Newbies

NBTech
Use-Case Specification

Version 1.0

Revision History

Date	Version	Description	Task	Author
12-07-2025	1.0	Write use-case specification	Section: 1, 5, 6, 7, 8, 15, 16, 17, 18, 26, 27, 28, 29	Nguyễn Thanh Owen
			Section: 2, 3, 4, 9, 14, 19, 20, 21, 22	Lê Nguyên Thảo
			Section: 10, 11, 12, 13, 23, 24, 25, 30, 31, 32	Nguyễn Ngọc Hưng Phát

Table of Contents

1. Use-Case Specification: Sign up

1.1. Brief Description

The "Sign Up" use-case allows new users to create an account in order to access the system. When the user clicks the “Sign Up” button on the home page, the system displays a registration form. The user enters personal information such as full name, email, phone number, password and submits the form. The system validates the input and, if valid, creates a new account and redirects the user to the login page.

1.2. Actor

Guest who has not had an account yet.

1.3. Flow of Events

1.3.1. Basic Flow

- Step 1: The actor (user) clicks the “**Sign up**” button on the **Home page.**
- Step 2: The system will display the registration form with following fields: Full name, Email, Phone Number, Password, Confirm Password.
- Step 3: The user fills out the form and clicks the “**Create Account**” button.
- Step 4: The system will validate the input:
  - Valid email format.
  - Password and Confirm Password must match.
  - Check whether the email already exists.
- Step 5: If the input is valid, the system:
  - Creates a new user account in the database.
  - Displays a success message and redirects the user to the login page.

1.3.2. Alternative Flows

- At step 3:
  - If the user submits the registration form without filling in all required fields, the system will highlight the missing fields and display an appropriate error message.
- At step 4:
  - If the email already exists in the system, the system will display: “Email already exists”.
  - If the password does not match the confirmation field, the system will show a specific message: “Passwords do not match”.

1.4. Special Requirements

- Email must be in a valid form: example@domain.com.
- Email must be unique: The system must not allow users to register with an email that already exists in the database.
- All fields are required to fill: The registration form can not be accepted if any required field is left blank.

1.5. Preconditions

- The user has not registered an account with their email address previously.

1.6. Postconditions

- A new user account is created and stored in the system.
- The user can log in after registration.

1.7. Extension Points

None

2. Use-Case Specification: Login

2.1.Brief Description

The **"Login"** use-case allows a previously registered user to log in to the system by entering their email and password. The login form is accessed via a link on the **Sign Up** page. The system validates the user's input and, if correct, authenticates the user and redirects them to their homepage. If the input is invalid or the credentials are incorrect, appropriate error messages are shown.

2.2.Actor

Guest who has had an account already.

2.3.Flow of Events

2.3.1. Basic Flow

- Step 1: The actor clicks the **"Log in"** link on the **Sign Up** page.
- Step 2: The system will display the registration form with following fields:
    - Input fields: Email, Password.
    - Button: **Log in.**
    - Links: "Don't have an account? *Sign up*" and "Forgot your account's password? *Click here*".
- Step 3: The actor fills in both fields and clicks the **Log in** button.
- Step 4: The system will validate the input:
    - Email and Password are not empty.
    - Email is in a valid format.
- Step 5: If the input is valid, the system:
    - The system logs in the user.
    - Displays a success message.
    - Redirects the user to Homepage.

2.3.2. Alternative Flows

- At step 4:
    - If the user submits the login form without filling in all required fields, the system will highlight the missing fields and display appropriate error messages such as "Please enter your email" or "Please enter your password".
    - If the email does not exist in the system or the password is incorrect, the system will display an error message.
- At step 5:
    - If the user clicks the "Sign up" link next to "Don't have account?", the system will redirect to the Sign Up page.
    - If the user clicks the "Click here" link next to "Forgot your account's password?", the system will redirect to the Forgot Password page.

2.4.Special Requirements

- Email must be in a valid format: The system only accepts email addresses that follow standard format rules, such as example@domain.com.
- Email must exist in the system: The user can only log in using an email that has already been registered.
- Password must match the one stored for that email: The entered password must exactly match the hashed password stored in the system for the provided email.
- All fields are required to fill: Both the Email and Password fields must be filled in before submission. If either is left blank, the system will highlight the missing field and prevent login.

2.5.Preconditions

- The user has already registered with a valid email and password.
- The user accesses the login page via the Sign Up page.

2.6. Postconditions

- If login is successful:
  - The user is authenticated.
  - A session or token is established/stored.
  - The user is redirected to their personal homepage.

2.7. Extension Points

2.7.1. Forgot Password

- **Location**: Step 2
- **Description**: Redirects the user to the password recovery flow.

2.7.2. Sign Up

- **Location**: Step 1
- **Description**: Link to return to the registration page.

3. Use-Case Specification: Forgot Password

3.1. Brief Description

The **"Forgot Password"** use case allows users who have forgotten their password to reset it. The user enters their registered email address, and the system sends a verification code to their email to continue the password reset process.

3.2. Actor

Guest who has had an account but forgot the password.

3.3. Flow of Events

3.3.1. Basic Flow

- Step 1: The user clicks the **“Forgot Password”** link on the **login** page.
- Step 2: The system navigates to the **Forgot Password** page, displaying an input field for the email and a "Send code" button.
- Step 3: The user enters their registered email address and clicks **“Send code”**.
- Step 4: The system checks:
  - If the email is in a valid format.
  - If the email exists in the database.
- Step 5: If the input is valid:
  - The system generates a unique reset code.
  - Stores the code and its expiration time (10 minutes).
  - Sends an email to the user containing the reset code.
  - Displays a success message: "Reset code has been sent to your email.”
  - Redirects the user to the **"Enter Reset Code"** page to proceed with verification.

3.3.2. Alternative Flows

- At step 3: If the email field is left blank, the system displays: "Please enter your email address."
- At step 4: If the email is not found in the database, the system responds with: "Email not found"
- At step 5: If the server encountered an error while sending the code, the system displays: "Something went wrong. Please try again later."

3.4. Special Requirements

- Reset code must:
  - Be unique and randomly generated.
  - Expire after a specific time (10 minutes).

- Email must be in a valid format example@domain.com.
- The system must securely send the email.
- The system should store and verify codes using secure database practices.

3.5.Preconditions

- The user has previously registered an account with a valid email address.
- The user has access to the email they registered with.

3.6.Postconditions

- A reset code is sent to the user's email.
- The system stores the reset request temporarily, linked to the user's email.

3.7.Extension Points

3.7.1. Verify Reset Code

- **Location**: After Step 5
- **Description**: After receiving the code, the user is redirected to a page where they must enter it. The system verifies that the code is correct and not expired.

3.7.2. Reset Password

- **Location**: After successful code verification
- **Description**: After successful code verification, the user can enter and confirm a new password. The system updates the password in the user database and deletes the reset code record.

4. Use-Case Specification: Logout

4.1.Brief Description

The **"Logout"** use-case allows authenticated users to securely exit the system. When the user clicks the **Logout** option from the user menu, the system clears the user's session data. The user interface updates to reflect the logged-out state.

4.2.Actor

User who logged in.

4.3.Flow of Events

4.3.1. Basic Flow

- Step 1: The actor (user) clicks the "**Logout**" button on the **header**.
- Step 2: The system clears all user-related data stored in localStorage.
- Step 3: The system redirects the user to the login page.
- Step 4: The UI updates to reflect the logged-out state.

4.3.2. Alternative Flows

- At step 2: If localStorage does not contain user data or it is already cleared:
  - The system proceeds to Step 3 without error.
  - UI still updates to the logged-out state.

4.4.Special Requirements

- All user session data must be reliably removed from the client side.
- The interface must immediately reflect the logout state.

4.5.Preconditions

- The user is currently logged in and has valid session data stored in localStorage.

4.6. Postconditions

- The user's session data is cleared.
- The user is redirected to the login page and shown as logged out on the interface.

4.7. Extension Points

None

5. Use-Case Specification: View Products List

5.1. Brief Description

This use case allows users (logged in or not) to view the list of products available on the system. Products are displayed based on the selected category such as (Laptop, Laptop Gaming, Flash Sales, View All Best Seller Products,..). Each product is shown with the key information such as name, image, price, discount and rating.

5.2. Actor

Guest or User.

5.3. Flow of Events

5.3.1. Basic Flow

- Step 1: The user navigates to the product listing page via: Homepage, A category menu, A “View All” button in sections like Flash Sale or Best Sellers.
- Step 2: The system retrieves relevant products from the database based on the user’s selection (category, tag, or default list).
- Step 3: The system displays the product list including: product image, product name, price, discount and rating.
- Step 4: The user can:
  - Scroll through the list.
  - Filter or sort the results.
  - Click on a product to view details.

5.3.2. Alternative Flows

- At step 2: If there is an error retrieving the products from the database, the system will display an error message: “Unable to load products. Please try again”.
- At step 4: If the user applies filters or sorting options that result in no matching products, the system will update the product list to empty.

5.4. Special Requirements

None

5.5. Preconditions

- The system has product data available in the database.
- The user is connected to the website or app (logged in or guest).
- The product listing page or category menu is accessible.

5.6. Postconditions

- The system displays a list of products based on the user’s selection or default listing.
- The user can see product details like name, price, image, and rating.
- The system updates the interface if the user applies filters or sorting.
- If no products match, the system displays white space.

5.7. Extension Points

5.7.1. Click to View Product Details

- **Location**: After step 3 in Basic Flow (When the user clicks on a product).
- **Description**: The system responds by loading and displaying the detailed information page for the selected product.

5.7.2. Filter products

- **Location**: After Step 3 (Displaying product list).
- **Description**: The user applies filters (e.g., price range, brand), and the system updates the product list accordingly

5.7.3. Sort products

- **Location**: After Step 3 (Displaying product list).
- **Description**: The user chooses a sorting option (e.g., price ascending, newest first), and the system reorders the displayed products.`

6. Use-Case Specification: View Products Details

6.1. Brief Description

This use case allows any user (logged in or guest) to view comprehensive details of a selected product, including images, specifications, pricing, availability, and user reviews.

6.2. Actor

Guest or User.

6.3. Flow of Events

6.3.1. Basic Flow

- Step 1: The user is browsing the product listing or search results page and clicks on a product to view its details.
- Step 2: The system fetches detailed information of the selected product from the database.
- Step 3: The user fills out the form and clicks the "**Create Account**" button.
- Step 4: The system displays the product detail page, showing:
    - Product images
    - Product name and brand
    - Price, discounts
    - Stock availability
    - Detailed product description and specifications
    - User ratings and reviews
- Step 5: The user can:
    - Click on buttons such as "Add to Cart" (if logged in, redirects to Add to Cart use case).
    - View or write reviews (if logged in, redirects to Review use case)

6.3.2. Alternative Flows

- At step 4:
    - If the product is currently out of stock, the system displays a label: "Out of stock".
    - If no related products are found (no other items in the same price), the **"Related Products"** section is still displayed on the page layout, but it remains **visually empty**, without any product items inside.

6.4. Special Requirements

None

6.5. Preconditions

- The user has access to the product via listing, search, recommendation, or direct link.

6.6. Postconditions

- The user can interact with available actions (e.g., add to cart, rate, view related products).

6.7. Extension Points

6.7.1. Add To Cart

- **Location**: At step 4.
- **Description**: The user can add the product to the shopping cart.

6.7.2. Write Review

- **Location**: At step 4.
- **Description**: Logged-in users may submit ratings and reviews for the product.

6.7.3. Add to cart

- **Location**: At step 4.
- **Description**: System shows recommended or related products at the bottom of the page and users can click to see detailed information about them.

7. Use-Case Specification: Add To Cart

7.1. Brief Description

The "Add To Cart" use-case allows a user to add a product to their shopping cart directly from the Product Detail Page. After the user views the product information and selects the necessary option (quantity), they click the “Add to Cart” button. The system validates the selection and availability, then adds the product to the user’s cart.

7.2. Actor

User who logged in.

7.3. Flow of Events

7.3.1. Basic Flow

- Step 1: The actor (user) is on the **Product Detail Page** of any products**.**
- Step 2: The user selects the quantity of products.
- Step 3: The user clicks the “Add to Cart” button.
- Step 4: The system performs validations.
    - Verifies the user is logged in.
    - Check that quantity does not exceed available stock.
    - Checks whether the product already exists in the shopping cart.
- Step 5: If valid, the system adds the product to the shopping cart.
- Step 6: The system displays a success message and updates the cart.

7.3.2. Alternative Flows

- At step 2: If the user tries to increase the quantity **beyond available stock**, the system:
    - Does not increase the quantity.
    - Displays a message:“The selected quantity exceeds available stock.”
- At step 3:
    - If the user submits the request without being logged in, the system:
        - Does not add the product to the cart.
        - Displays a message: “Please log in to continue.”
        - Optionally redirects the user to the login page.

    - If the product already exists in the shopping cart, the system does not add the product again and displays a message: "This product is already in your cart."
- At step 4: If the quantity of the product is out of stock, the system will display a message: "This product is out of stock".

7.4.Special Requirements

- The system must prevent users from adding a quantity that exceeds the available stock.
- Quantity increment ("+" button) should be restricted when the selected quantity equals the stock limit.
- The system must verify that the user is logged in before allowing the product to be added to the cart.
- If the user is not logged in and attempts to add to cart, the system must prompt the user to log in.

7.5.Preconditions

- The user is viewing a valid Product Detail Page.
- The user has selected a valid quantity (greater than 0).
- The selected quantity does not exceed the available stock.
- If the system requires login to add to cart, the user must be logged in.

7.6.Postconditions

- The selected product and quantity are added to the user's shopping cart.
- The system displays a confirmation message indicating successful addition.
- The system stores the updated cart data in the database.

7.7.Extension Points

None

8. Use-Case Specification: Rating

8.1.Brief Description

The "Rating" use-case allows logged-in users to submit ratings and reviews for a product. After viewing the product detail page, the user selects a rating score, writes an optional review comment, and submits it. The system validates the input and saves the rating associated with the product and user.

8.2.Actor

User who logged in.

8.3.Flow of Events

8.3.1. Basic Flow

- Step 1: The actor (user) is on the **Product Detail Page** of a product.
- Step 2: The user scrolls to and views the rating section located at the bottom of the product detail page.
- Step 3: The system displays the rating form including rating stars and a comment box.
- Step 4: The user selects a rating score (1 to 5 stars) and optionally writes a review comment.
- Step 5: The user submits the rating.
- Step 6: The system validates the input and checks whether the user is logged in.
- Step 7: If valid, the system stores the rating and review in the database and submits the review for moderator approval.
- Step 8: The system displays a success message confirming the submission.

8.3.2. Alternative Flows

- At step 6:
    - If the user submits the rating without selecting a rating score or without entering a review comment, the system:
        - Does not save the rating.

        - Displays a message: “Please provide a rating score and a review comment before submitting.”
        - Returns the user to the rating form to complete the missing information.
    - If the user is not logged in when submitting the rating, the system:
        - Displays a message: “Please log in to continue.”
    - If the user has already submitted a rating for this product, the system:
        - Does not save the new rating.
        - Displays a message: “You can only submit one rating per product”.

8.4. Special Requirements

- Each user can submit only one rating per product.
- Rating scores must be integers between 1 and 5.
- All submitted reviews require moderator approval before being publicly displayed.

8.5. Preconditions

- The user is viewing a valid Product Detail Page.
- The user is logged in.
- The user has not previously submitted a rating for this product.

8.6. Postconditions

- The user’s rating and optional review are stored in the database.
- The review is marked as pending approval by a moderator.

8.7. Extension Points

None

9. Use-Case Specification: Search

9.1. Brief Description

The **"Search"** use-case allows users to find products by entering keywords into a search input field. As the user types, the system dynamically filters the product list in real-time, displaying only the items that match the entered keyword(s). This functionality improves user experience by enabling quick and responsive product discovery without reloading the page.

9.2. Actor

Guest or User.

9.3. Flow of Events

9.3.1. Basic Flow

- Step 1: The actor (user) clicks on the **search bar** and enters a keyword.
- Step 2: The system listens to the input event and retrieves the keyword.
- Step 3: The system converts the keyword to lowercase and trims whitespace.
- Step 4: The system filters the list of products.
- Step 5: The system updates the displayed product list

9.3.2. Alternative Flows

- At step 1: If the user clears the input field or submits an empty string, the system displays the full list of products without applying any filter.
- At step 4: If the filter returns no matching results, the system displays an empty list.
- At step 5: If the user clicks the "Clear Search" button, the system clears the input field, resets the product list, and re-renders all available products.

9.4.Special Requirements

- Case-insensitive matching must be applied.
- The keyword is trimmed of any leading/trailing whitespace.
- The search is handled on the client side using a preloaded product list.
- Search results update dynamically without requiring a form submission or page reload.

9.5.Preconditions

- The user is on a page where the search input and product list are rendered.

9.6.Postconditions

- The filtered product list is displayed based on the entered keyword.

9.7.Extension Points

None

10. Use-Case Specification: Manage Cart

10.1.Brief Description

The "**Manage Cart**" use-case allows a logged-in user to manage their shopping cart by changing the quantity of products, removing items, or proceeding to checkout.

10.2.Actor

User who is logged in.

10.3.Flow of Events

10.3.1. Basic Flow

- Step 1: The user accesses the shopping cart icon on the header.
- Step 2: The system displays the list of items currently in the cart.
- Step 3: The user performs one or more of the following actions:
  - Increase or decrease the quantity of a product.
  - Remove a product from the cart.
- Step 4: Click the "Buy" button to redirect the user to the **Checkout** page.

10.3.2. Alternative Flows

- At step 3:
  - The system limits the quantity to the maximum available stock.
  - If the quantity is reduced to **0**, the system prompts the user to confirm removal. Upon confirmation, the item is removed from the cart.
  - If the product becomes out of stock during cart management, The system notifies the user and removes the item from the cart.
- At step 4:
  - If there is not any product in the cart, the system notifies the user and does not allow the user to click the "**Buy**" button.

10.4.Special Requirements

- The system must enforce quantity constraints based on real-time stock levels.

10.5.Preconditions

- The user is logged in.

10.6.Postconditions

- The cart reflects updated product quantities and total price.

- If "Buy" is selected, the user is redirected to the Checkout page.

10.7. Extension Points

None

11. Use-Case Specification: Checkout

11.1. Brief Description

The "**Checkout**" use-case allows a logged-in user to complete a purchase by entering shipping information, applying a coupon (if any), and submitting the order.

11.2. Actor

User who is logged in.

11.3. Flow of Events

11.3.1. Basic Flow

- Step 1: The user clicks "**Buy**" on the shopping cart page.
- Step 2: The system displays the checkout form and the list of products.
- Step 3: The user fills in shipping information (name, street address, town/city, phone number, email).
- Step 4: The user optionally enters a coupon code.
- Step 5: The system validates the input and calculates the total amount.
- Step 6: The user clicks "**Place Order**".
- Step 7: The system confirms the order, generates an order ID, stores order data, displays a confirmation message and redirects the user to the **My Order** page.

11.3.2. Alternative Flows

- At step 4:
  - If the coupon is invalid, the system shows an error message and excludes the coupon from total price calculation.
- At step 5:
  - If the required fields are not fully filled, the system highlights the problematic fields and prevents submission.

11.4. Special Requirements

- All the required fields must be filled in

11.5. Preconditions

- The user is logged in.
- There is at least one product.

11.6. Postconditions

- The order is stored in the database.
- The cart is cleared.
- The user is shown a confirmation.
- The user is redirected to **My Order** page.

11.7. Extension Points

None

12. Use-Case Specification: View Orders History

12.1. Brief Description

The “**View Orders History”** use-case enables users to view all of their past orders, including order statuses and key details such as date, total price, and shipping status.

12.2. Actor

User who is logged in.

12.3. Flow of Events

12.3.1. Basic Flow

- Step 1: The user clicked on the “**Place Order**” button on the **Checkout** page, or clicked on the “**My Order**” button
- Step 2: The system displays the list of orders with relevant details: list of products with quantity and price, total price, date, status.

12.3.2. Alternative Flows

- At step 2:
    - If there is no order, the system displays blank space.

12.4. Special Requirements

- Orders must be sorted in reverse chronological order (latest first).

12.5. Preconditions

- The user is logged in.

12.6. Postconditions

None

12.7. Extension Points

None

13. Use-Case Specification: View User’s Account

13.1. Brief Description

The “View User’s Account” use-case allows a logged-in user to view and update personal information (name, email, phone, address) and change their password securely.

13.2. Actor

User who is logged in.

13.3. Flow of Events

13.3.1. Basic Flow

- Step 1: The user clicked on the “**My Profile**” button from the navigation.
- Step 2: The system displays the current user profile, including:
    - Name
    - Email
    - Phone number
    - Address
    - Change password button
- Step 3: The user edits any of the visible fields and clicks **“Save Changes”**.

- Step 4: The system validates the data and updates the account.
- Step 5: The system shows confirmation: **“Saved!”**

13.3.2. Alternative Flows

- At step 3:
  - If the user clicks **“Change Password?”**, the system displays three input fields:
    - Current Password
    - New Password
    - Confirm New Password
- At step 4:
  - If “**current password is incorrect**” appears:
    - The process is aborted.
  - If new password and confirmation do not match:
    - The system shows error: **“New passwords do not match”**
    - The process is aborted.
- At step 5:
  - If all validations pass:
    - The system updates the password.
    - The system shows a message: **“Saved!”**

13.4.Special Requirements

- Passwords must be hidden on input and stored encrypted.
- The current password must be verified against the database.
-

13.5.Preconditions

- The user is authenticated (logged in).

13.6.Postconditions

- Updated profile and/or password is saved in the system.
- A success message is displayed if the update is successful.

13.7.Extension Points

None

14. Use-Case Specification: Login (Admin only)

14.1.Brief Description

The **"Login for Admin"** use-case allows an admin user to log into the system by entering a valid email or phone number and password. The login form is accessed from the admin login page. The system validates the inputs and, upon successful authentication, redirects the user to the appropriate admin dashboard based on their role. If the login fails due to invalid input or insufficient privileges, the system displays appropriate error messages.

14.2.Actor

Admin (a_admin or p_admin).

- a_admin who manages accounts,
- p_admin who manages products, reviews, coupons,..

14.3.Flow of Events

14.3.1. Basic Flow

- Step 1: The actor accesses the **admin login page**.
- Step 2:The system displays the login form with the following fields and button: Identifier (email or phone),

Password and "**Login**" Button

- Step 3: The actor fills in the fields and clicks the **Login** button.
- Step 4: The system will validate the input:
  - Both fields are not empty.
  - Identifier follows the correct format (valid email or phone number).
- Step 5: If the input is valid, the system:
  - The system sends a login request to the backend API.
  - The backend authenticates the user and checks the user's role.
- Step 6: Based on the role:
  - If the role is a_admin, the user is redirected to the **User Account Management** page.
  - If the role is p_admin, the user is redirected to the **Dashboard** page.
  - Otherwise, the system shows an alert "You do not have permission to access this page."

14.3.2. Alternative Flows

- At step 5: If the login fails
  - Backend-provided message "Login failed"
  - In case of server issues: "Server error. Please try again."

14.4.Special Requirements

- Email or phone number must be in a valid form.
- Only users with the role a_admin or p_admin can access this login route.
- All fields are required to fill: The registration form can not be accepted if any required field is left blank.

14.5.Preconditions

- The admin user has been previously registered in the system.

14.6.Postconditions

- If login is successful:
  - The system authenticates the admin.
  - The admin's role is verified.
  - A session (via localStorage) is established.
  - The admin is redirected to their corresponding admin dashboard.

14.7.Extension Points

None

15. Use-Case Specification: View List Of Accounts (Admin Only)

15.1.Brief Description

The **"View List of Accounts"** use-case allows an admin user to view a list of all user accounts in the system. The feature includes basic filtering/search functionality (by name, email, phone, etc.) and the ability to delete accounts if needed. The system displays a table with account details such as name, email and role. If a user without sufficient permissions tries to access the page, an error message will be shown.

15.2.Actor

Admin managing accounts (User with the role **a_admin)**

15.3.Flow of Events

15.3.1. Basic Flow

- Step 1: The actor logs in as an admin.

- Step 2: The system sends a request to the backend API to retrieve the list of user accounts.
- Step 3: The backend responds with the list of accounts including details such as name, email, and role.
- Step 4: The system displays the list of accounts in a table format with columns: Name, Email, Role, and Actions (Details, Delete, Edit).
- Step 5: The actor can enter a keyword into the search bar (e.g., name, email, phone) to find accounts that have related information.

15.3.2. Alternative Flows

- At step 3: if the backend returns no user accounts, the system displays an empty list (no accounts shown).
- At step 4: when the actor clicks **Delete** on an account, the system immediately displays a confirmation dialog: “Are you sure you want to delete this account?”
  - If the actor **confirms**, the system sends a delete request to the backend to remove the account from the database.
  - If the actor **cancels**, the system closes the confirmation dialog without deleting the account.
- At step 5: If the actor enters a keyword that does not match any user account, the system displays an empty list (no accounts shown).

15.4. Special Requirements

- Only users with the role **a_admin** are authorized to access the account list and perform delete, view detail, or edit actions.
- The system must display the account list with up-to-date information retrieved from the backend API.
- Deletion of an account must be confirmed by the user via a confirmation dialog before proceeding.

15.5. Preconditions

- The actor is a registered user with the role **a_admin** (admin managing accounts).
- The system backend and database are available and operational.
- The user account data exists in the database to be retrieved and displayed.

15.6. Postconditions

- The system successfully displays the requested list of user accounts (filtered or full list).
- If an account was deleted, it is permanently removed from the database and no longer appears in the list.

15.7. Extension Points

15.7.1. Search

- **Location:** Step 5 – When the actor enters a keyword into the search bar to filter the account list.
- **Description:** This extension filters the currently displayed list of user accounts directly on the client side based on the entered keyword (e.g., name, email, phone). The system validates the input and dynamically updates the displayed table to show only the accounts matching the search criteria. If no accounts match, an empty list is displayed.

15.7.2. Delete

- **Location:** Step 4 – When the actor selects the Delete action for a specific account in the list.
- **Description:** This extension handles the deletion of a user account by sending a delete request directly to the backend API to remove the account from the database. Upon confirmation from the actor, the system displays a confirmation dialog, then sends the delete request. If the deletion is successful, the system updates the account list by removing the deleted account from the view.

16. Use-Case Specification: View Account Details (Admin Only)

16.1. Brief Description

The "View Account Details" use-case allows an admin user to view full information of a specific user account.

This function is accessed from the user account list page, where the admin can click a "Detail" action on any listed account. The system retrieves and displays the account's complete profile from the backend.

16.2.Actor

Admin managing accounts (User with the role **a_admin)**

16.3.Flow of Events

16.3.1. Basic Flow

- Step 1: The actor logs in as an admin.
- Step 2: The system displays the list of user accounts, each with associated data already loaded.
- Step 3: The actor clicks the **Detail** action for a specific account.
- Step 4: The system displays the full details of the selected account using the preloaded data including:
  - Name
  - Email
  - Phone Number
  - Password
  - Address
  - Role

16.3.2. Alternative Flows

- At step 4: if any field in the preloaded account data is missing (e.g., phone number or address is null), the system still displays the account detail view and any missing fields will be shown as **empty**

16.4.Special Requirements

- Only users with the role a_admin are authorized to access and view account details.
- If any data field is missing (e.g., phone number, address), the system must display an **empty value.**
- Sensitive fields such as passwords must be masked (e.g., ********) or displayed according to admin permissions.
- The detail view should include a “Back” or “Return” button to allow easy navigation back to the account list.

16.5.Preconditions

- The actor is a registered user with the role **a_admin** (admin managing accounts).
- The selected account exists in the preloaded data.

16.6.Postconditions

- The system successfully displays the detailed information of the selected user account using the preloaded data.
- Any missing fields are shown as empty.
- The admin remains on the detail view page and may choose to return to the account list

16.7.Extension Points

None

17. Use-Case Specification: Create New Account (Admin Only)

17.1.Brief Description

The "Create New Account" use-case allows an admin (a_admin) to register a new user account in the system. The admin accesses a registration form, fills in the required information, and submits it. The system validates the input and creates the new account in the backend database.

17.2. Actor

Admin managing accounts (User with the role **a_admin)**

17.3. Flow of Events

17.3.1. Basic Flow

- Step 1: The actor clicks the **"Create"** button from the **User Account Management page**.
- Step 2: The system displays the account creation form with the following fields:
  - Name
  - Email
  - Phone Number
  - Address
  - Role
- Step 3: The actor fills in the form and submits it.
- Step 4: The system validates the input:
  - All required fields are filled.
  - Email and phone number are in valid formats.
- Step 5: If validation passes, the system sends a request to the backend to create the new account.
- Step 6: Upon success, the system displays a success message and adds the new account to the list.

17.3.2. Alternative Flows

- At step 2: If the admin does not select any role (user, a_admin, p_admin), the system automatically assigns the default role: user. No error is shown.
- At step 4: If any field is invalid or missing, the system shows a message: “Please enter required information,”
- At step 5: if the backend detects that the email is already associated with another account, the system shows a message: “This email is already registered.”

17.4. Special Requirements

- Only users with the a_admin role can access the Create New Account form.
- Email must be in valid formats and unique within the system.
- Duplicate emails must be rejected to prevent conflicts.

17.5. Preconditions

- The actor is logged in and authenticated as a user with the a_admin role.

17.6. Postconditions

- If the account creation is successful:
  - A new user account is created and saved in the system database with the specified details and role.
  - The newly created account appears in the user account list.
- If the account creation fails:
  - No new account is created or saved.

17.7. Extension Points

None

18. Use-Case Specification: Edit Account (Admin Only)

18.1. Brief Description

The "Edit Account" use-case allows an admin user to modify the details of an existing user account, including name, email, phone number, address, password, and role. The admin accesses the edit form from the user account list, updates the necessary fields, and submits the changes.

18.2. Actor

Admin managing accounts (User with the role **a_admin)**

18.3. Flow of Events

18.3.1. Basic Flow

- Step 1: The actor selects an account from the user account list and clicks the **Edit** action/button.
- Step 2: The system displays the account edit form, pre-filled with the existing account details:
  - Name
  - Email
  - Phone Number
  - Address
  - Role
- Step 3: The actor modifies the desired fields.
- Step 4: The actor submits the edit form.
- Step 5: The system validates the input.
- Step 6: The backend updates the account data in the database.
- Step 7: The system displays a success message, and prefills the form with new information.

18.3.2. Alternative Flows

- At step 5:
  - If any required field is empty
    - The system displays appropriate error messages next to the affected fields.
    - The actor must correct the input before submitting again.
  - If the email is already used by another account, The system shows an error message: "This email is already in use."

18.4. Special Requirements

- Only users with the a_admin role can access and perform account edits.
- Email must be validated for the correct format before submission.
- Duplicate emails must be rejected to prevent conflicts.

18.5. Preconditions

- The actor is logged in and authenticated as a user with the a_admin role.
- The selected account exists in the preloaded data.

18.6. Postconditions

- If the edit operation is successful:
  - The updated account information is saved in the system database.
  - The changes are immediately reflected in the user account list.
- If the edit operation fails:
  - No changes are saved.

18.7. Extension Points

None

19. Use-Case Specification: View Statistics (Admin Only)

19.1. Brief Description

The **"View Statistics"** use-case allows an admin user to view overall business performance including total revenue, number of products sold, and number of orders. The statistics can be filtered by day, month, or year using a dropdown and input fields. The system fetches this data from the backend and displays it in a dashboard format.

19.2. Actor

Admin managing accounts (User with the role **a_admin)**

19.3. Flow of Events

19.3.1. Basic Flow

- Step 1: The actor logs in as an admin.
- Step 2: The actor is redirected to the dashboard page.
- Step 3: The system loads and verifies the presence of admin data in localStorage.
- Step 4: If admin data exists, the system shows the dashboard and displays the admin's name.
- Step 5: The system initializes the statistics display using all available data.
- Step 6: The actor selects a filter option (day, month, or year) from the dropdown menu.
- Step 7: The system shows relevant input fields (day, month, year) depending on the filter type.
- Step 8: When the actor inputs a value (or changes selection), the system calculates statistics using local data and updates:
    - Total revenue.
    - Products sold.
    - Number of orders.

19.3.2. Alternative Flows

- At step 3: If no admin is found in localStorage, the system automatically redirects to the login page
- At step 6: If the actor selects a filter type but leaves the required input(s) (e.g. date/year) empty:
    - The system does **not** update statistics.
    - The previous statistics remain displayed.
    - No alert is shown, but you can enhance the UX by prompting the user (currently not implemented).

19.4. Special Requirements

- Only users with the role **a_admin** can access the dashboard. The admin session is stored and checked in localStorage.
- The system must prevent access to the dashboard via the back button after logout.
- Admin must manually log out to clear their session.

19.5. Preconditions

- The actor is authenticated.
- The page has access to the embedded or static list of orders and products.

19.6. Postconditions

- The system displays the calculated statistics based on the selected filter.
- Admin can log out and is redirected appropriately.
- If admin logs out, session data is cleared, and dashboard access is restricted.

19.7. Extension Points

19.7.1. Filter

- **Location:** Step 6
- **Description:** When the actor changes the filter type, the system shows the relevant input fields so the actor can enter valid filter data.

20. Use-Case Specification: View list of reviews (Admin Only)

20.1. Brief Description

The **"View List of Reviews"** use-case allows an admin user to view, search, filter, update, and delete product reviews submitted by users. The system retrieves review data from the backend API and displays them in a list/table format. Admins can search by keywords and filter by review status. Actions include viewing details, editing, and deleting reviews.

20.2. Actor

Admin managing accounts (User with the role **a_admin)**

20.3. Flow of Events

20.3.1. Basic Flow

- Step 1: The actor logs in as an admin.
- Step 2: The system sends a request to the backend API to retrieve all review records.
- Step 3: The backend responds with a list of reviews.
- Step 4: The system displays the list of reviews in a structured UI with:
  - Review ID, User ID, Product ID, Rating, Date, and Actions.
  - Action buttons: Details, Edit, Delete.
- Step 5: The actor can:
  - Use the search bar to filter reviews by ID, user ID, product ID, or date.
  - Use the dropdown to filter reviews by status.

20.3.2. Alternative Flows

- At step 3: If the backend returns an empty list, the system displays no reviews and shows an empty state.
- At step 4: Delete
  - When the actor clicks **DELETE**, a confirmation dialog appears: "Are you sure you want to delete review *"reviewId"*?"

  - If confirmed: the system sends a **DELETE** request
    - If successful: the system re-fetches and re-renders the updated list.
    - If the request fails: an error alert is shown.
  - If canceled: no action is taken.
- At step 5: If no matches are found for the entered search keyword or selected status, the system displays an empty list.

20.4. Special Requirements

- Only users with the role **a_admin** are authorized to view and manage reviews.
- The system must always fetch the most recent review data from the backend.
- Deletion must be confirmed before executing the backend request.
- The UI should dynamically adjust its height and layout based on content size.

20.5. Preconditions

- The backend and database are operational.
- Review records exist in the system.

20.6. Postconditions

- The system displays the list of reviews according to the applied search or filter.
- If a review was deleted, it is permanently removed from both the database and the displayed list.

20.7. Extension Points

20.7.1. Delete

- **Location:** Step 4
- **Description:** Sends a DELETE request to the backend for a selected review. Upon success, the list is refreshed.

20.7.2. Filter

- **Location:** Step 6
- **Description:** Filters reviews by status using a dropdown menu. The list updates accordingly.

20.7.3. Search

- **Location:** Step 6
- **Description:** Filters the list of reviews on the client side based on a keyword match

21. Use-Case Specification: View review's details (Admin Only)

21.1. Brief Description

The **"View Review's Details"** use-case allows an admin to view the full information of a selected product review. When the admin clicks the "Details" link from the review list, the system extracts the review ID from the URL, fetches the detailed review data from the backend, and displays it on the details page.

21.2. Actor

Admin managing accounts (User with the role **a_admin)**

21.3. Flow of Events

21.3.1. Basic Flow

- Step 1: The actor clicks on the "Details" button for a review from the review list.
- Step 2: The system retrieves the review ID from the query string in the URL.
- Step 3: The system sends a GET request to fetch the review data.
- Step 4: If the data is successfully fetched, the system populates the following fields on the page:
  - Review ID
  - Product ID
  - User ID
  - Rating
  - Comment
  - Date
  - Status
- Step 5: The system also updates the height and visual layout of the review detail page dynamically to ensure proper display.

21.3.2. Alternative Flows

- At step 3: If the review ID is missing in the URL, the system cancels the request and does not attempt to fetch data.
- At step 4: If the request to the server fails, the system logs the error and shows an alert with the message: "Failed to load review data."

21.4. Special Requirements

- Only users with the role **a_admin** should be able to access this functionality.
- The page layout should automatically adjust its height to fit all content.

21.5. Preconditions

- Admin is authenticated and authorized.
  The review ID is included in the URL as a query parameter.

21.6. Postconditions

- The system displays the complete details of the selected review.
- If the admin refreshes the page, the data is reloaded using the ID in the URL.

21.7. Extension Points

None

22. Use-Case Specification: Edit review (Admin Only)

22.1. Brief Description

The **"Edit Review"** use-case allows an admin to modify the information of a selected product review. After selecting a review (from the details or list view), the admin can update fields such as rating, comment, date, and status. Upon submission, the system validates the inputs and sends an update request to the backend.

22.2. Actor

Admin managing accounts (User with the role **a_admin)**

22.3. Flow of Events

22.3.1. Basic Flow

- Step 1:The actor clicks on the "Edit" button from the review list to access the edit review page.
- Step 2: The system parses the reviewId from the URL.
- Step 3: The system sends a request to retrieve the review data from the backend using the review ID.
- Step 4: If the data is fetched successfully, the system fills in the edit form with:
  - Product ID
  - User ID
  - Rating
  - Comment
  - Date
  - Status
- Step 5: The actor modifies one or more fields and clicks the **"Update"** button.
- Step 6: The system validates required fields (Product ID, User ID, Rating).
- Step 7: The system prompts the user with a confirmation dialog.
- Step 8: Upon confirmation, the system sends a request with the updated data.
- Step 9: If the update is successful, a success message is shown and the user is redirected to the review list page.

22.3.2. Alternative Flows

- At Step 2: If reviewId is missing from the URL, the system shows an alert: "Review ID not found in the URL." and halts the process.
- At Step 4: If the attempt to retrieve the review data from the server fails, the system logs the error and shows an alert: "Failed to load review data."
- At Step 6: If any required field is empty or invalid, the system shows an alert: "Please fill in all required fields." and prevents submission.
- At Step 8: If the attempt to submit the updated data to the server fails, the system logs the error and shows an alert: "An error occurred while updating the review."

22.4.Special Requirements

- Only users with the role **a_admin** are authorized to view and manage reviews.
- The form inputs must be validated before sending the request.
- The system must format the date into an HTML-compatible format before filling the input field.
- The page layout must dynamically adjust height when the page loads.

22.5.Preconditions

- Admin is logged in and authorized.
- The reviewId is present in the URL as a query parameter.
- The review data exists in the backend.

22.6.Postconditions

- If the update is successful, the review is updated in the backend.
- The admin is redirected to the review list page after confirmation.
- If the page is refreshed, the review data is reloaded using the ID from the URL.

22.7.Extension Points

None

23. Use-Case Specification: View List of Orders (Admin Only)

23.1.Brief Description

The "**View List of Orders**" use-case allows the admin to view all customer orders in a tabular format. The system supports search by keyword and filtering by order status. Admins can also delete an order or change the status of an order..

23.2.Actor

Admin managing products (User with the role **p_admin)**

23.3.Flow of Events

23.3.1. Basic Flow

- Step 1: The actor logs in as **p_admin**.
- Step 2: The system sends a request to the backend to fetch all order records.
- Step 3: The backend responds with order data including ID, date, customer name, phone number, total price, and status.
- Step 4: The system displays the orders in a table format, with each row showing order info and actions (Details, Edit, Delete).
- Step 5: The actor can:
  - Use the search bar to find orders by ID, date, full name, phone number, or total.
  - Use the status dropdown to filter orders.

23.3.2. Alternative Flows

- At step 3:
  - If there are no orders, the system displays blank space.
- At step 4:
  - If the actor clicks **Delete** on an order:
    - The system immediately displays a confirmation dialog: "Are you sure you want to delete this order?"
    - If the actor **confirms**, the system sends a delete request to the backend to remove the order from the database.
    - If the actor **cancels**, the system closes the confirmation dialog without deleting the account.

23.4.Special Requirements

- Only users with the role **p_admin** are authorized to view and manage orders.
- Deleting an order must require confirmation before proceeding.

23.5.Preconditions

- The actor is logged in with the p_admin role.
- Orders exist in the backend database.

23.6.Postconditions

- Orders are displayed or filtered correctly.
- Deleted orders are removed from the list and backend.

23.7.Extension Points

23.7.1. Delete

- **Location:** Step 4
- **Description:** This extension handles the deletion of an order by sending a delete request directly to the backend API to remove the order from the database. Upon confirmation from the actor, the system displays a confirmation dialog, then sends the delete request. If the deletion is successful, the system updates the order list by removing the deleted order from the view.

23.7.2. Filter

- **Location:** Step 5
- **Description:** This extension enables the actor to filter orders based on their status using a dropdown menu. When a status is selected, the system updates the order list to show only orders matching that status. If no orders match the selected status, an empty list is shown.

23.7.3. Search

- **Location:** Step 5
- **Description:** This extension filters the currently displayed list of orders directly on the client side based on the entered keyword (e.g., name, email, phone). The system validates the input and dynamically updates the displayed table to show only the orders matching the search criteria. If no orders match, an empty list is displayed.

24. Use-Case Specification: View Order Details (Admin Only)

24.1. Brief Description

The "**View Order Details**" use-case allows a **p_admin** user to view complete information of a specific customer order. This includes shipping details and the list of purchased products, total price, order date, and current order status.

24.2. Actor

Admin managing products (User with the role **p_admin)**

24.3. Flow of Events

24.3.1. Basic Flow

- Step 1: The actor logs in as **p_admin**.
- Step 2: The system displays a list of existing orders.
- Step 3: The actor clicks the **Details** link of a specific order.
- Step 4: The system sends a request to the backend to retrieve full order information.
- Step 5: The backend responds with:
  - Shipping Information: name, email, phone number, address.
  - Order Details: list of products (name, quantity, price, image), total cost, order date, and order status.
- Step 6: The system displays all the retrieved data on a dedicated view page.

24.3.2. Alternative Flows

None

24.4. Special Requirements

- Only users with the role **p_admin** are authorized to view order details.
- The system must ensure that the displayed information is consistent with the backend database.

24.5. Preconditions

- The actor is logged in with the **p_admin** role.
- Orders exist in the backend database.

24.6. Postconditions

- The system displays detailed information about the selected order.

24.7. Extension Points

None

25. Use-Case Specification: Edit Order (Admin Only)

25.1. Brief Description

The "**Edit Order**" use-case allows a **p_admin** user to update the status of a specific customer order. The admin accesses the edit form from the order list. The form displays shipping and order details, and the admin can change the order's status using a dropdown. Upon saving, the system asks for confirmation before updating the status in the backend.

25.2. Actor

Admin managing products (User with the role **p_admin)**

25.3. Flow of Events

25.3.1. Basic Flow

- Step 1: The actor logs in as **p_admin**.
- Step 2: The system displays a list of orders.
- Step 3: The actor selects an order and clicks the **Edit** icon/button.
- Step 4: The system sends a request to the backend to retrieve the order data.
- Step 5: The backend responds with:
  - Shipping Information: name, email, phone number, address
  - Order Details: list of products (name, price, quantity, image), total cost, date
  - Order Status
- Step 6: The system displays an edit form with the retrieved data, where all fields are non-editable except the status, which is displayed as a dropdown.
- Step 7: The actor changes the order status and clicks **Save Changes**.
- Step 8: The system displays a confirmation dialog: "Are you sure you want to change the status of the order?"
- Step 9: If the actor confirms, the system sends a request to the backend to update the status in the database.
- Step 10: The backend updates the order record and responds with success.
- Step 11: The system shows a success message and refreshes the order data with the new status.

25.3.2. Alternative Flows

- At step 9:
  - If the actor cancels the confirmation dialog, the system closes the dialog and does not send any update request.

25.4. Special Requirements

- Only users with the role **p_admin** are authorized to view and manage orders.
- All order details except status are read-only.
- A confirmation dialog must appear before applying any status change.

25.5. Preconditions

- The actor is logged in with the p_admin role.
- The selected order exists in the database.

25.6. Postconditions

- If the status is changed:
  - The order's status is updated in the backend.
  - The updated status is displayed in the system.
- If canceled:
  - No change is made to the order.

25.7. Extension Points

None

26. Use-Case Specification: View List of Products (Admin Only)

26.1.Brief Description

The “View List of Products” use-case allows the admin (p_admin role) to view, search, filter, and manage the list of products in a tabular or card format. Admins can view details, edit a product, or delete it after confirmation. Search is performed on product name or ID, and filtering is based on product category.

26.2.Actor

Admin managing products (User with the role **p_admin)**

26.3.Flow of Events

26.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The system sends a request to the backend to fetch all products.
- Step 3: The system displays the products in a table format, with each row showing order info and actions (Details, Edit, Delete).
- Step 4: The actor can:
  - Use the search bar to find orders by ID or Product Name.
  - Use the status dropdown to filter.

26.3.2. Alternative Flows

- At step 3:
  - If there are no products, the system displays blank space.
  - If the actor clicks **Delete** on an order:
    - The system immediately displays a confirmation dialog: “Are you sure you want to delete this order?”
    - If the actor **confirms**, the system sends a delete request to the backend to remove the order from the database.
    - If the actor **cancels**, the system closes the confirmation dialog without deleting the account.
- At step 4:
  - If the actor selects a **product category** from the **filter dropdown**:
    - The system reads the selected category value.
    - The system filters the current list of products to display only those matching the selected category.
    - The product list in the UI is updated dynamically to reflect the filtered results.
  - **If** no products match the selected category, the system displays blank space.

26.4.Special Requirements

- Only users with the role p_admin are authorized to view and manage products.
- Deleting a product must require user confirmation before proceeding.
- The user interface must dynamically update the product list based on filtering and search actions.

26.5.Preconditions

- The actor is logged in with the p_admin role.
- Products exist in the backend database.

26.6.Postconditions

- Products are correctly displayed and updated in the UI.

- Deleted products are removed from both the frontend list and the backend database.

26.7. Extension Points

26.7.1. Delete

- **Location:** Step 3
- **Description:** Handles product deletion. Upon user confirmation, sends a delete request to the backend. If successful, the product is removed from the UI.

26.7.2. Filter

- **Location:** Step 4
- **Description:** Allows filtering of products by category. The system updates the displayed list to match the selected category. If no products match, an empty list is shown.

26.7.3. Search

- **Location:** Step 4
- **Description:** Allows client-side searching by product name or ID. The displayed product list is updated dynamically as the user types. If no results match, an empty list is shown.

27. Use-Case Specification: View Product's Details (Admin Only)

27.1. Brief Description

The **"View Product's Details"** use-case allows the admin (p_admin role) to view full details of a specific product. This includes basic information (name, price, stock, etc.), product images, highlights, category-specific specifications (like CPU, battery, DPI...), and metadata like stock date and sold quantity. This view is used to verify or manage product information.

27.2. Actor

Admin managing products (User with the role **p_admin)**

27.3. Flow of Events

27.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The actor navigates to the **Product Details page** (e.g., via clicking a "Details" button from the product list).
- Step 3: The system reads the **product ID** from the URL (?id=...).
- Step 4: The system sends a **GET request** to the backend API.
- Step 5: The backend responds with the full product data including:
  - Name, price, category, stock
  - Description and highlights
  - Images (up to 4)
  - Discount, stock date, quantity sold
  - Category-specific specifications (e.g., CPU, DPI, Switch, Battery, etc.)
- Step 6: The system dynamically displays this information in a structured and styled layout, with:
  - General info at the top
  - Highlights and description sections
  - Images
  - A specifications section that changes depending on the product category

27.3.2. Alternative Flows

- At step 5: If some product fields (e.g., images, highlights, specs) are missing:
    - The system leaves those sections blank.
    - The rest of the product data is still displayed properly.
- At step 3: If the URL does not contain a product ID, no product details are displayed.

27.4. Special Requirements

- Product specifications must be dynamically generated based on the product category using a predefined mapping.
- The system must handle missing or undefined product fields gracefully.

27.5. Preconditions

- The actor is logged in as p_admin.
- The product exists in the database and is accessible via its ID.

27.6. Postconditions

- The complete product details are displayed correctly on the page.

27.7. Extension Points

27.7.1. Dynamic Specifications

- **Location:** Step 6
- **Description:** Dynamically loads and displays specifications based on product category. Uses a predefined mapping to generate specifications labels and values.

28. Use-Case Specification: Create Product (Admin Only)

28.1. Brief Description

The **"Create Product"** use-case allows an admin (p_admin role) to add a new product to the system by filling in the required fields such as name, price, category, stock, images, description, highlights, and specifications. The system validates the input and sends a request to store the new product in the backend database.

28.2. Actor

Admin managing products (User with the role **p_admin)**

28.3. Flow of Events

28.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The actor navigates to the **Create Product** page.
- Step 3: The system loads the list of available categories and populates the category dropdown.
- Step 4: The actor selects a product category.
- Step 5: The system dynamically generates and displays specification fields based on the selected category.
- Step 6: The actor fills in the following required fields:
    - Name, price, category, brand
    - Description, stock, discount, stock date
    - 4 image URLs
    - Product highlights
    - Category-specific specifications

- Step 7: The actor clicks the "Submit" button.
- Step 8: Upon success, the system sends a request to add a new product with the full product object and redirects the actor to the product list page.

28.3.2. Alternative Flows

- At step 4: If no category is selected:
  - The specification section remains hidden.
  - The actor cannot proceed without selecting a valid category.
- At step 6: If any required field is left empty or invalid:
  - The system displays an alert: "Please enter complete product information before creating."
  - Product is not submitted.
- At step 7: If the actor cancels the confirmation dialog:
  - The product is not submitted.
  - The actor remains on the Create Product page.

28.4. Special Requirements

- The system must support dynamic form generation for specifications per category.
- 4 images, 1 or more highlights, and complete specifications are required.
- The system must include a category selection to classify products (e.g., Laptop, Mouse, Keyboard).

28.5. Preconditions

- The actor is logged in as p_admin.
- The backend endpoint is available.

28.6. Postconditions

- A new product is stored in the backend database.
- The actor is redirected to the product listing page

28.7. Extension Points

28.7.1. Category Selection Change

- **Location:** Step 4
- **Description:** When the user selects a product category, the system dynamically loads and displays the corresponding specification input fields relevant to that category. This ensures that only applicable fields are shown, improving user experience and data accuracy.

28.7.2. Highlight Addition

- **Location**: Step 6
- **Description**: Users can add multiple highlight points dynamically by clicking an “Add Highlight” button, which appends new input fields without page reload.

29. Use-Case Specification: Edit Product (Admin Only)

29.1. Brief Description

The **“Edit Product”** use case allows an admin user (with the p_admin role) to modify existing product information in the system. The admin can update various fields including name, price, category, description, images, stock, discount, highlights, and specifications depending on the selected category.

29.2. Actor

Admin managing products (User with the role **p_admin)**

29.3. Flow of Events

29.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The actor navigates to the **Product Edit page**, typically by clicking a “Edit” button from the product list.
- Step 3: The system calls an API to fetch the product data based on the **productId**.
- Step 4: The system displays the retrieved product data in the form fields: name, price, description, category, images, stock, discount, date, sold count, etc.
- Step 5: Based on the product category, the system dynamically generates and displays the relevant specification input fields.
- Step 6: The actor modifies any desired fields and may click “Add Highlight” to append more highlight input boxes.
- Step 7: The actor clicks the **“Save”** button.
- Step 8: The system displays a confirmation dialog: “Are you sure you want to update this product?”.
- Step 9: Upon confirmation, the system validates the form and sends a PUT request to the backend with the updated product object.
- Step 10: On success, the system shows a success message and reloads the page with the updated product data.

29.3.2. Alternative Flows

- At step 7: If any required field is left empty or contains invalid data (e.g., price is not a number, or a highlight is missing):
  - the system displays an alert: “Please enter complete and valid product information before submitting.”
  - Product is not submitted.
  - The actor remains on the Edit Product page to correct the data.
- At step 8: If the actor clicks **“Cancel”** in the confirmation dialog after pressing the "Save" button:
  - No data is sent to the server.
  - The actor remains on the Edit Product page, with all inputs preserved.

29.4. Special Requirements

- The system must automatically generate input fields for product specifications based on the selected product category (e.g., Laptop, Mouse, Keyboard).
- All required fields—including name, category, price, stock, discount, description, date, brand, specifications, highlights, and image URLs—must be validated on the client side before submission.

29.5. Preconditions

- The actor is logged in as p_admin.
- The backend endpoint is available.
- Product Exists in Database.

29.6. Postconditions

- Product Data is Successfully Updated.
- Confirmation is Displayed to the Admin.
- The system reloads or redirects to the same Edit Product page, now reflecting the newly saved data.

29.7. Extension Points

29.7.1. Dynamic Specification Generation

- **Location:** Step 5
- **Description:** When the product data is loaded (including its category), the system dynamically generates and displays the corresponding specification input fields based on a predefined mapping. This ensures that

only relevant fields are shown, improving user accuracy and efficiency.

29.7.2. Highlight Addition

- **Location**: Step 6
- **Description**: Users can add multiple highlight points dynamically by clicking an “Add Highlight” button, which appends new input fields without page reload.

30. Use-Case Specification: View list of coupons (Admin Only)

30.1. Brief Description

The “**View list of coupons**” use-case allows the admin (**p_admin**) to view all available coupons in the system in a tabular format. It includes search functionality and the ability to delete coupons.

30.2. Actor

Admin managing products (User with the role **p_admin)**

30.3. Flow of Events

30.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The system sends a request to the backend API to fetch all coupons
- Step 3: The backend responds with the list of coupons, including fields such as ID, code, discount value.
- Step 4: The system displays the list of coupons and the actions (Edit, Delete)
- Step 5: The actor can use the search bar to find a coupon by any field.

30.3.2. Alternative Flows

- At step 3:
  - If no coupons exist, the system shows blank space.
- At step 4:
  - If the actor clicks **Delete** on an order:
    - The system immediately displays a confirmation dialog: “Are you sure you want to delete this coupon?”
    - If the actor **confirms**, the system sends a delete request to the backend to remove the coupon from the database.
    - If the actor **cancels**, the system closes the confirmation dialog without deleting the coupon.

30.4. Special Requirements

- Only users with role **p_admin** can view, delete or manage coupons

30.5. Preconditions

- The actor is logged in as **p_admin**.
- Coupon data exists in the backend.

30.6. Postconditions

- The system displays the list of coupons.
- Confirmation is displayed to the Admin.
- Deleted coupons are no longer shown.

30.7. Extension Points

30.7.1. Search

- **Location:** Step 5
- **Description:** Allows client-side searching by ID or code. The displayed coupon list is updated dynamically as the user types. If no results match, an empty list is shown.

31. Use-Case Specification: Create coupon (Admin Only)

31.1. Brief Description

The "**Create coupon**" use-case allows a **p_admin** to create a new coupon by filling out a form with coupon code and discount amount.

31.2. Actor

Admin managing products (User with the role **p_admin)**

31.3. Flow of Events

31.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role p_admin.
- Step 2: The actor clicks the "**Create**" button.
- Step 3: The system shows a form with two fields to fill out:
    - Coupon code
    - Discount amount
- Step 4: The actor fills out the form and submits.
- Step 5: The system validates all inputs.
- Step 6: The system sends a request to the backend API to create a new coupon in the database.
- Step 7: The system displays a success message and redirects the actor to view the list of coupons.

31.3.2. Alternative Flows

- At step 4:
    - If the actor cancels the confirmation, the system closes the confirmation dialog without creating the coupon.
- At step 5:
    - If the coupon code already exists, the system shows: "**Failed to create the coupon**".

31.4. Special Requirements

- Coupon code must be unique.
- Only users with role **p_admin** can create a coupon.

31.5. Preconditions

- The actor is logged in as **p_admin**.

31.6. Postconditions

- A new coupon is stored in the database and appears in the list.

31.7. Extension Points

31.7.1. Category Selection Change

- **Location:** Step 4

- **Description:** When the user selects a product category, the system dynamically loads and displays the corresponding specification input fields relevant to that category. This ensures that only applicable fields are shown, improving user experience and data accuracy.

31.7.2. Highlight Addition

- **Location**: Step 6
- **Description**: Users can add multiple highlight points dynamically by clicking an "Add Highlight" button, which appends new input fields without page reload.

32. Use-Case Specification: Edit coupon (Admin Only)

32.1. Brief Description

The "**Edit coupon**" use-case allows users with role **p_admin** to modify the details of an existing coupon.

32.2. Actor

Admin managing products (User with the role **p_admin)**

32.3. Flow of Events

32.3.1. Basic Flow

- Step 1: The actor logs in as a user with the role **p_admin**.
- Step 2: The actor clicks the "**Details**" of an existing coupon.
- Step 3: The system sends a request to the backend API to fetch and display the current coupon information including code and discount amount.
- Step 4: The actor updates the fields (e.g., code, discount amount).
- Step 5: The actor clicks the "**Save**" button.
- Step 6: The system shows a confirmation message and the actor confirms.
- Step 7: The system sends a request to the backend API to update the information of the current coupon.
- Step 8: The system shows a success message and redirects the actor to view the list of coupons.

32.3.2. Alternative Flows

- At step 5:
  - If the actor clicks the "**Cancel**" button, the system redirects the actor to view the list of coupons.
- At step 6:
  - If the actor cancels the confirmation, the system closes the confirmation dialog without updating the coupon.

32.4. Special Requirements

- Only users with role **p_admin** can edit a coupon.

32.5. Preconditions

- The actor is logged in as **p_admin**.
- The coupon exists in the database.

32.6. Postconditions

- The updated coupon information is saved and reflected in the list.

32.7. Extension Points

None