Design Document: Flipkart Web Scraper

1. Project Objectives

- Extract product details, prices, ratings, and reviews from Flipkart.
- Analyze market trends and consumer behavior.

2. Features

- **Product Categories:**
  - Smartphones
  - Laptops
  - Electronics
  - Fashion (Men's, Women's)
  - Home Appliances
- **Data Points:**
  - Product Name
  - Price
  - Rating
  - Number of Reviews
  - Product Description (optional)
  - Image URL (optional)
  - Product URL

3. Data Structure

The extracted data will be stored in a Python list of dictionaries, with each dictionary representing a product.

Python

```
products = [
    {
        'product_name': 'Product A',
        'price': 19999,
        'rating': 4.5,
        'reviews': 1234,
        'description': 'Product description',
        'image_url': 'https://example.com/image.jpg',
        'product_url': 'https://www.flipkart.com/product-a'
    },
    # ... other products
]
```

4. Scraping Process Flowchart

1. **Start**
2. **Import necessary libraries:** `requests`, `BeautifulSoup`, `fake_useragent`
3. **Define product categories and data points**
4. **Create empty lists for storing scraped data**
5. **Iterate through product categories:**
    - Construct search URL for the category
    - Make a request to the URL with appropriate headers
    - Parse HTML content using BeautifulSoup
    - Extract product details and append to data lists
    - Handle errors and exceptions
6. **Create a DataFrame from scraped data**
7. **Save data to CSV or other format**
8. **End**

5. Error Handling and Data Validation

- **HTTP Error Handling:** Check for status codes (e.g., 404, 500) and retry failed requests.
- **Parsing Errors:** Handle exceptions that occur during HTML parsing.
- **Data Validation:** Ensure extracted data is in the correct format (e.g., price is a number, rating is a float).
- **Data Cleaning:** Remove unnecessary characters or whitespace from extracted data.

6. Libraries

- **requests:** For making HTTP requests.
- **BeautifulSoup:** For parsing HTML content.
- **fake_useragent:** For generating random user-agent headers.
- **pandas:** For creating DataFrames and saving data to CSV.

7. Additional Considerations

- **Anti-Scraping Measures:** Be aware of Flipkart's anti-scraping measures and implement countermeasures (e.g., delays, random user-agents).
- **Dynamic Content:** If Flipkart uses JavaScript to load product data, consider using Selenium or other tools to handle dynamic content.
- **Scalability:** For large-scale scraping, explore using asynchronous programming or distributed systems.
- **Data Storage:** Decide on the appropriate data storage format (CSV, JSON, database) based on project requirements.