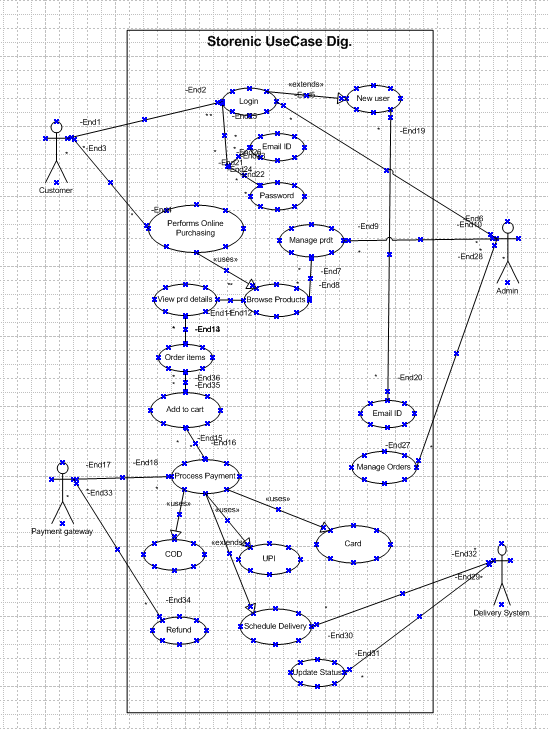
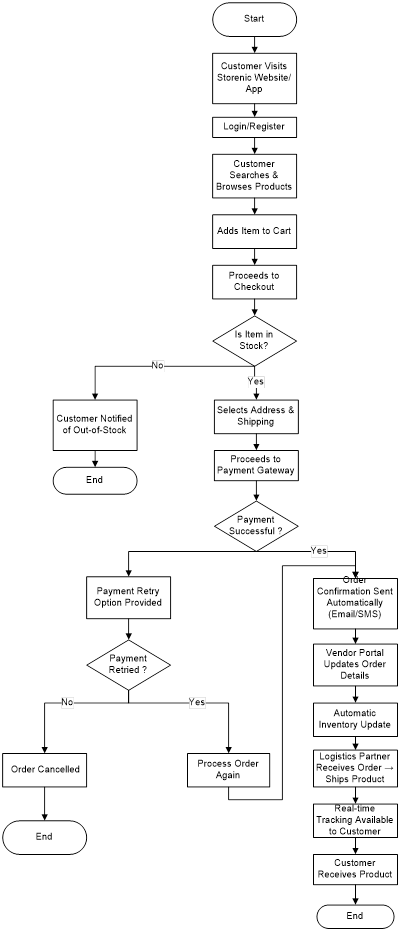
**Document 6- Please prepare a use case diagram, activity diagram and a use case specification document.**

**Use case Diagram -**



**Activity Diagram -**



### **1. Use Case Specification: Browse Products**

|  |  |
| --- | --- |
| **Brief Description** | **This use case describes how a customer browses products on the Storenic e-commerce platform.** |
| **Actors** | Customer |
| **Pre-Conditions** | - The customer has access to the internet. |
|  | - The customer is logged into the platform (optional for browsing). |
|  | - The platform's product catalog is loaded and accessible. |
| **Basic Flow** | 1. The use case begins when the customer opens the Storenic platform. |
|  | 2. The customer selects a product category or uses the search bar to find a specific product. |
|  | 3. The application displays a list of products matching the search or category. |
|  | 4. The customer can apply filters (e.g., price range, ratings, brand) to refine the results. |
|  | 5. The customer sorts the products (e.g., by price, popularity, or newest arrivals). |
|  | 6. The customer selects a product to view its details. |
| **Alternative Flows** | - **No Products Found**: If no products match the search or filters, the application displays a message: "No products found. Try a different search." |
|  | - **Slow Network**: If the network is slow, the application displays a loading spinner while fetching products. |
| **Key Scenarios** | - Customer applies multiple filters to find a specific product. |
|  | - Customer browses products without logging in. |
| **Post-Conditions** | - The customer views a list of products or a specific product's details. |
|  | - The application logs the customer's browsing activity for personalized recommendations. |
| **Special Requirements** | - The platform must support real-time updates to product availability and pricing. |
|  | - The platform must cache product data to improve loading speed. |

### **2. Use Case Specification: Add to Cart**

|  |  |
| --- | --- |
| **Brief Description** | **This use case describes how a customer adds products to their shopping cart.** |
| **Actors** | Customer |
| **Pre-Conditions** | - The customer has selected a product to view its details. |
|  | - The product is in stock and available for purchase. |
| **Basic Flow** | 1. The use case begins when the customer selects the "Add to Cart" button on a product page. |
|  | 2. The application validates the product's availability. |
|  | 3. The product is added to the customer's shopping cart. |
|  | 4. The application updates the cart total and displays a confirmation message: "Product added to cart." |
|  | 5. The customer can continue shopping or proceed to checkout. |
| **Alternative Flows** | - **Out of Stock**: If the product is out of stock, the application displays a message: "This product is currently unavailable." |
|  | - **Quantity Exceeded**: If the customer tries to add more than the available stock, the application displays a message: "Only X items available." |
| **Key Scenarios** | - Customer adds multiple products to the cart. |
|  | - Customer adds the same product multiple times, increasing the quantity in the cart. |
| **Post-Conditions** | - The product is successfully added to the customer's cart. |
|  | - The cart total is updated in real-time. |
| **Special Requirements** | - The platform must update the cart in real-time and reflect changes across all devices. |
|  | - The platform must prevent adding out-of-stock products to the cart. |

### **3. Use Case Specification: Checkout**

|  |  |
| --- | --- |
| **Brief Description** | **This use case describes how a customer completes the checkout process.** |
| **Actors** | Customer, Payment Gateway |
| **Pre-Conditions** | - The customer has items in their shopping cart. |
|  | - The customer is logged into their account. |
| **Basic Flow** | 1. The use case begins when the customer clicks the "Checkout" button. |
|  | 2. The application prompts the customer to enter or confirm shipping details. |
|  | 3. The customer selects a payment method (e.g., credit card, PayPal, wallet). |
|  | 4. The application redirects the customer to the Payment Gateway for payment processing. |
|  | 5. The Payment Gateway confirms the payment. |
|  | 6. The application displays an order confirmation page with a summary of the purchase. |
|  | 7. The customer receives an order confirmation email. |
| **Alternative Flows** | - **Payment Failed**: If the payment fails, the application displays a message: "Payment failed. Please try again." |
|  | - **Invalid Shipping Address**: If the shipping address is invalid, the application prompts the customer to correct it. |
| **Key Scenarios** | - Customer completes the checkout process successfully. |
|  | - Customer uses a saved payment method for faster checkout. |
| **Post-Conditions** | - The order is placed, and the customer receives an order confirmation email. |
|  | - The inventory is updated to reflect the purchased items. |
| **Special Requirements** | - The platform must support multiple payment methods and ensure secure transactions. |
|  | - The platform must validate shipping addresses before proceeding to payment. |

### **4. Use Case Specification: Track Order**

|  |  |
| --- | --- |
| **Brief Description** | **This use case describes how a customer tracks the status of their order.** |
| **Actors** | Customer, Delivery System |
| **Pre-Conditions** | - The customer has placed an order and received an order confirmation. |
|  | - The order has been processed and is ready for tracking. |
| **Basic Flow** | 1. The use case begins when the customer logs into their account. |
|  | 2. The customer navigates to the "Order History" section. |
|  | 3. The application displays a list of the customer's recent orders. |
|  | 4. The customer selects an order to view its status. |
|  | 5. The application retrieves the latest delivery status from the Delivery System. |
|  | 6. The application displays the order status (e.g., processing, shipped, delivered). |
|  | 7. The customer can view estimated delivery dates and tracking links. |
| **Alternative Flows** | - **Order Not Found**: If the order is not found, the application displays a message: "Order not found." |
|  | - **Delivery Delayed**: If the delivery is delayed, the application displays a message: "Your order is delayed. Please check back later." |
| **Key Scenarios** | - Customer tracks the delivery status of an order. |
|  | - Customer views detailed tracking information (e.g., courier details, live tracking). |
| **Post-Conditions** | - The customer views the current status of their order. |
|  | - The application logs the customer's tracking activity for analytics. |
| **Special Requirements** | - The platform must provide real-time updates from the Delivery System. |
|  | - The platform must support integration with multiple courier services. |

### **5. Use Case Specification: Manage Products (Admin)**

|  |  |
| --- | --- |
| **Brief Description** | **This use case describes how an admin manages products on the Storenic platform.** |
| **Actors** | Admin |
| **Pre-Conditions** | - The admin is logged into the admin dashboard. |
|  | - The admin has the necessary permissions to manage products. |
| **Basic Flow** | 1. The use case begins when the admin navigates to the "Manage Products" section. |
|  | 2. The admin selects an option to add, update, or delete a product. |
|  | 3. The application prompts the admin to enter product details (e.g., name, price, stock). |
|  | 4. The admin uploads product images and sets product categories. |
|  | 5. The admin submits the changes. |
|  | 6. The application updates the product catalog and displays a confirmation message. |
| **Alternative Flows** | - **Invalid Data**: If the admin enters invalid data, the application displays an error message: "Invalid data. Please check your inputs." |
|  | - **Duplicate Product**: If the admin tries to add a duplicate product, the application displays a message: "Product already exists." |
| **Key Scenarios** | - Admin updates the stock level of a product. |
|  | - Admin adds a new product to the catalog. |
| **Post-Conditions** | - The product catalog is updated with the new changes. |
|  | - The application sends a notification to the admin confirming the changes. |
| **Special Requirements** | - The platform must ensure that product updates are reflected in real-time across the system. |
|  | - The platform must support bulk product uploads via CSV files. |

**Assumptions**

* The platform will support multiple product categories, including electronics, fashion, home essentials, etc.
* **Storenic** will primarily serve B2C customers but may expand to B2B in future phases.
* Sellers and admins will be responsible for managing inventory and updating product details.
* Users will have internet access to use the platform.
* Payment gateways will comply with security standards.
* The platform will comply with relevant **eCommerce regulations, data privacy laws, and tax policies**.

**Constraints**

* The system must handle at least 20,000 concurrent users.
* Implementation should be completed within twelve months.

## **Dependencies**

* **Inventory System:** Ensures product availability before checkout.
* **Payment Gateway Integration:** Secure transactions through third-party payment processors.
* **User Authentication Service:** Validates user login and account access.
* **Shipping API:** Tracks orders and provides real-time delivery status.
* **Database Management System:** Stores and retrieves product, user, and order information.

## **Inputs and Outputs**

### **Inputs:**

* **User Inputs:** Login credentials, product selection, payment details, shipping address.
* **Admin Inputs:** Product details, discount codes, inventory updates.
* **Vendor Inputs:** Stock availability, order fulfillment status.
* **System Inputs:** Payment processing details, order tracking updates.

### **Outputs:**

* **User Outputs:** Order confirmation, invoice generation, shipping updates.
* **Admin Outputs:** Sales reports, inventory status, user activity logs.
* **Vendor Outputs:** Order requests, stock movement reports.
* **System Outputs:** Notification alerts, transaction logs, API responses.

## **Business Rules**

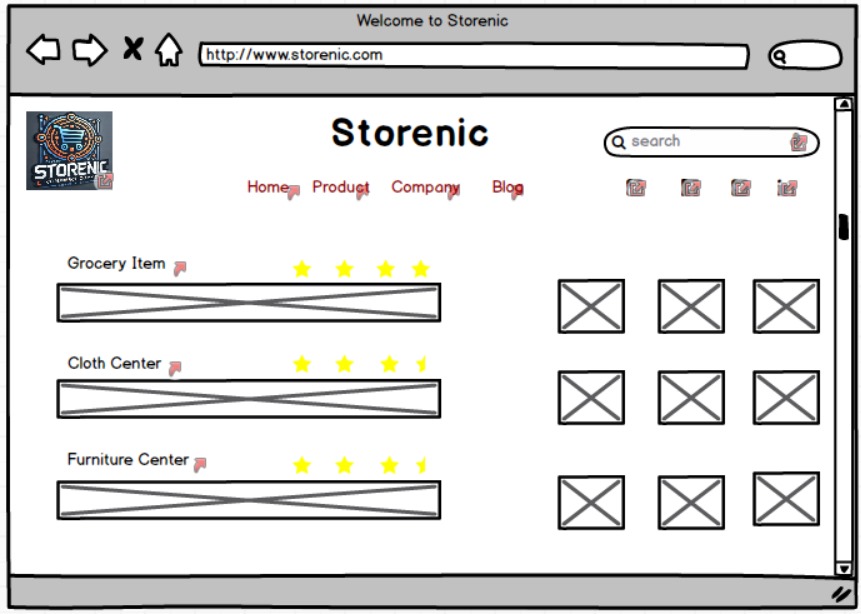
* **Authentication & Authorization:**
  + Users must have an active account to place orders.
  + Admins can modify product listings and user roles.
* **Cart & Checkout Rules:**
  + Items are reserved for 15 minutes after adding to the cart.
  + Coupons apply only if valid and within the expiry date.
* **Payment Processing:**
  + Orders are confirmed only after successful payment.
  + Refunds take 5-7 business days to process.
* **Stock Management:**
  + If an item is out of stock, users receive a notification.
  + Vendors must update stock levels regularly.
* **Order Fulfillment:**
  + Users receive tracking details after the order is dispatched.
  + Returns are allowed within 7 days of delivery.

**Miscellaneous Information**

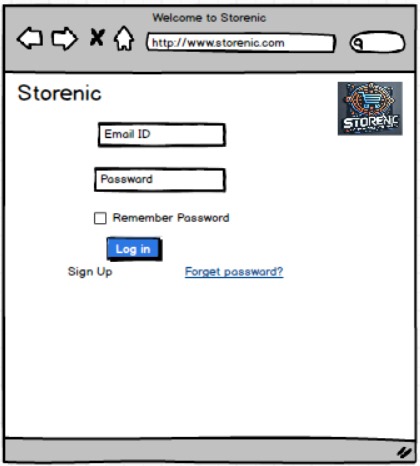
* **Performance Metrics:** The checkout process must complete within 10 seconds under normal conditions.
* **Scalability:** The platform can handle peak traffic of 20,000 concurrent users.

**Document 7- Screens and pages**

1. **Home page of Storenic**

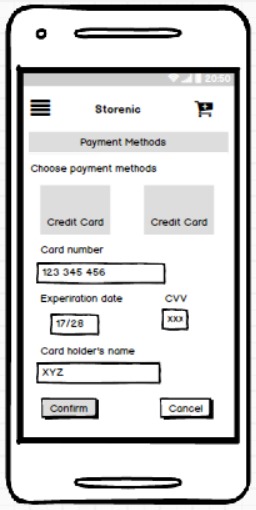


1. **Login Page**





1. **Payment Page**



1. Add to Cart Page



**Document 8- Tools-Visio and Axure**

I used Visio and Axure for the project. Visio helped me draw diagrams and flowcharts. It was easy to put shapes and lines together to show how things work. Axure let me make prototypes of the app. I created screens and buttons that looked real. Both tools were simple to use and made my work easier.

If you're working on a project, tools like Visio and Axure can help a lot. They make it easy to share ideas and plans with others.

**Document 9- BA experience**

**My experience as BA in following phases:**

1. **Requirement gathering:**
2. To gather requirements, we used MOSCOW technique.
3. Client is not available for some period of time during this phase. So as a BA I need to source out point of contacts from his side and get the information asap.
4. I validate the requirements using FURPS technique
5. There are many requirements which are duplicated or repeated. We need to remove them immediately
6. Prototyping is used to give more specific requirements
7. **Requirement Analysis:**
8. We need to draw UML diagrams to visually describe the requirements
9. Activity diagrams also used to describe the process flow
10. Communicate the diagrams to team. Some team members might not agree with them and might make changes. As a BA we need to consider the points and make modifications
11. Prepare BRD and FRS
12. **Design:**
13. From the use case diagrams.
14. Communicate with client on design and solution documents
15. Do not miss a single test case. It might have huge impact on project development in later stages
16. Prepare test data for testing
17. Update RTM. This is must as we need to make sure that all the requirements are met
18. **Development:**
19. Organized JAD sessions
20. Clarifying queries of tech team during coding
21. There might be some team members who doesn't agree with the concept or who doesn’t cooperate during JAD sessions. As a BA I handle the situation gently and had one on one discussion with them. Explained how their actions are going to affect the project. Setup healthy environment within the team.
22. Referred diagrams to code the Unit
23. Conduct regular meetings with technical team and client which is challenging. Some team members might not be available for the meeting. Recording the session and providing that to missed one and having one to one discussion later with that missed person is all I need to do
24. **Testing:**
25. Prepare test cases from use cases
26. Perform high level testing
27. Test data is requested by BA from client
28. Updated RTM
29. Take signoff from client
30. Prepare client for UAT
31. **Deployment:**
32. Forwarded RTM to client which should be attached to project closure document
33. Coordinates to complete and share end user manuals
34. Plans and organizes training sessions
35. Make sure all the candidates attend the meeting