# Chapter 6
# Angular RxJS for E-commerce Development


**Swarom Padalkar, Maheshwar D. Jaybhaye, and Sangita M. Jaybhaye**

**Abstract** This paper explores the use of RxJS in combination with Angular for dynamic filtering and sorting in e-commerce websites. It highlights the importance of filtering and sorting in e-commerce and the challenges faced in implementing effective filtering and sorting systems. The paper then explains how RxJS operators such as map, filter, and scan can be used to create responsive and scalable filtering and sorting systems. Additionally, it discusses the advantages of using Angular for e-commerce websites and how it facilitates the implementation of dynamic filtering and sorting systems. Code examples and case studies are provided to demonstrate the effectiveness of using RxJS and Angular for dynamic filtering and sorting in e-commerce.

## 6.1 Introduction

Angular and RxJS play a significant role in the development of e-commerce applications, offering powerful tools and capabilities for building robust and responsive web applications. Angular, a widely adopted framework for web development, provides a comprehensive set of features and tools for creating scalable and maintainable applications. RxJS, on the other hand, is a reactive programming library that enhances Angular applications by enabling efficient handling of asynchronous events and managing complex data flows.

Hevery and Erickson [1] aforementioned, Angular provides a structured and modular approach to application development, making it suitable for building large-scale e-commerce platforms. It offers a rich set of features, such as component-based architecture, dependency injection, and a powerful templating system, which facilitate the development of dynamic and interactive user interfaces.

S. Padalkar (✉) · M. D. Jaybhaye
COEP Technological University, Pune, India
e-mail: padalkarsj19.prod@coeptech.ac.in

S. M. Jaybhaye
Vishwakarma Institute of Technology, Pune, India

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2024

P. H. V. S. Talpa Sai et al. (eds.), *Intelligent Manufacturing and Energy Sustainability*,
Smart Innovation, Systems and Technologies 372,
https://doi.org/10.1007/978-981-99-6774-2_6

Kaluža [2], compared Angular with other popular frameworks, including React, to evaluate their suitability for web application development. The study highlights Angular's robustness, maintainability, and extensive ecosystem, making it an excellent choice for complex e-commerce projects.

Bulgakova [3] explored the effectiveness of reactive programming, of which RxJS is a key component, in web development. The study showcases the benefits of reactive programming in handling asynchronous operations, managing state, and composing complex data streams, all of which are crucial aspects of e-commerce application development.

By leveraging the capabilities of Angular and RxJS, e-commerce developers can build feature-rich and responsive applications that provide seamless user experiences, efficient data handling, and real-time updates. These technologies enable the creation of dynamic product catalogs, interactive shopping carts, real-time notifications, and smooth checkout processes.

The study focuses on the use of RxJS for filtering and sorting in e-commerce development, providing practical insights and recommendations for optimizing website performance and enhancing the user experience.

The study also explores the broader applications of reactive programming in web development and the potential for future research in this area.

The study aims to contribute to the field of e-commerce development and provide practical guidance for businesses looking to improve their website functionality and user experience.

### *6.1.1 Angular for E-commerce Development*

Angular is a front-end framework that enables developers to create dynamic, responsive, and intuitive user interfaces. With its built-in components and modules, Angular offers a flexible and scalable framework that is ideal for e-commerce development.

Some of the key features of Angular for e-commerce development include:

1. Dynamic User Interfaces: Angular allows developers to create dynamic and responsive user interfaces that adapt to user input and changes in data.
2. Powerful Shopping Cart Features: Angular provides powerful shopping cart features that enable developers to manage and update shopping carts in real-time.
3. Robust Security Features: Angular provides a range of security features that help protect sensitive user data, including built-in support for HTTPS, CSRF protection, and content security policies [4, 5].

### 6.1.2 RxJS for E-commerce Development

RxJS is a reactive programming library that enables developers to create complex data streams and manipulate data in real-time. With RxJS, developers can create observable data streams that can be easily managed and transformed. This is particularly useful in e-commerce development, where real-time data updates and dynamic user interfaces are critical. Some of the key features of RxJS for e-commerce development include:

1. Asynchronous Data Management: RxJS enables developers to manage asynchronous data streams and manipulate data in real-time.
2. Reactive Forms: RxJS provides support for reactive forms, which can respond to user input in real-time and provide an interactive user experience.
3. Real-Time Data Updates: RxJS enables real-time data updates, which can be particularly useful in e-commerce development for managing product catalogues, shopping carts, and other critical aspects of the user experience [6].

### 6.1.3 Significance of Study

E-commerce websites are a critical component of modern business, and optimizing their performance and user experience is crucial for success.

Filtering and sorting functionality is a key feature of e-commerce websites, but traditional approaches can be inefficient and lead to poor user experience.

RxJS offers a reactive programming approach that can handle complex data streams in real-time, making it an ideal solution for dynamic filtering and sorting in e-commerce websites.

By studying the use of RxJS for filtering and sorting in e-commerce development, we can identify best practices and strategies for optimizing performance and enhancing the user experience.

This study can contribute to the broader field of reactive programming and its application in web development, as well as provide practical insights and recommendations for e-commerce businesses looking to improve their website performance and user experience. Some of companies which uses angular for its e-commerce websites are-

1. Google Shopping: Google Shopping is an e-commerce platform developed by Google. The website is developed using Angular and offers online shopping services for various products.
2. Nike: Nike is a well-known sports brand that sells its products online. The website is developed using Angular and provides customers with a range of products to choose from.
3. iStock: iStock is a popular website that provides users with a wide range of stock photos, vectors, illustrations, and videos. The website is developed using Angular and is known for its easy-to-use interface and fast search results.

## 6.2 Angular and Its Features for E-commerce Development

Angular is a widely used front-end framework for building web applications, including e-commerce websites. It offers a range of features that make it an ideal choice for e-commerce development. In this section, we will explore some of the key features of Angular and their benefits for e-commerce development.

Angular is a TypeScript-based open-source front-end web application platform that was developed by Google. It is a complete rewrite of AngularJS, which is also a front-end framework. Angular uses a component-based architecture, which makes it easier to develop, test, and maintain complex web applications. It is designed to work well with other libraries and tools, such as RxJS, which makes it ideal for e-commerce development [1, 2, 4].

### *6.2.1 Angular Modules and Components*

Angular modules are used to organize related functionality into cohesive blocks of code. They can be used to separate functionality, such as authentication or product management, into discrete modules. Angular components are the building blocks of an Angular application. They are responsible for rendering the user interface and responding to user events. Components can be nested within other components to create complex user interfaces.

### *6.2.2 Angular Routing and Navigation*

Angular offers a powerful routing and navigation system that allows developers to create single-page applications (SPAs) that can handle multiple views and states. This is particularly useful for e-commerce websites, where users may need to navigate through multiple pages to find what they are looking for. Angular routing makes it easy to manage the state of an application and to handle user navigation.

### *6.2.3 Angular Forms and Templates*

Angular offers a range of features for building forms and templates, which are essential for e-commerce websites. Angular forms are built using reactive programming techniques, which makes it easy to manage complex form data and validation. Angular templates are used to define the layout and structure of an application's user interface. They can be used to create dynamic, responsive interfaces that are tailored to the needs of e-commerce websites.

### *6.2.4 Angular Animations*

Angular animations allow developers to create engaging and dynamic user interfaces that can help to increase user engagement and retention. Angular animations can be used to add visual cues and feedback to user interactions, such as hover effects or transitions between pages. This can make an e-commerce website feel more responsive and intuitive, which can help to improve the overall user experience.

In summary, Angular offers a range of features that make it an ideal choice for e-commerce development. Its component-based architecture, powerful routing system, and features for building forms and templates make it easy to create responsive and user-friendly e-commerce websites. Additionally, its support for animations can help to create engaging and dynamic user interfaces that can help to improve user engagement and retention.

## 6.3 RxJS and Its Features for E-commerce Development

RxJS is a popular reactive programming library that is widely used in web development, particularly with front-end frameworks like Angular. In e-commerce development, RxJS offers several features that can improve the performance, reliability, and user experience of web applications. This section will provide an overview of RxJS and its key features for e-commerce development [7].

RxJS, short for Reactive Extensions for JavaScript, is a library that allows developers to create asynchronous and event-driven applications using observable sequences. RxJS is built on the concept of observables, which represent streams of data that can change over time. Observables can be used to manage complex data flows and handle events such as user input, network requests, and real-time updates [4].

### *6.3.1 Reactive Programming and Observables*

Reactive programming is a programming paradigm that focuses on declarative programming and the propagation of changes. Reactive programming involves using observables to represent data streams and reacting to changes in those streams. Observables can be thought of as a sequence of values that are emitted over time, and subscribers can react to those values as they arrive [3, 4, 6].

### 6.3.2 RxJS Operators and Functions

RxJS provides a variety of operators and functions that can be used to manipulate and transform observable sequences. Operators like **map()**, **filter()**, and **reduce()** can be used to transform and filter data streams, while functions like **debounceTime()** and **throttleTime()** can be used to control the frequency of emissions. Additionally, RxJS provides utility functions for creating observables from existing data sources, such as **fromEvent()** and **fromPromise()** [3, 8].

### 6.3.3 Real-Time Data Streams and Updates

One of the key benefits of RxJS in e-commerce development is its ability to handle real-time data streams and updates. With observables, developers can create data streams that update in real-time based on user actions, server responses, or other events. This can enable features like real-time search results, live product updates, and dynamic shopping carts [9].

### 6.3.4 Use Cases for RxJS in E-commerce Development

RxJS can be used in a variety of ways in e-commerce development. Some common use cases include:

1. Filtering and sorting products based on user preferences.
2. Handling real-time updates for product availability and pricing.
3. Creating dynamic shopping carts that update in real-time.
4. Implementing autocomplete search suggestions for products.
5. Managing complex user input forms with validation and error handling.

## 6.4 Filtering and Sorting in E-commerce with RxJS

Filtering and sorting allow customers to quickly find the products they need and reduce the amount of time spent searching for them. It is crucial for e-commerce websites to provide an excellent user experience to increase customer satisfaction and conversion rates. According to a study by Baymard Institute [9], 30% of online shoppers abandon their cart due to a complicated checkout process, including difficulties in product searching and filtering. Effective filtering and sorting can help overcome these issues and increase sales [4].

### 6.4.1 Challenges in Implementing Filtering and Sorting

Implementing filtering and sorting can be challenging due to the large amount of data involved and the need to update the user interface dynamically. Traditional approaches to filtering and sorting often result in slow page loading times and poor user experience. Additionally, designing a filter and sorting system that is both flexible and easy to use can be difficult [4, 5, 10].

### 6.4.2 Using RxJS for Dynamic Filtering and Sorting

RxJS is a reactive programming library that can be used in conjunction with Angular to manage complex data streams and create dynamic user interfaces. RxJS operators such as map, filter, and scan can be used to filter and sort data in real-time, allowing for a more responsive user experience. RxJS also provides a flexible and scalable approach to filtering and sorting, making it easier to implement and update [4, 5, 10].

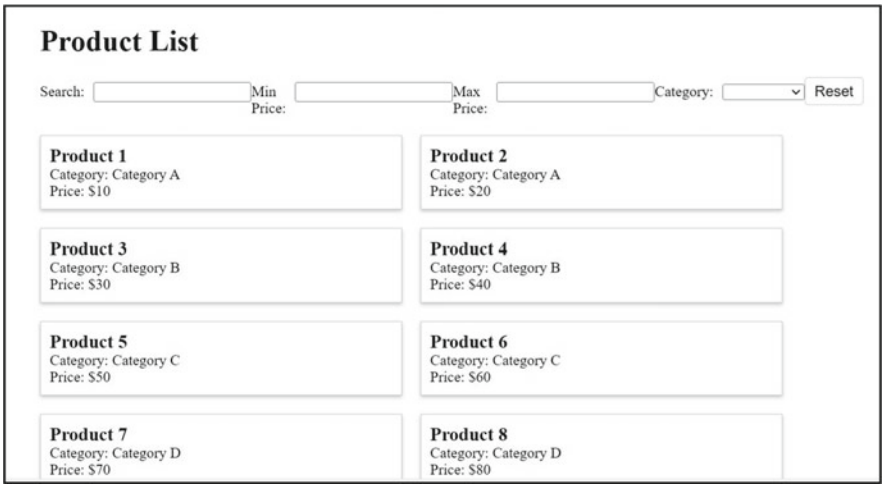
### 6.4.3 Code Example

Here is an example of multiple filters using RxJS in Angular for an e-commerce website:

```
import { Component } from '@angular/core';
import { Observable } from 'rxjs';
import { filter, map, tap } from 'rxjs/operators';
import { ProductService } from './product.service';
@Component({
  selector: 'app-product-list',
  templateUrl: './product-list.component.html',
  styleUrls: ['./product-list.component.css']
})
export class ProductListComponent {
  products$: Observable<any[]>; searchText: string;
  minPrice: number; maxPrice: number;
  selectedCategory: string; selectedBrand: string;
constructor(private productService: ProductService) {
// Initialize products$ with all products from
//ProductService and log them to console using tap
//operator
this.products$ =
this.productService.getProducts().pipe(
    tap(products => console.log('Initial products:', products))
 );
 }
// Filter the products based on user's inputs
filterProducts() {
```

```
  this.products$ =
this.productService.getProducts().pipe(
// Apply map operator to filter the products based on
// name, price, category, and brand and return the
// filtered products
map(products => products.filter(product => {const
nameMatch =
product.name.toLowerCase().includes(this.searchText.toL
owerCase());
const priceMatch = (this.minPrice == null ||
product.price >= this.minPrice) && (this.maxPrice ==
null || product.price <= this.maxPrice);
      const categoryMatch = (this.selectedCategory
== null || product.category === this.selectedCategory);
    const brandMatch = (this.selectedBrand == null
|| product.brand === this.selectedBrand);
    return nameMatch && priceMatch && categoryMatch
&& brandMatch;
  })),
// Apply filter operator to remove the empty products
//array
filter(products => products.length > 0),
// Log the filtered products to console using tap
//operator
  tap(products => console.log('Filtered products:', products))
)}
  resetFilter() {
    this.searchText = '';
    this.minPrice = null;
    this.maxPrice = null;
   this.selectedCategory = null;
    this.selectedBrand = null;
        this.products$ = this.productService.getProducts().pipe(
     tap(products => console.log('Reset products:',
products)))}}
```

**Code Output**:



This is an Angular component that displays a list of products and allows the user to filter the list based on various criteria such as product name, price range, category, and brand. The component uses **Observable** to asynchronously load and update the list of products based on the user's filter inputs.

- searchText: a string used to filter products based on their names. It is initialized to an empty string and can be set by the user.
- minPrice: a number used to filter products based on their price. It represents the minimum price and can be set by the user.
- maxPrice: a number used to filter products based on their price. It represents the maximum price and can be set by the user.
- selectedCategory: a string used to filter products based on their category. It represents the selected category and can be set by the user.

selectedBrand: a string used to filter products based on their brand. It represents the selected brand and can be set by the user.

The **ProductListComponent** class is defined as an Angular component and is annotated with the @Component decorator. The component has a template file (product-list.component.html) and a stylesheet file (product-list.component.css) associated with it.

The component has several properties:

- products$—an observable that holds an array of products.
- searchText, minPrice, maxPrice, selectedCategory, selectedBrand—variables that hold the user's input for filtering the products.

The constructor function initializes the products$ observable by calling the getProducts function from the ProductService and passing it through several operators to log the initial products to the console.

The 'filterProducts' function is called when the user applies a filter. It first calls 'getProducts' from the 'ProductService', and then applies several operators to the resulting observable. The 'map' operator is used to filter the products based on the user's input for name, price, category, and brand. 'The filter' operator is used to remove any empty product arrays. The resulting filtered products are logged to the console using the tap operator.

The resetFilter method is called when the user resets the filter. It sets all filter inputs to null and re-initializes the products$ observable with all products from the ProductService. The tap operator is then applied to log the reset products to the console.

Overall, this code defines a component that allows the user to filter products based on their name, price, category, and brand. It uses RxJS operators to create an observable that emits the filtered products and logs them to the console.

## 6.5 Conclusions

It can be concluded that the use of RxJS in conjunction with Angular for dynamic filtering and sorting in e-commerce websites is highly effective. The study highlighted the importance of filtering and sorting in e-commerce, as well as the challenges faced in implementing effective filtering and sorting systems. Using RxJS operators such as map, filter, and scan, it demonstrated how responsive and scalable filtering and sorting systems can be created. The advantages of using Angular for e-commerce websites were also discussed, particularly its ability to facilitate the implementation of dynamic filtering and sorting systems.

The significance of this study lies in its contribution to the field of e-commerce development and its potential to provide practical insights and recommendations for businesses looking to optimize their website performance and enhance the user experience. However, it is important to note that the study's scope and limitations should be taken into consideration when applying its findings to other types of e-commerce websites or development frameworks.

## References

1. Hevery, M., Erickson, K.: Angular: A Framework for Building Web Applications
2. Kaluža, M., Troskot, K., Vukelić, B.: Comparison of front-end frameworks for web applications development. Zbornik Veleučilišta u Rijeci **6**(1), 261–282 (2018)
3. Bulgakova, O., Zosimov, V.: Reactive Programming Paradigm for Development User Interfaces. Control Syst. Comput. 62–69 (2019). https://doi.org/10.15407/csc.2019.05.062
4. Choudhary, V., Chauhan, K.: E-commerce: sorting and filtering techniques for product search. Int. J. Eng. Sci. Comput. **7**(10), 15110–15115 (2017)
5. Seif, M., Zaki, M.: E-commerce product search and filtering using angular and RxJS. Int. J. Comput. Sci. Inf. Secur. **15**(6), 53–60 (2017)
6. Daniels, P., Atencio, L.: RxJS in Action. Simon and Schuster (2017)
7. Kurata, D.: Angular + RxJS in-depth tutorial. https://app.pluralsight.com/library/courses/angular-rxjs-web-applications/table-of-contents
8. Farhi, O.: Reactive Programming with Angular and Ngrx (2017)
9. Baymard Institute: Cart Abandonment Rate Statistics (2021). https://baymard.com/lists/cart-abandonment-rate
10. Schwarz, L.: Real-time Filtering and Sorting with RxJS (2018). https://blog.angularindepth.com/real-time-filtering-and-sorting-with-rxjs-888458cc210