# Shopping Application - Microservices Architecture

## 1. Overview of Microservices

This document explains the business and technical flow of a microservices-based shopping application. The application is composed of several services that interact to handle user requests, manage products, process orders, and facilitate seamless checkout and payment processes.

Microservices involved:   
1. Product-Service - Manages product catalog and inventory.   
2. Cart-Service - Handles cart operations, storing product selections.   
3. Order-Service - Manages order creation, checkout, and payment processing.   
4. User-Service - Handles user authentication and profile management.   
5. Notification-Service - Sends email confirmations and notifications.

## 2. Business Flow (User Journey)

The business flow outlines how users interact with the application and how different services collaborate to fulfill user actions.

1. User Browses Products (Product-Service):  
 - The user views available products by fetching data from the product-service.  
2. Adding Items to Cart (Cart-Service):  
 - Users add products to their cart. Cart-service validates product availability by calling product-service.  
3. Checkout Process (Order-Service & User-Service):  
 - Users proceed to checkout. User-service authenticates users, and order-service calculates the order total.  
4. Order Confirmation (Order-Service & Product-Service):  
 - Order-service places the order and updates product inventory. Cart-service clears the user's cart.  
5. Payment and Notification:  
 - Payments are processed by external APIs, and notifications are sent by notification-service.

## 3. Technical Flow (Service Communication)

In the technical flow, services communicate through REST APIs, gRPC, and asynchronous message brokers such as Kafka or RabbitMQ. The architecture involves an API Gateway that routes client requests to appropriate microservices.

Technical Components:  
- API Gateway – Routes client requests to backend services.  
- Service Discovery – Ensures services can locate each other (e.g., Eureka).  
- Circuit Breaker – Prevents cascading failures (e.g., Hystrix, Resilience4j).  
- Message Broker – Facilitates asynchronous communication.

## 4. Service Interactions and API Examples

1. Product-Service:  
 - Endpoint: GET /products/{id}  
 - Fetches product details for display or validation.  
2. Cart-Service:  
 - Endpoint: POST /cart  
 - Adds products to the user's cart.  
3. Order-Service:  
 - Endpoint: POST /checkout  
 - Places orders by validating stock and processing payments.  
4. User-Service:  
 - Endpoint: GET /user/{id}  
 - Fetches user profile and authenticates users.

## 5. Diagram Explanation

The flow diagram represents the interaction between the services:  
- The frontend sends requests to the API Gateway, which forwards them to respective microservices.  
- Product browsing, cart operations, and checkout are handled in sequence by respective services.  
- Upon order placement, the product inventory is updated, and the cart is cleared.  
- Notification-service sends a confirmation email to the user.