, uniqueness for Customer emails).   
6. If validation is successful, the system updates the Email record in the database.   
7. The system provides a confirmation message that the Email was updated successfully.   
  
Alternative Flow:   
1. If the updated email data is invalid (e.g., incorrect email format, missing required fields), the system displays an error message and prompts the user to correct the information.   
2. If the new email address is already associated with another Customer, the system alerts the user and prevents the update to maintain data integrity.   
3. If the selected Email record does not exist or has been deleted, the system displays an error message and returns to the email management screen.   
4. If the system fails to update the Email database due to an error, it logs the error and displays a message to the user.   
5. If the user is a Customer and attempts to update an email template, the system displays an access denied message and terminates the process.  
  
Use Case Name: Delete Email   
Use Case ID: UC-32   
Actors: Administrator, Plugin   
Preconditions:   
1. The Administrator is logged into the system.   
2. The system has access to the Email data entity.   
3. A valid Email record already exists in the database.   
4. The Email is not associated with any active Order or Customer communication that is required for system integrity.   
  
Postconditions:   
1. The selected Email record is successfully removed from the system.   
2. The system updates the Email database by deleting the entry.   
3. If the Email is associated with a Customer, the Customer's email address is updated to reflect the change, if applicable.   
4. The system ensures that no dependent processes rely on the deleted Email unless appropriate safeguards are in place.   
  
Main Flow:   
1. The Administrator selects the "Delete Email" option from the email management section.   
2. The system displays a list of existing Email records, including system-generated emails and customer contact emails.   
3. The Administrator selects the Email record to delete.   
4. The system verifies the Email is not linked to any active or required system processes (e.g., pending order confirmations).   
5. The system prompts the Administrator to confirm the deletion.   
6. The Administrator confirms the deletion.   
7. The system removes the selected Email record from the database.   
8. If the Email is associated with a Customer, the system updates the Customer's contact information accordingly.   
9. The system provides a confirmation message that the Email was deleted successfully.   
  
Alternative Flow:   
1. If the Administrator cancels the deletion, the system returns to the Email list without making any changes.   
2. If the selected Email is associated with an active Order or Payment confirmation that is required for system integrity, the system alerts the Administrator and prevents deletion.   
3. If the selected Email does not exist or has already been deleted, the system displays an error message and returns to the Email list.   
4. If the system fails to update the Customer or Email database due to an error, it logs the error and displays a message to the Administrator.   
5. If the system detects that the deletion could cause data inconsistency (e.g., a required email template), it alerts the Administrator and suggests alternative actions.