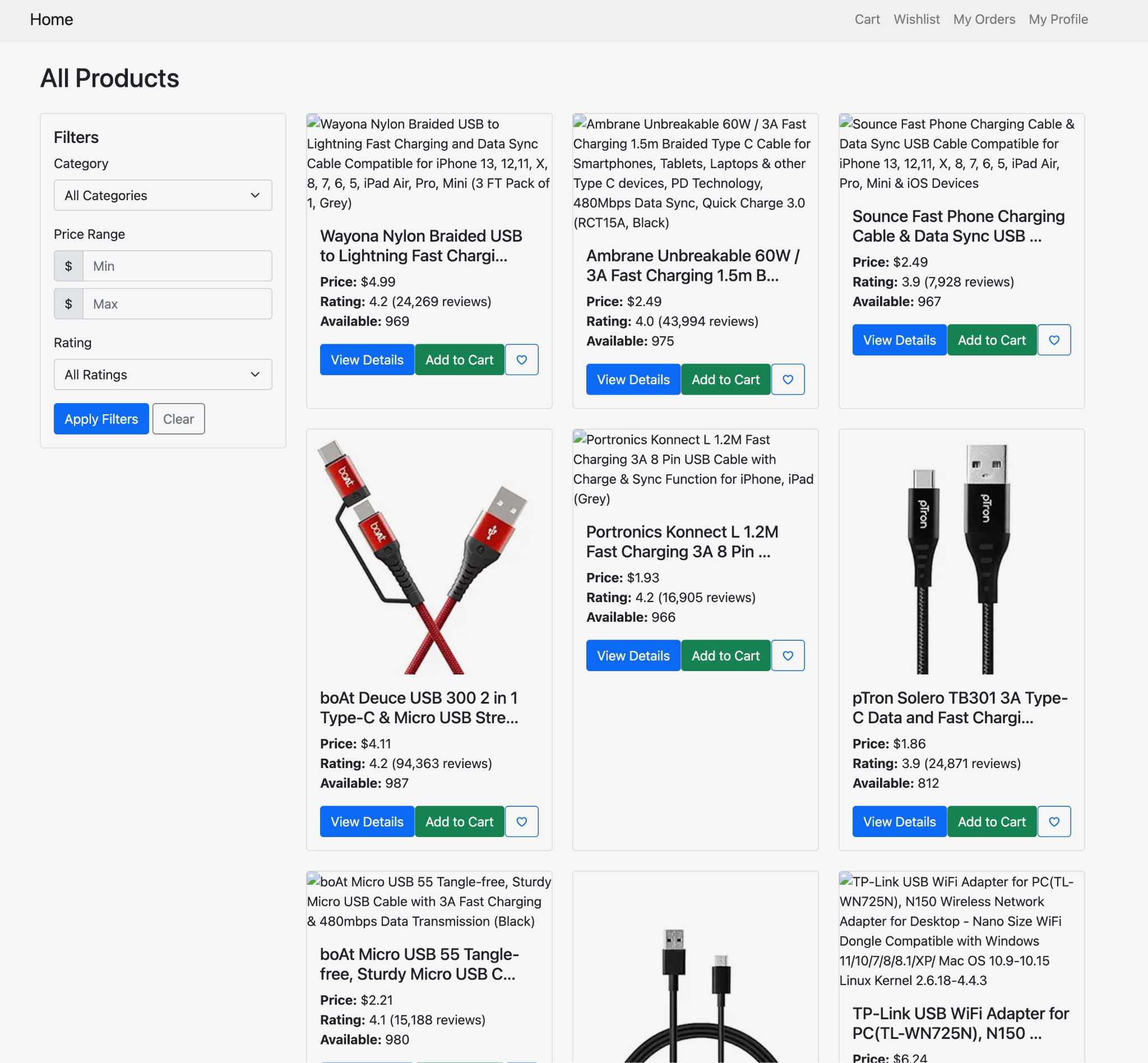
**End-to-End Solution Design and Implementation**

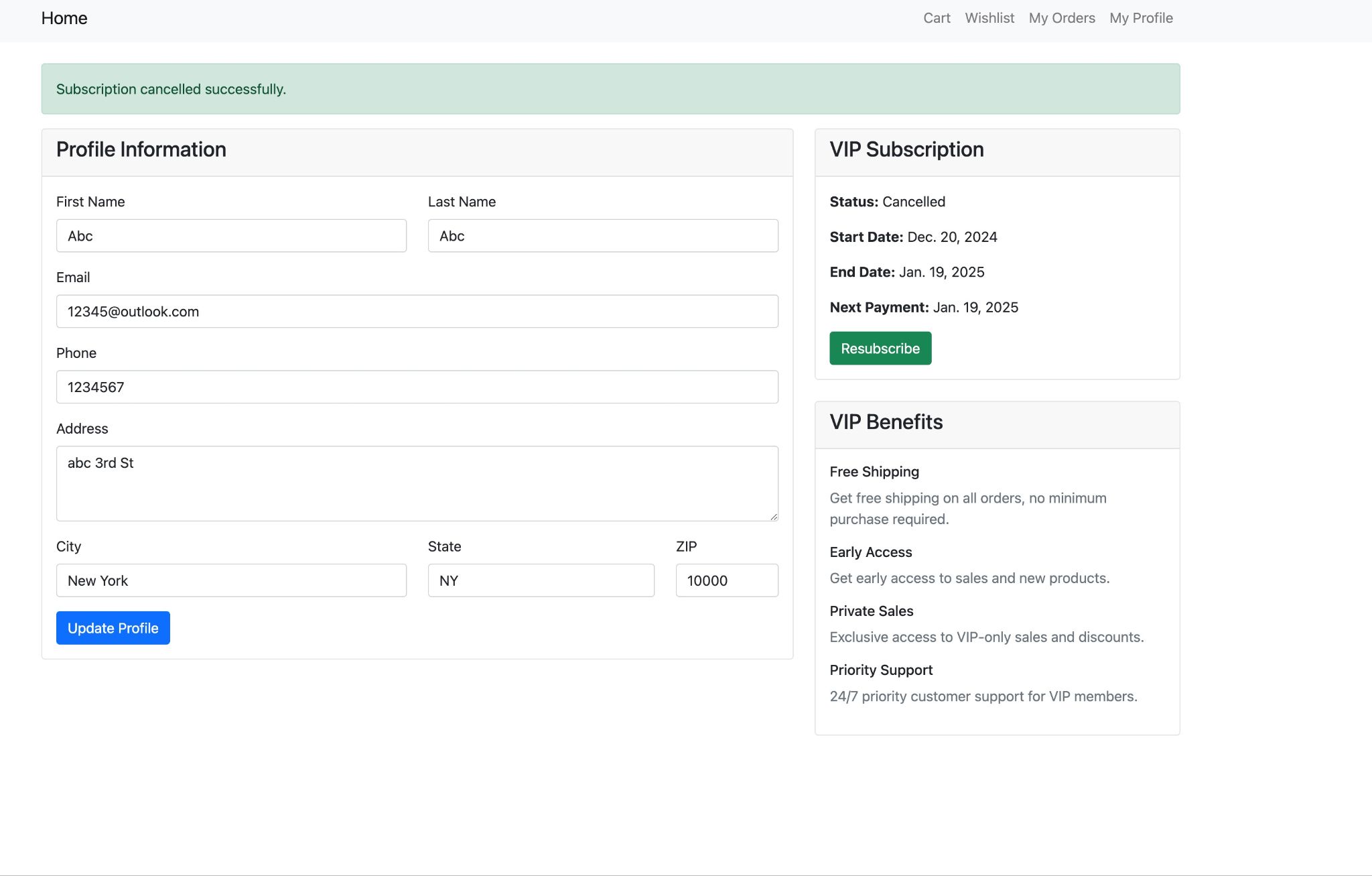
*Github Link:* [*https://github.com/Joannepistulli/Database-Systems-Group-Project*](https://github.com/Joannepistulli/Database-Systems-Group-Project) **Front End General Overview:**

****

1. Cases Supporting Customer Design Decisions

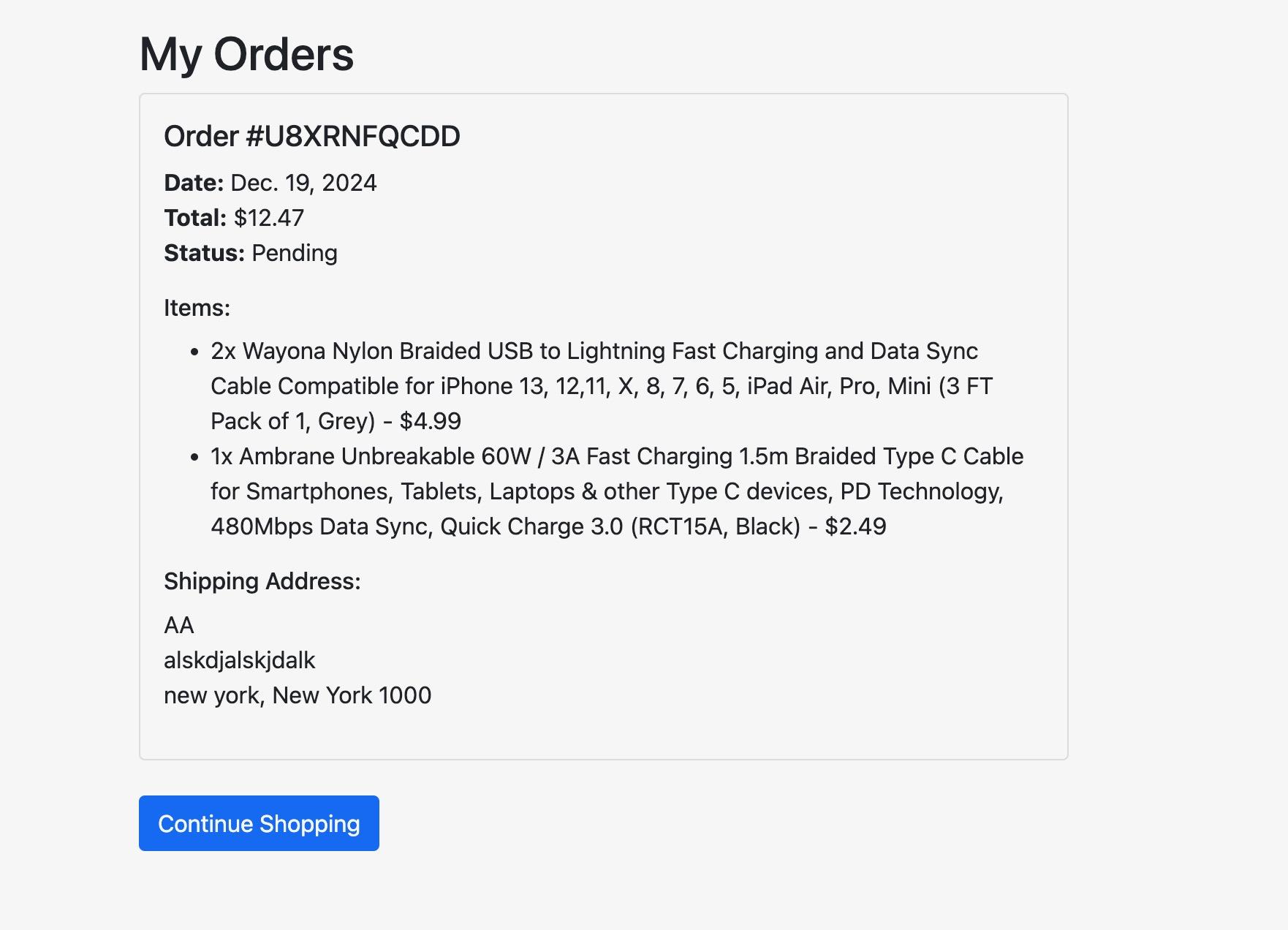
The customer entity is central to Amazon’s EDA as it stores critical information about each user, such as contact details and identification. This data is crucial for personalized marketing, tracking user behaviors, and analyzing customer satisfaction.

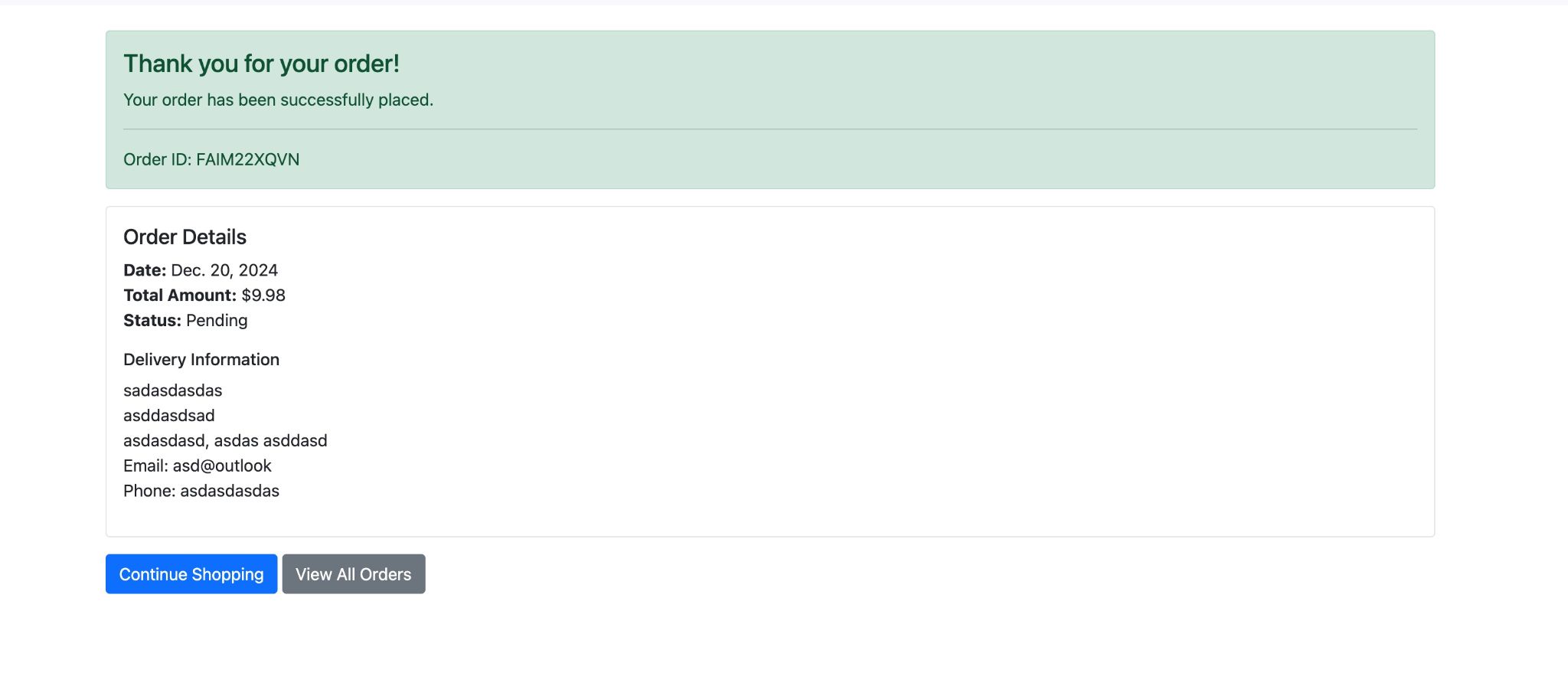
The design decision of the following customer attributes (ID, Name, Email, Address, and Phone) were selected to support real-time insights into customer interactions and purchasing behaviors. This design enables Amazon to develop targeted marketing, quickly respond to customer support needs, and enable account and order recovery in case of security incidents.



2. Cases Supporting Order Design Decisions  
 The order entity represents transactions and are essential for tracking Amazon's revenue generation and operational flow. The entity’s design supports real-time order processing, customer order tracking, and inventory management.

The design decision of the following order attributes (ID, Date, Amount) capture necessary transaction details that allow Amazon to track order fulfillment in almost real-time, analyze purchasing trends which includes peak times and product demand, manage inventory by tracking the relationship between orders and items

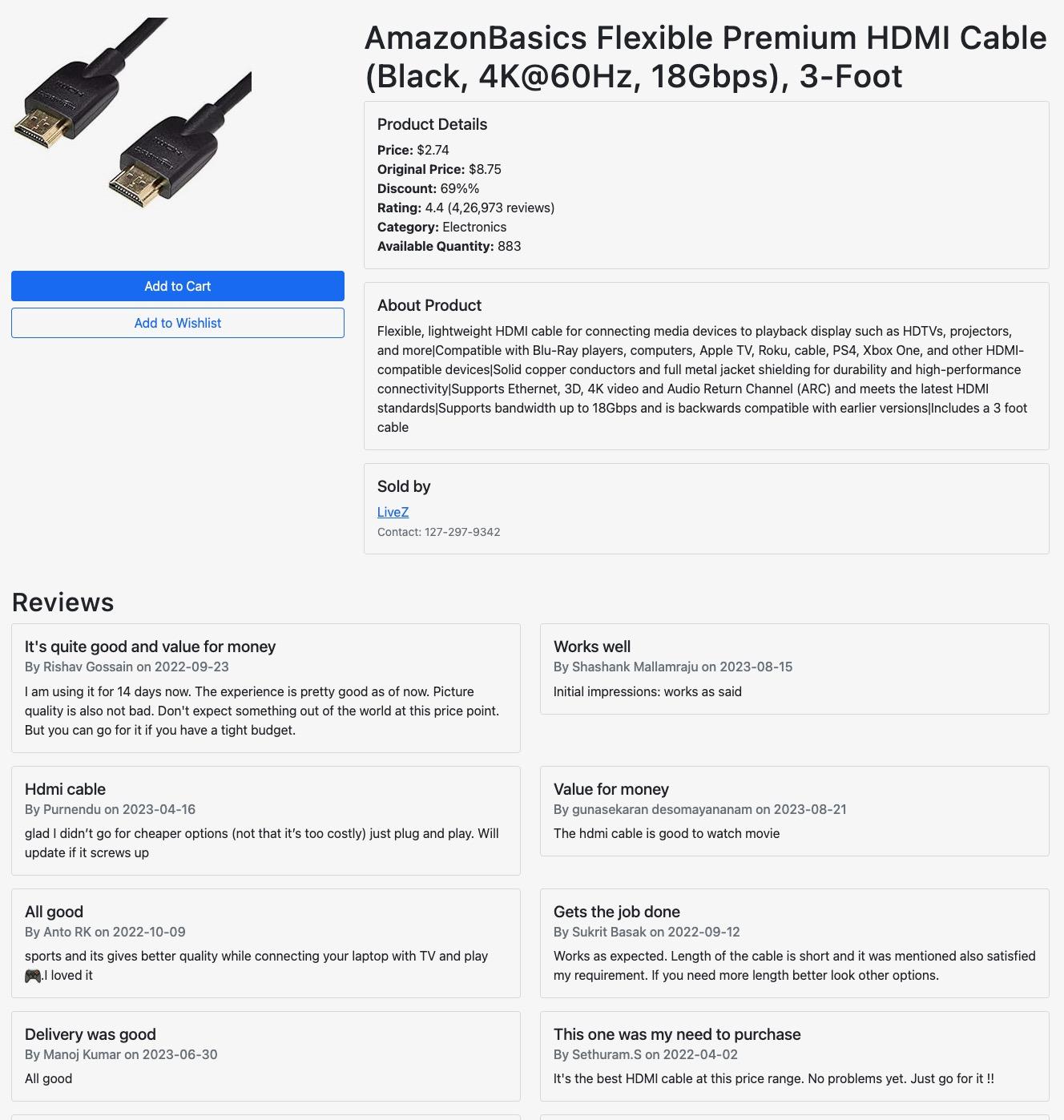




3. Cases Supporting Review Design Decisions

The review entity represents customer reviews, which are critical for Amazon’s recommendation system and help other customers make informed purchasing decisions. Reviews also provide unstructured data, which, when integrated with structured data, enriches Amazon’s insights.

The design decision of the following review attributes (ID, Text, Rating, and Date) provides comprehensive details on customer feedback, enabling Amazon to sentiment analysis on products and services, assess product quality and vendor performance, automated flagging of negative feedback for further investigation, supporting data governance and quality.

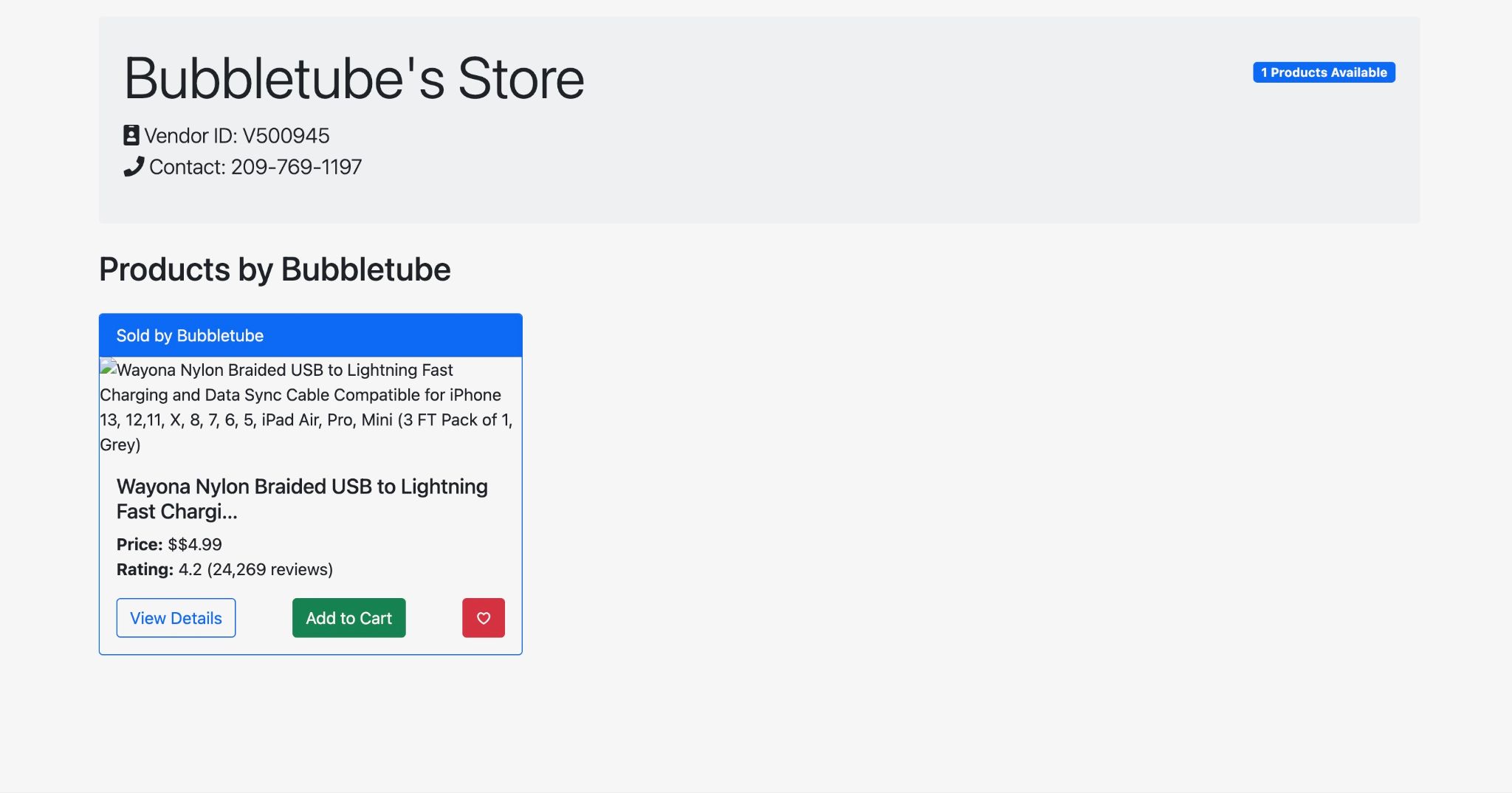


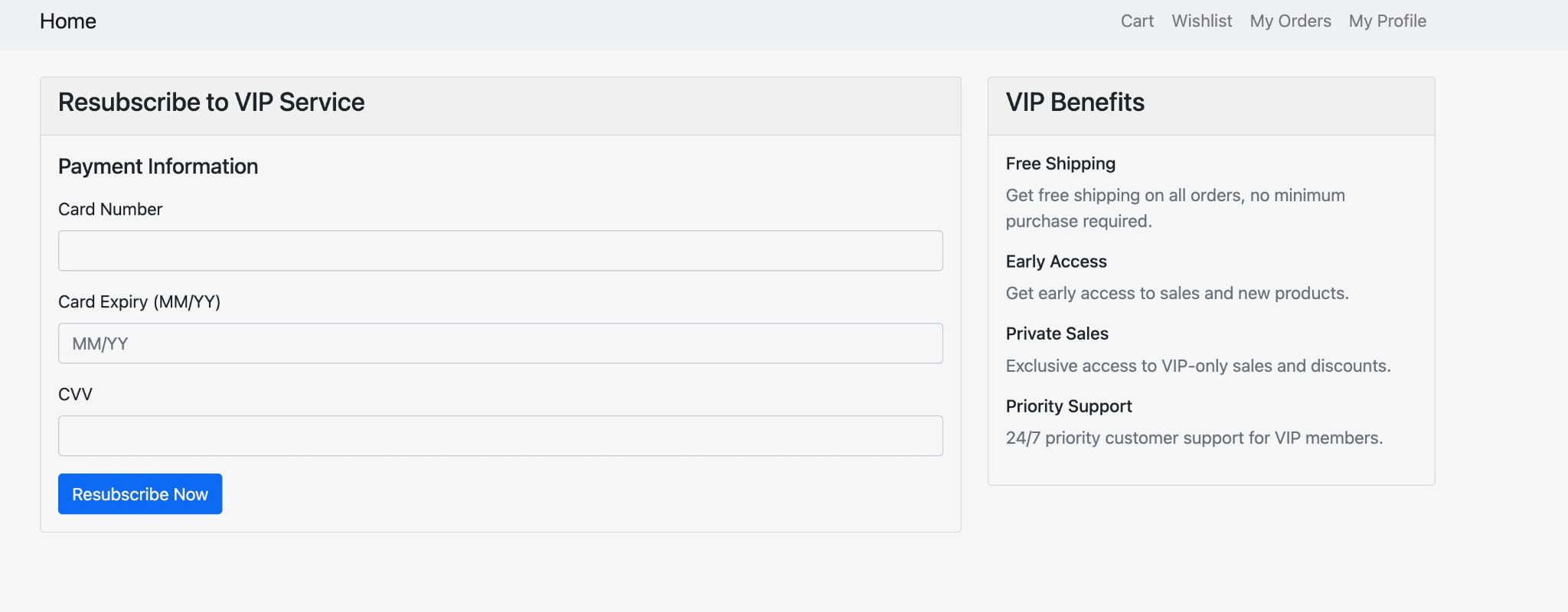
4. Cases Supporting Vendor Design Decisions

The vendor entity represents the vendors that supply the products and services which Amazon sells, and having a well-defined Vendor entity supports supplier management, product sourcing, and supply chain analysis.

The design decision of the following vendor attributes (ID, Name, Contact) allows Amazon to assess vendor reliability and performance by tracking product returns and customer feedback, maintain a contact database to streamline communication with suppliers,

track vendor data for compliance and quality assurance purposes, supporting the data governance framework.

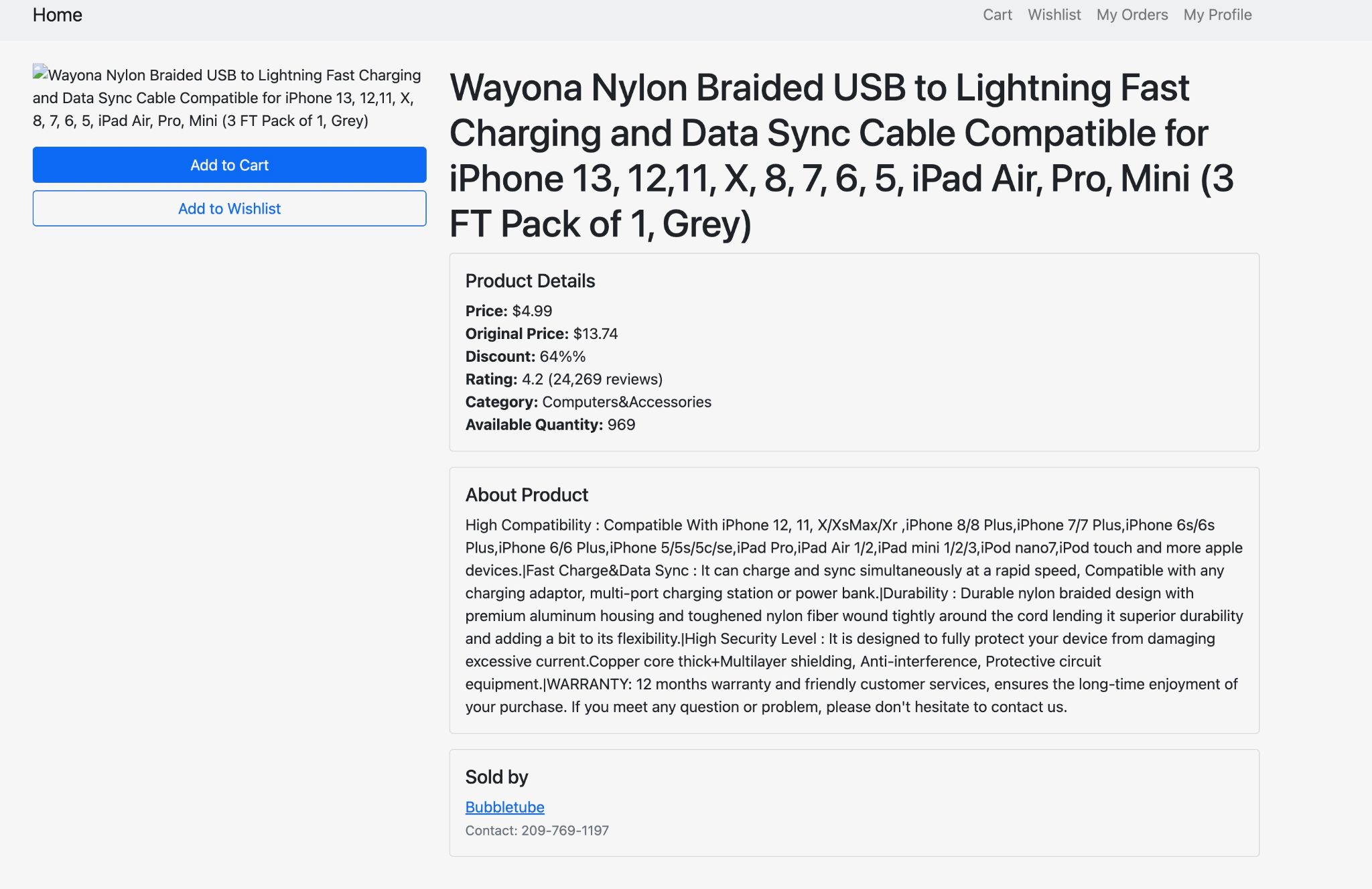




5. Cases Supporting Item Design Decisions

The item entity represents products available on Amazon. This entity is central to inventory management, customer purchase behavior analysis, and product recommendations.

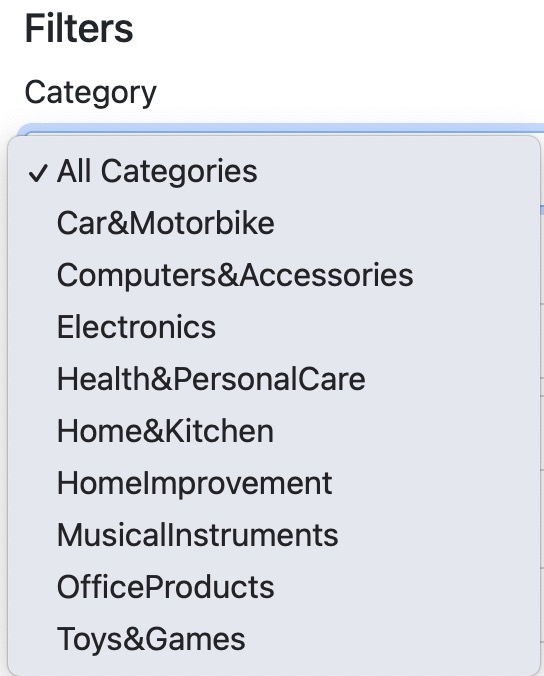
The design decision of the following item attributes (ID, Name, Description, Price), is designed to facilitate seamless product tracking and categorization for easy searching and recommendations, allow integration with reviews and orders, supporting comprehensive analytics on product performance and customer satisfaction, aid in real-time stock updates and inventory tracking through relationships with stock and category entities.

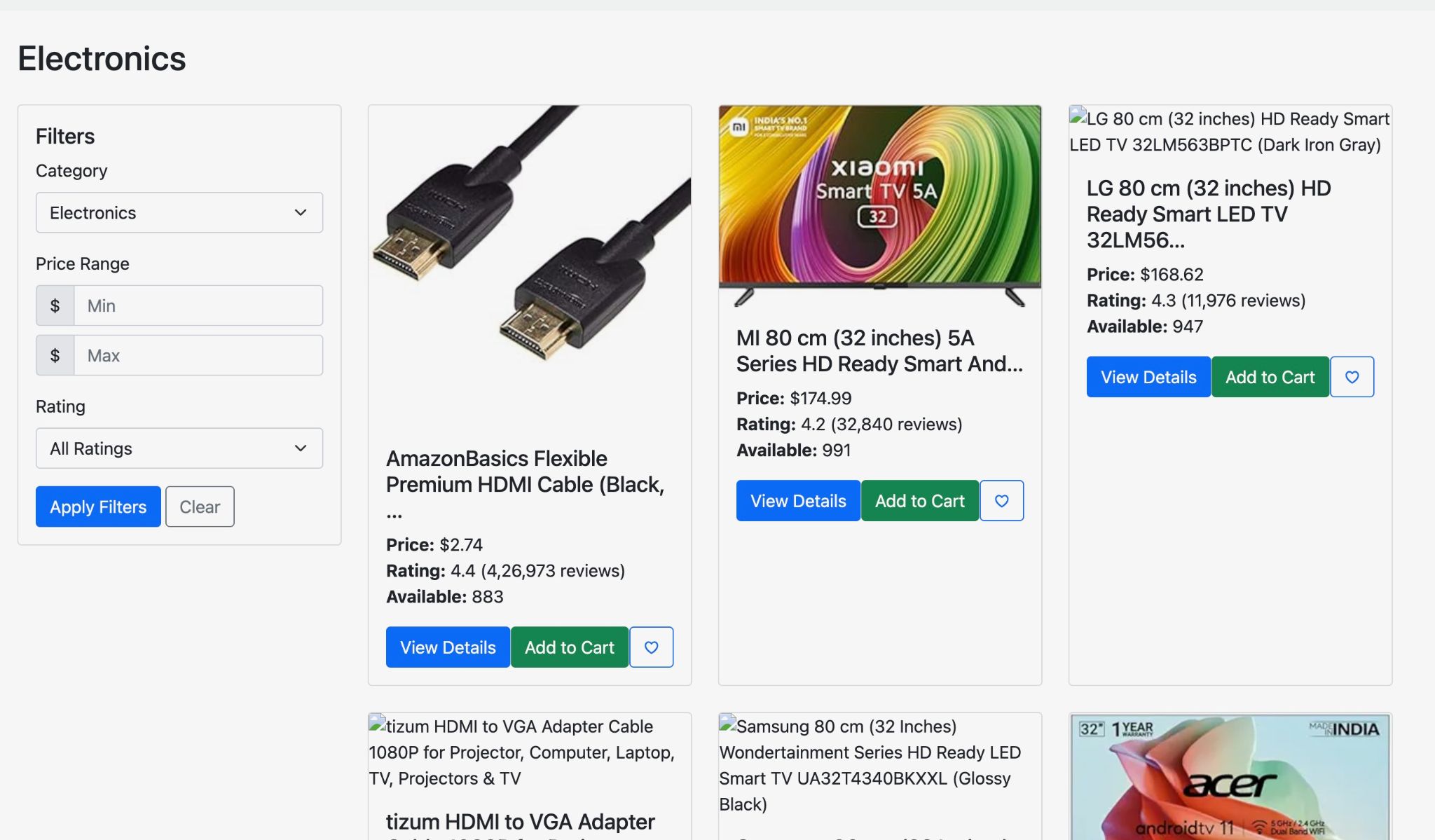


6. Cases Supporting Category Design Decisions

The category entity represents the categories found on Amazon when browsing for specific items. Categories allow Amazon to group items, making it easier to organize the vast array of products available on the platform. Categorization also aids in improving the customer browsing experience and supports recommendations.

The design decision of the following category attributes (ID, Name) helps Amazon to implement effective product browsing and filtering options, conduct category-based sales analysis, identify high-demand product groups, streamline inventory management by enabling category-specific insights into demand and stock levels.

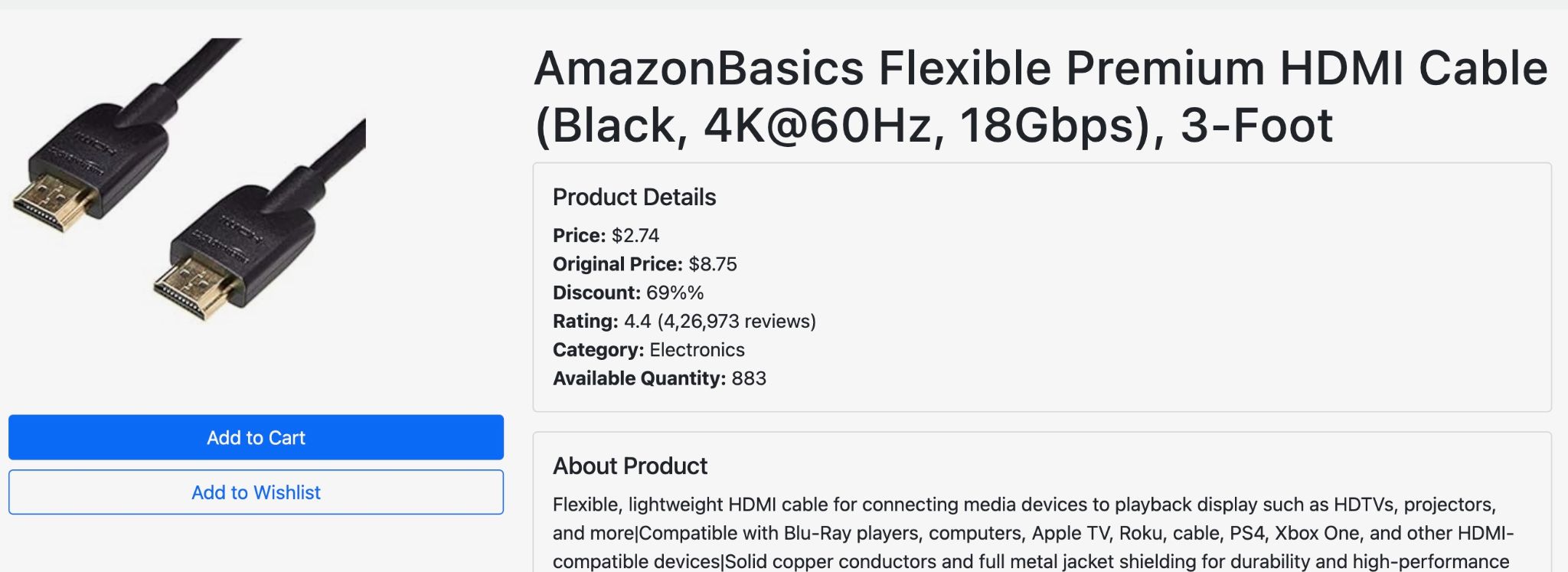




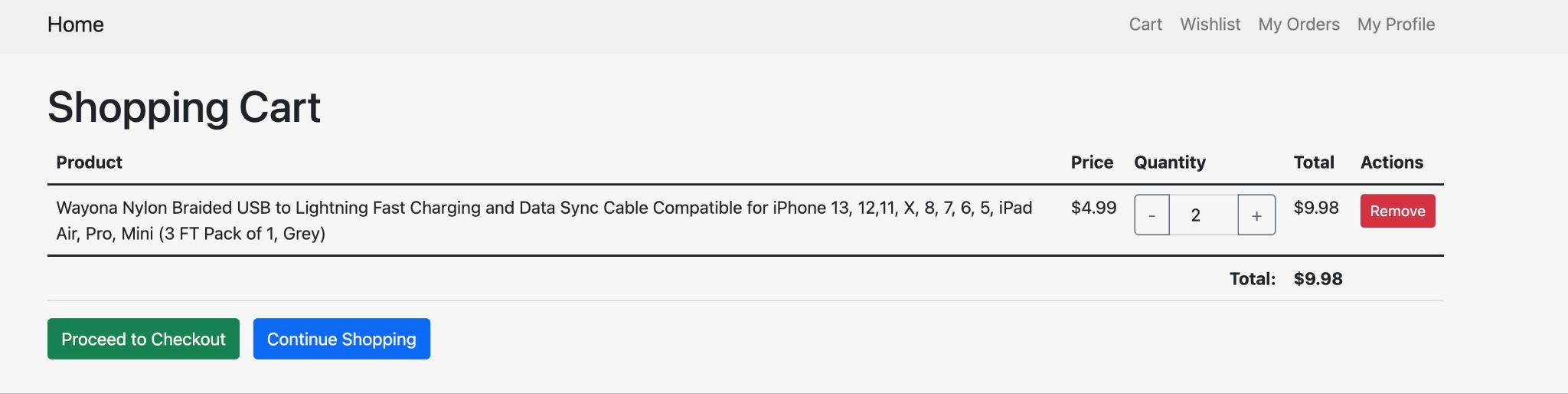
7. Cases Supporting Stock Design Decisions

The stock entity represents Amazon’s inventory levels and storage locations, essential for ensuring that products are available to fulfill customer orders in a timely manner.

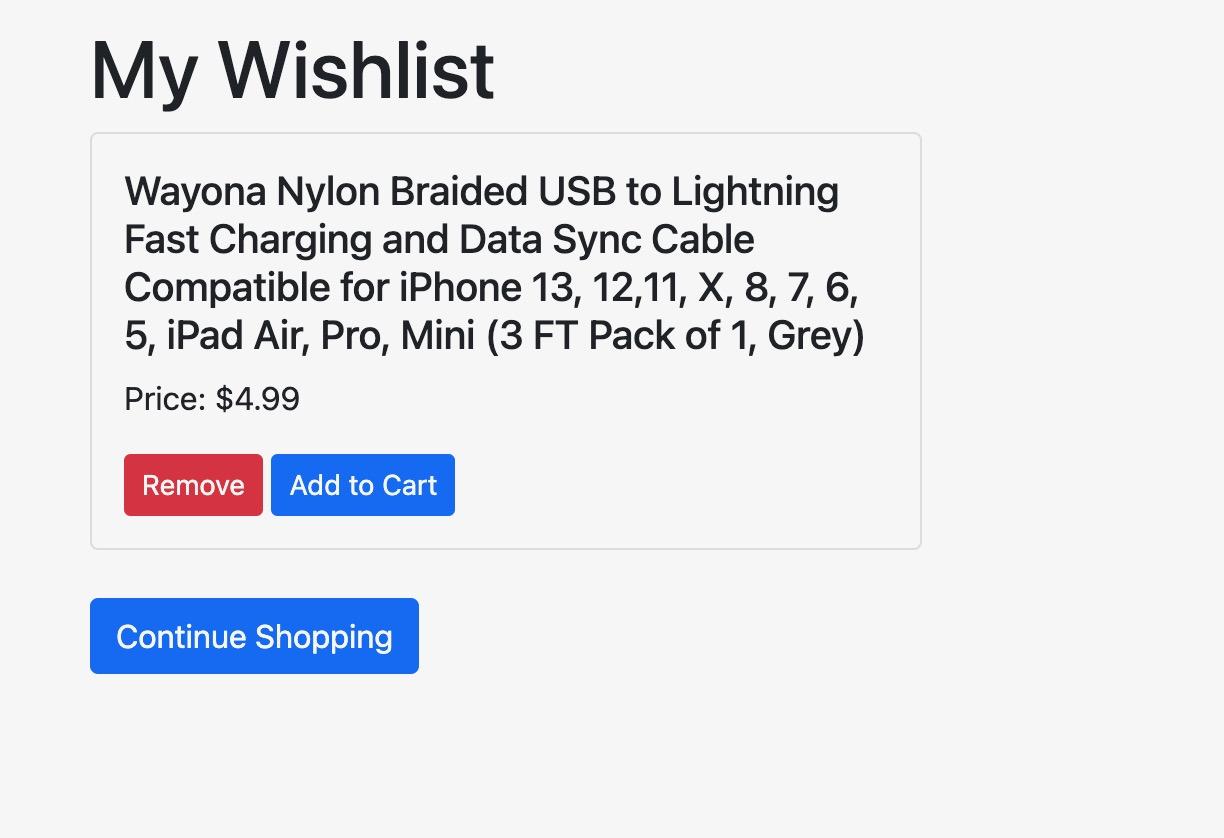
The design decision of the following stock attributes (ID, Quantity, Location) to enable Amazon to monitor real-time inventory levels across locations, essential for fast delivery and minimizing stockouts, support warehouse optimization by analyzing item locations and storage capacity, facilitate replenishment alerts based on quantity thresholds, thereby enhancing operational efficiency.



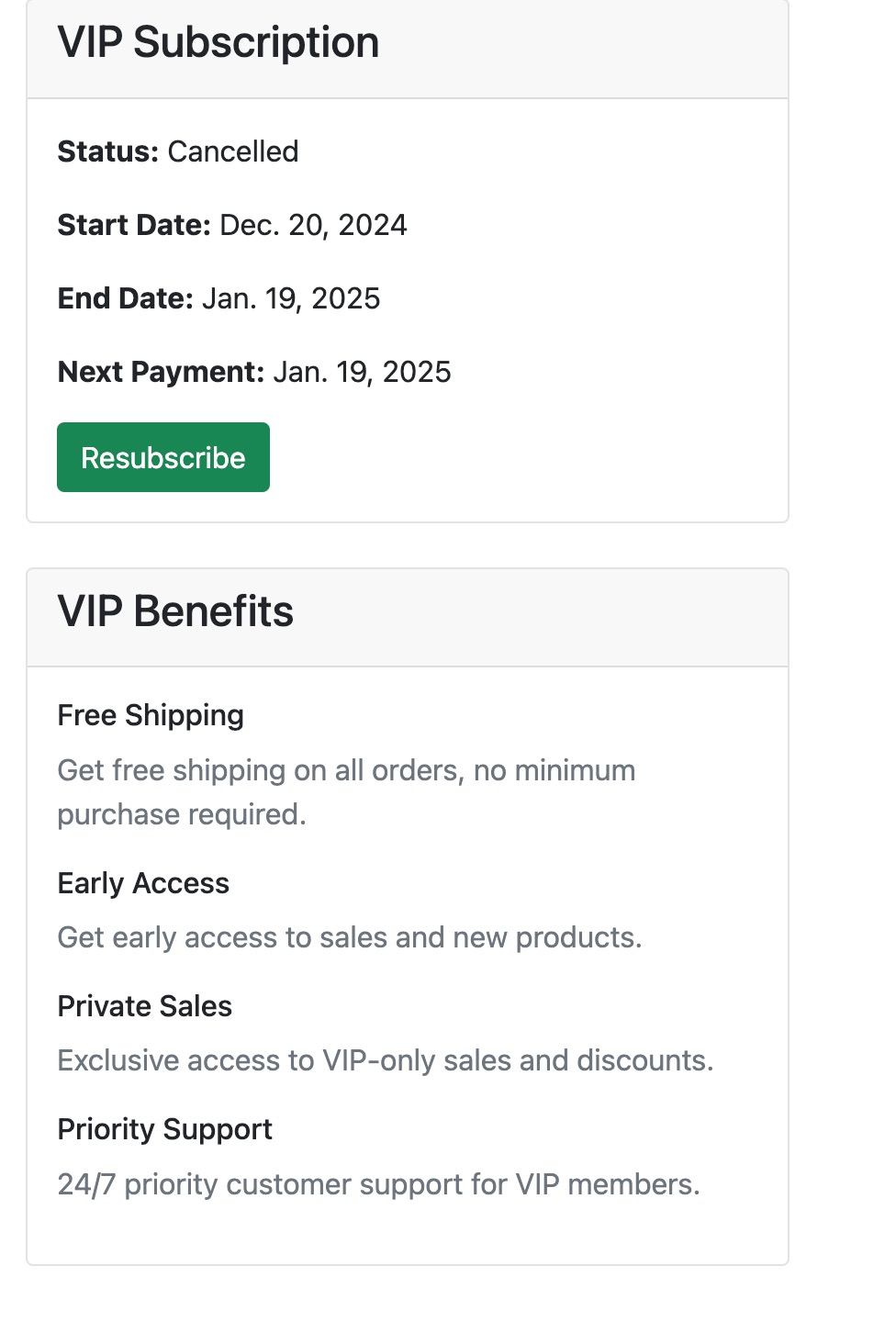
8. Cases Supporting ShoppingCart Design Decisions  
 The shopping cart entity holds information on items customers intend to purchase, supporting an intuitive and dynamic shopping experience. This entity is key for analyzing customer intent and improving sales conversions.  
 The design decision of the following shopping cart attributes (ID and CreatedDate) helps Amazon analyze the time customers spend considering purchases, abandoned cart trends, and supports targeted remarketing efforts, improving the shopping experience and potentially increasing conversion rates.

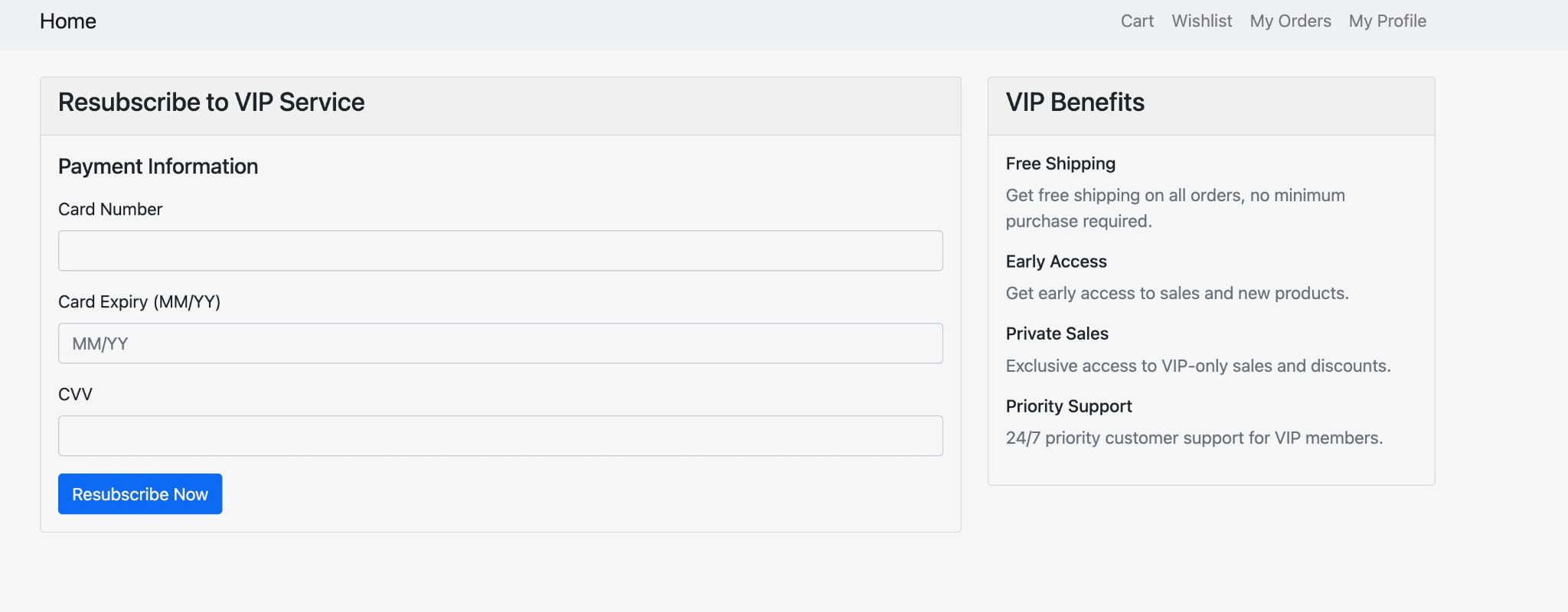


9. Cases Supporting Wishlist Design Decisions  
 The wishlist entity represents items customers save for potential future purchases. It plays a role in enhancing customer engagement and insights into customer preferences.  
 The design decision of the following wishlist attributes (ID and CreatedDate) supports tracking customer preferences over time, allowing for personalized recommendations, and helping Amazon identify trending products, which can inform stock and promotion strategies.

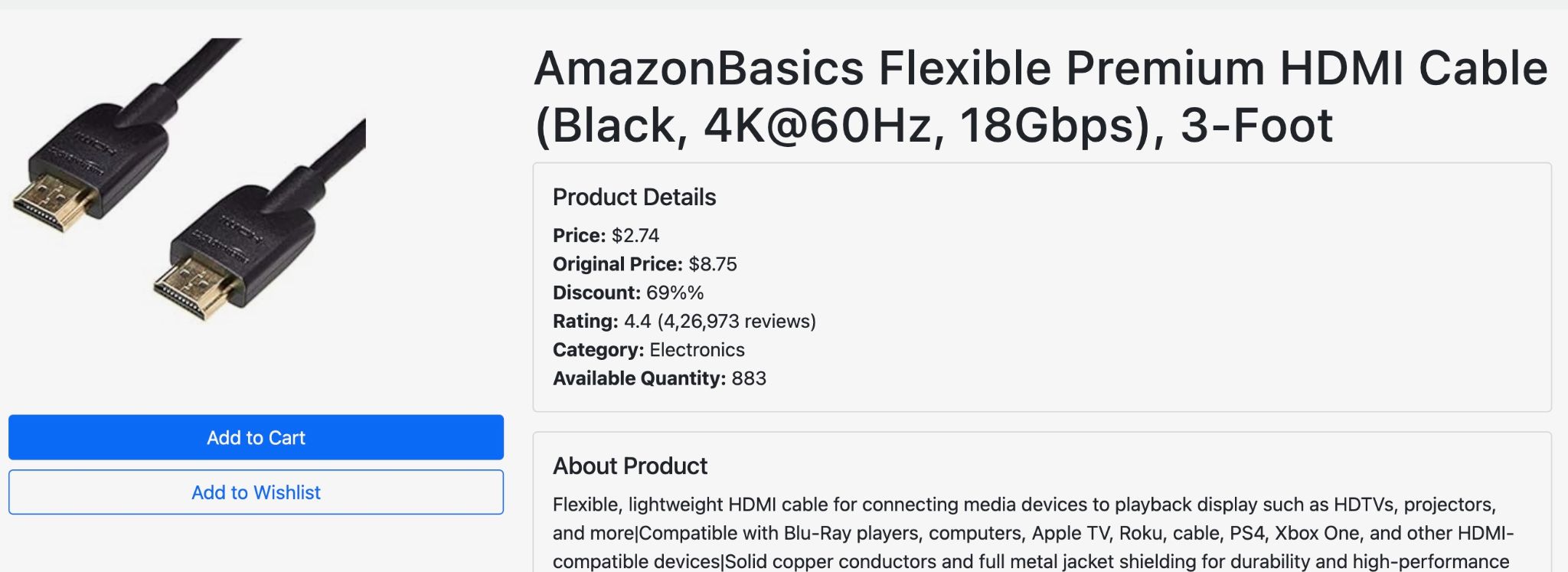


10. Cases Supporting Subscription Design Decisions  
 The subscription entity tracks subscription details for customers who enroll in programs such as Amazon Prime. It is essential for analyzing subscription-based revenue and loyalty.  
 The design decision of the following subscription attributes (ID, Type, StartDate, and EndDate) supports tracking subscription lifecycle stages, enabling renewal reminders, and enhancing customer retention by understanding subscription trends, aiding in targeted marketing for subscription upgrades.





11. Cases Supporting Promotion Design Decisions  
 The promotion entity tracks promotional offers available to customers, which play a vital role in marketing and driving sales. Promotions encourage purchases and attract new customers to Amazon.  
 The design decision of the following promotion attributes (ID, Name, Discount, StartDate, and EndDate) helps Amazon monitor promotion effectiveness, track customer response to discounts, and optimize promotional timing and duration, supporting sales growth and customer acquisition strategies.



**Back End General Overview:**

**Data Model Design**

In our project, we implemented a sophisticated database management system utilizing Django's ORM framework. The core architecture revolves around a well-structured relational database schema that efficiently handles product data, vendor information, and user interactions. The system's foundation is built on optimized data models that ensure quick access and reliable data storage while maintaining scalability.

**Data Loading and Optimization**

The data loading process leverages pandas for efficient CSV integration, incorporating smart deduplication mechanisms to maintain data integrity. Our optimization strategy includes implementing a caching system that significantly reduces I/O operations, coupled with pagination that displays 12 items per page to manage large datasets effectively. These implementations have notably improved system response times and reduced server load.

Our database schema incorporates interconnected models that handle various aspects of the e-commerce platform. The shopping features include comprehensive cart management, wishlist functionality, and order tracking systems. The review system is designed with careful consideration of user-product relationships, implementing proper indexing and date-based ordering for optimal data retrieval and display.

Performance optimization was a key focus during development. We implemented strategic database indexing with carefully selected primary keys and properly indexed foreign key relationships. Query efficiency was achieved through optimized filtering mechanisms and minimized database hits. The system maintains data integrity through structured migration management and robust transaction handling, ensuring consistent and reliable operations even during high-load scenarios. The project's data relationships are designed to support complex queries while maintaining performance. We established clear connections between vendors and products, enabling efficient filtering and search capabilities. The user interaction system is integrated seamlessly with the product database, allowing for smooth operation of features like reviews, ratings, and purchase history.

**Product Data:**

class Product(models.Model):

product\_id = models.CharField(*max\_length*=20, *primary\_key*=True)

vendor = models.ForeignKey(

Vendor,

*on\_delete*=models.CASCADE,

*related\_name*='products',

*default*='DEFAULT'

)

product\_name = models.CharField(*max\_length*=500)

category = models.CharField(*max\_length*=200)

discounted\_price = models.DecimalField(*max\_digits*=10, *decimal\_places*=2)

actual\_price = models.DecimalField(*max\_digits*=10, *decimal\_places*=2)

discount\_percentage = models.IntegerField()

rating = models.DecimalField(*max\_digits*=3, *decimal\_places*=1)

rating\_count = models.CharField(*max\_length*=20)

about\_product = models.TextField()

img\_link = models.URLField(*max\_length*=500)

product\_link = models.URLField(*max\_length*=500)

stock = models.IntegerField(*default*=100)

**Vendor Management:**

**class Vendor(models.Model):**

**vendor\_id = models.CharField(*max\_length*=20, *primary\_key*=True)**

**vendor\_name = models.CharField(*max\_length*=200)**

**vendor\_contact = models.CharField(*max\_length*=20)**

**Review System:**

**class Review(models.Model):**

**review\_id = models.CharField(*max\_length*=20, *primary\_key*=True)**

**product = models.ForeignKey(Product, *on\_delete*=models.CASCADE, *related\_name*='reviews')**

**user\_id = models.CharField(*max\_length*=50)**

**user\_name = models.CharField(*max\_length*=100)**

**review\_title = models.CharField(*max\_length*=200)**

**review\_content = models.TextField()**

**review\_date = models.DateField()**

**Chatbot General Overview:**

*A multi source integration which combines structured data from a database with natural language capabilities of OpenAI’s GPT-3.5-Turbo model. This balances database lookups with AI generated answers to provide comprehensive support*

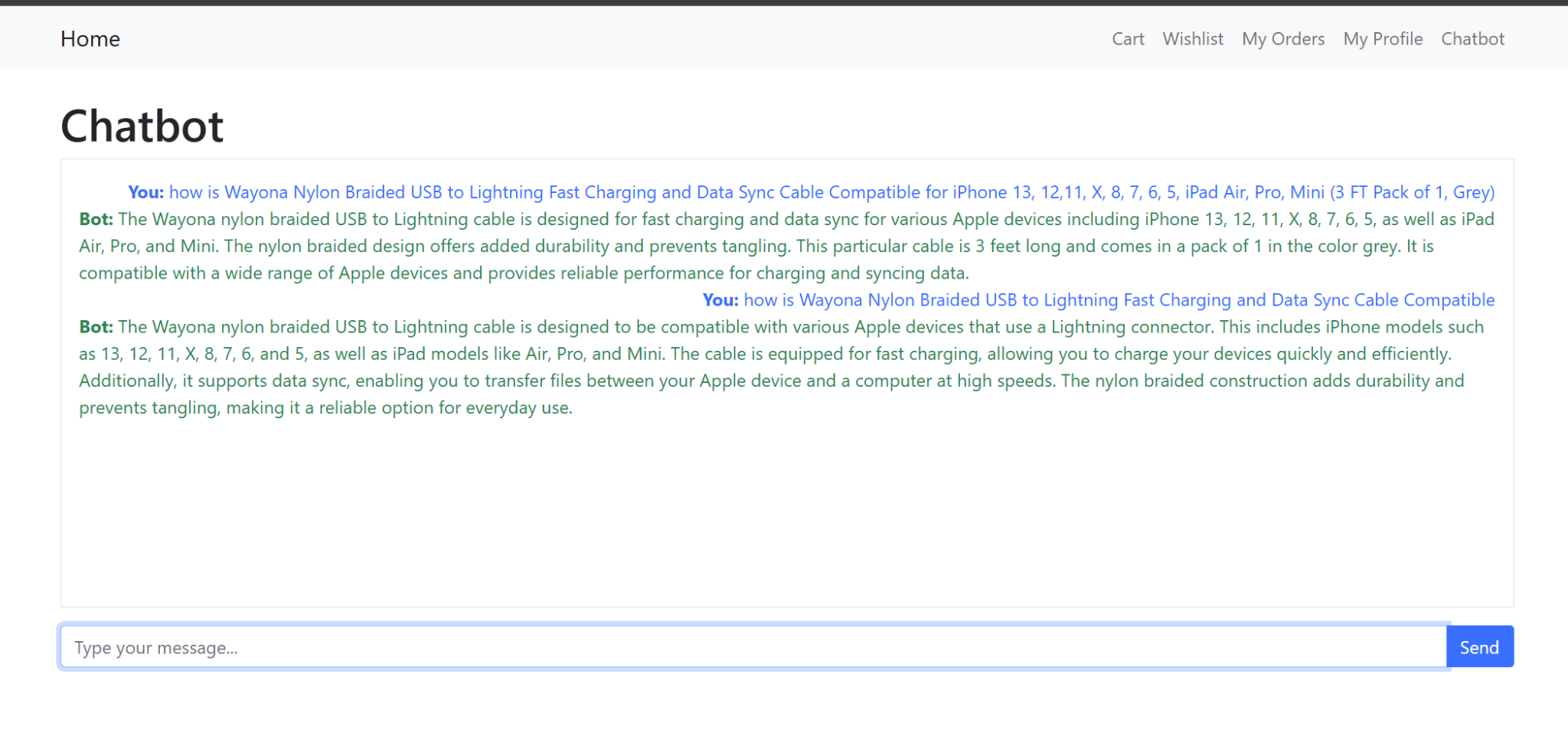
**Main Features**

1. Handles Product Reviews & Ratings

If the user asks for reviews or if the message contains keywords like "good," "review," or "评价," the chatbot tries to provide review-based responses for a specific product.

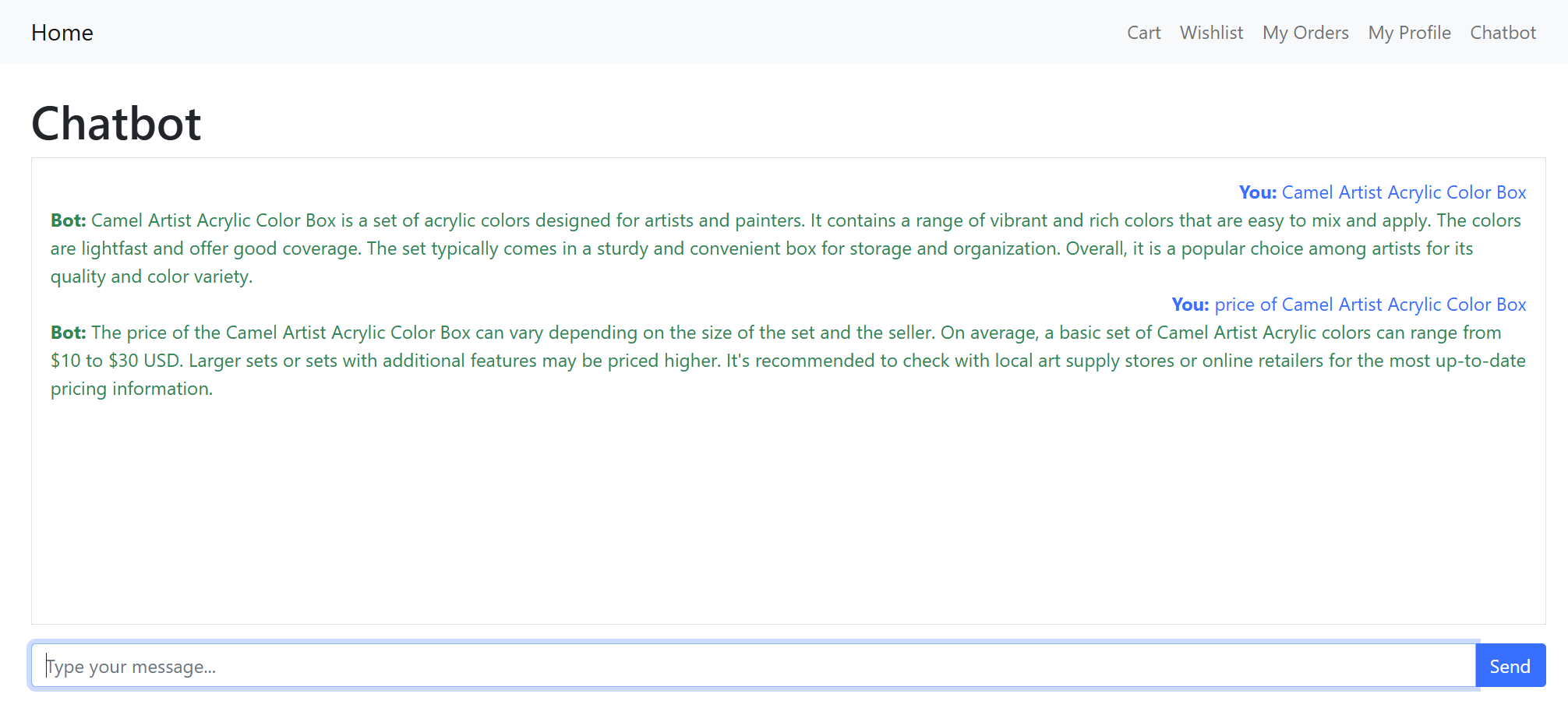
This process extracts the product name from the user’s input, and uses *rapidfuzz* for fuzzy matching between the user's input and product names in the database.

If a match is found then it Retrieves the reviews, average rating for the product, and summarizes the reviews using OpenAI's GPT-3.5-Turbo model (if reviews exist). However if there are no reviews, it informs the user that there are no reviews available or if there is no match, it falls back to OpenAI’s API to generate a general response.

****

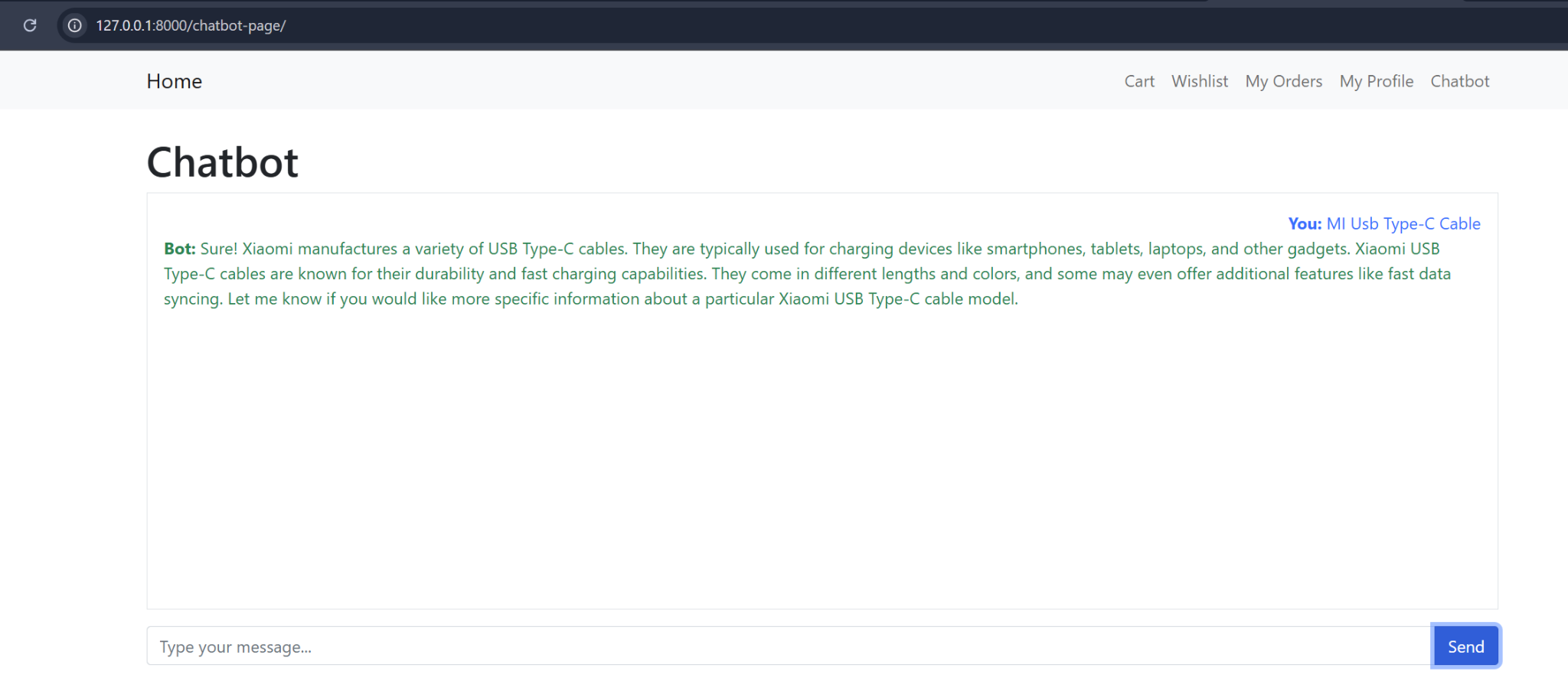
1. General Product Inquiries  
   If the user mentions "product" or "商品" (product in Chinese), the chatbot provides information about available products.

This process fetches up to five products from the database and constructs a list of product details which entails the product name, discounted price, stock availability, etc. If no products are available then it will inform the user.

****

1. Fall back to OpenAI for general Inquiries  
   For all other types of queries (e.g., questions unrelated to products or reviews), the chatbot forwards the conversation history to OpenAI's GPT-3.5-Turbo model.

This process sends the user’s query along with the full conversation history to the OpenAI API and receives responses from OpenAI

****

1. Conversation Management

Maintains conversation history where it can store user queries, chatbot responses, limits the conversation history ro the most recent 20 messages to reduce payload size, and ensures context continuity for OpenAI's model to generate contextually appropriate responses.

1. Error Handling

If an error occurs during processing (e.g., database query failure or API issues), the chatbot will automatically fall back to OpenAI's API and return a polite error-handling response to the user, ensuring the chatbot remains functional and user-friendly.