SHOPEZ - ONE STOP SHOP FOR ONLINE PURCHASES

Submitted for the course

SmartBridge

Modern Application Development (Java Spring Boot)

Submitted By

-

20BEC1116-Sweta Thanu

20BCE11002-Hemant Kumar Yadav

20BCE10447-Sanket Kabra

CONTENTS

1. Introduction

1.1. Overview

A One-Stop Shop application is a centralized platform that consolidates multiple services and functionalities into a single interface. It aims to simplify the user experience by providing a convenient and efficient way to access various products, services, and information without the need to switch between different applications or platforms. By integrating diverse features into one place, users can save time, effort, and enjoy a more cohesive experience. Whether it's shopping, financial management, travel planning, or other tasks, a One-Stop Shop application offers a unified solution for users' diverse needs.
By integrating multiple services and features, the One-Stop Shop application eliminates the need for users to navigate through different websites or apps to accomplish different tasks. It offers a seamless experience by providing access to various services, such as e-commerce, financial transactions, bookings, customer support, and more, all in one place.

1.2. Purpose

A One-Stop Shop application is developed with the purpose of providing users a centralized platform where they can access multiple services and functionalities in a convenient and efficient manner. By eliminating the need to switch between various apps or websites, it simplifies the user experience, saving time and effort. Whether it's shopping, banking, booking, or other tasks, users can access diverse services from a unified solution. These applications enhance convenience, streamline management, and provide a cohesive user experience. The ultimate goal is to improve efficiency, accessibility, and user satisfaction by offering a single, comprehensive platform for fulfilling diverse needs.

2. Literature Survey

No.	Title	Author	Proposed Work
1.	Modern API Development with Spring and Spring Boot: Design highly scalable and maintainable APIs with REST, gRPC, GraphQL, and the reactive paradigm (2021)	S. Sharma	This book is a comprehensive guide that explores the fundamentals of RESTful APIs, Spring and Spring Boot concepts, API specifications, business logic implementation, and asynchronous API design. The book provides practical insights, best practices, and hands-on examples for building scalable and efficient APIs. It covers a range of topics including API design principles, error handling, data persistence, and the migration to asynchronous programming. This book serves as a valuable resource for developers seeking to enhance their API development skills using Spring

			and Spring Boot frameworks.
2.	Beginning React: Simplify your frontend development workflow and enhance the user experience of your applications with React (2018)	A. Chiarelli	This book introduces React and covers key aspects of frontend development using React. The book provides insights into designing user interfaces, creating and managing components, and handling user interactivity. With practical examples and a focus on enhancing the user experience, this book serves as a valuable resource for developers seeking to streamline their frontend development workflow and leverage the power of React in their applications.
3.	Comparison of MySQL and MongoDB with focus on performance (2020)	P. Filip, L. Čegan	The paper provides an overview and comparison of NoSQL databases, specifically focusing on the performance differences between MySQL and MongoDB. The study includes benchmark analyses of various operations, such as data insertion, update, and deletion, with and without transactions. The paper aims to evaluate the impact of data storage structures on database performance and assess the added value of benchmarks in terms of indexed field costs. This research contributes to the understanding of performance considerations in selecting between MySQL and MongoDB for different application scenarios.
4.	Light-Weight and Scalable Hierarchical-MVC Architecture for Cloud Web Applications (2018)	M. Ma, J. Yang, P. Wang, W. Liu J. Zhang	This paper addresses the challenges of modular and scalable web application development in the cloud computing era. The proposed approach, called Web Module Definition (WMD), introduces a hierarchical-MVC architecture that supports feature-based modularization and application structure. By decomposing the web application into interconnected modules, WMD enables better scalability and maintainability. The paper presents a demonstration website and a web application framework implementation that supports the WMD-based architecture. This research contributes to the advancement of web application development in the

			cloud by offering a lightweight and scalable solution for modularizing and organizing web applications.
5.	Building Modern Clouds: Using Docker, Kubernetes & Google Cloud Platform (2019)	J. Shah, D. Dubaria	The presented work explores the transformative impact of Docker and Kubernetes on cloud infrastructure and DevOps practices. The abstract highlights the capabilities of Docker in building, shipping, and running applications using containers, resulting in faster deployments, resource efficiency, and reliability. It also emphasizes the automation of container management, deployment, and scaling provided by Kubernetes, along with the benefits of using Google Cloud Platform for deploying containers on Kubernetes Engine. This paper serves as a concise resource for understanding the significance of Docker, Kubernetes, and Google Cloud Platform in developing modern cloud architectures and facilitating efficient application development and management.

2.1. Existing problem

One-Stop Shop applications face several existing problems that can hinder their effectiveness. First, integrating a wide range of services from different providers can be challenging due to limited collaboration and data sharing. This may result in a restricted selection of offerings within the application. Second, user adoption and behavior pose a significant challenge. Users may already be accustomed to using specific apps or platforms for specific tasks, making it difficult to convince them to switch to a new One-Stop Shop application and change their established behavior.

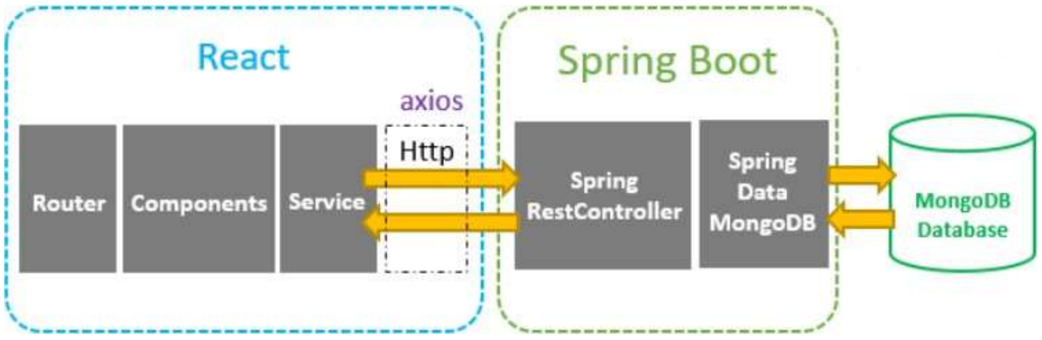
Additionally, concerns regarding data security and privacy arise when consolidating multiple services into one application. The combination of various data streams increases the risk of unauthorized access and potential data breaches. Maintaining a consistent user experience across different services within the One-Stop Shop application can also be problematic. Each service may have its own design and user interface, leading to a fragmented or disjointed experience for users. Overcoming these challenges requires strategic partnerships, technical expertise, user education, strong security measures, and continuous efforts to improve the user experience.

2.2. Proposed Solution

One-Stop Shop applications offer several benefits to users. They provide convenience by consolidating multiple services into a single platform, saving users time and effort. The efficiency of these applications streamlines user interactions, allowing them to perform various tasks seamlessly. With enhanced accessibility, users can manage their accounts and transactions from a centralized hub. Personalization features provide tailored recommendations based on user preferences. The applications ensure a consistent user experience, simplifying management and offering cost savings through exclusive deals. Real-time updates and notifications keep users informed, ultimately leading to higher satisfaction with the overall experience.

3. Theoretical Analysis

3.1. Block Diagram



3.2. Hardware / Software designing

Hardware Requirements

Software Requirements

1. JDK
2. Maven
3. IntelliJ IDE
4. MongoDB Atlas
5. Bootstrap
6. React
7. Axios

4. Experimental Investigations

Experimental investigations on One-Stop Shop applications can provide valuable insights into their effectiveness, user satisfaction, and impact on various metrics. Here are some potential areas of investigation:

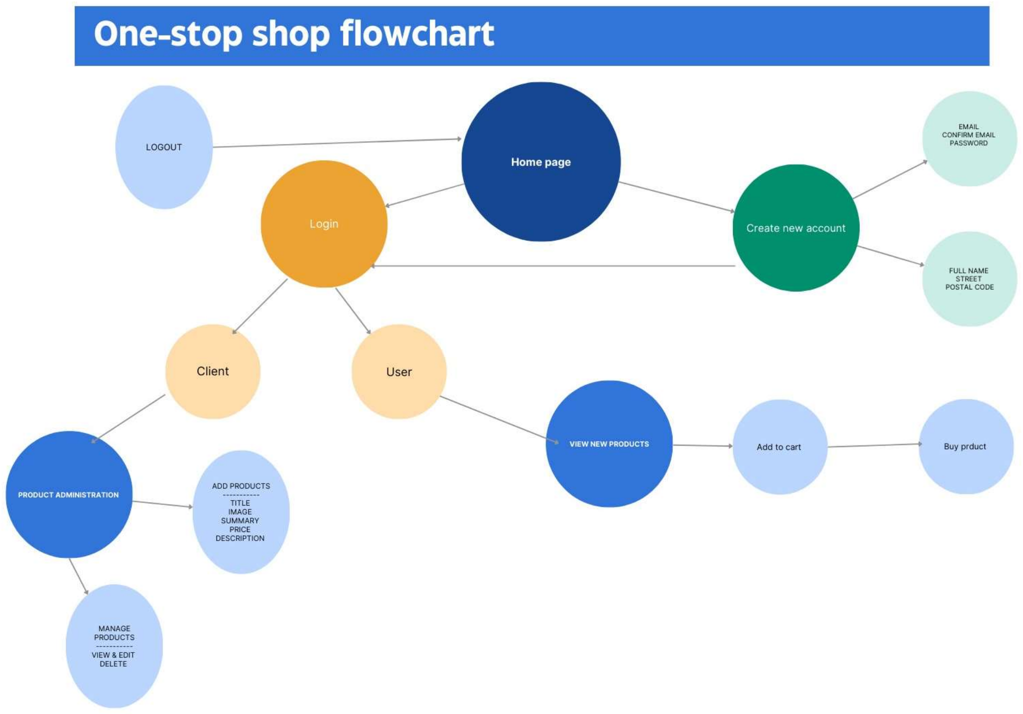
User Experience: Conducting user experience studies can assess the ease of use, efficiency, and overall satisfaction of users while interacting with a One-Stop Shop application. This can involve gathering qualitative feedback, conducting usability testing, and analyzing user behavior data to identify areas for improvement.

Efficiency and Time Savings: Experimental investigations can compare the time required for users to complete tasks using a One-Stop Shop app versus traditional methods involving multiple platforms. Quantitative measurements and metrics can be used to evaluate the efficiency and time savings achieved by using the application.

User Adoption and Behavior: Experimental research can explore user adoption patterns, factors influencing users' decisions to use or not use the One-Stop Shop app, and the impact of incentives or promotional strategies on user behavior.

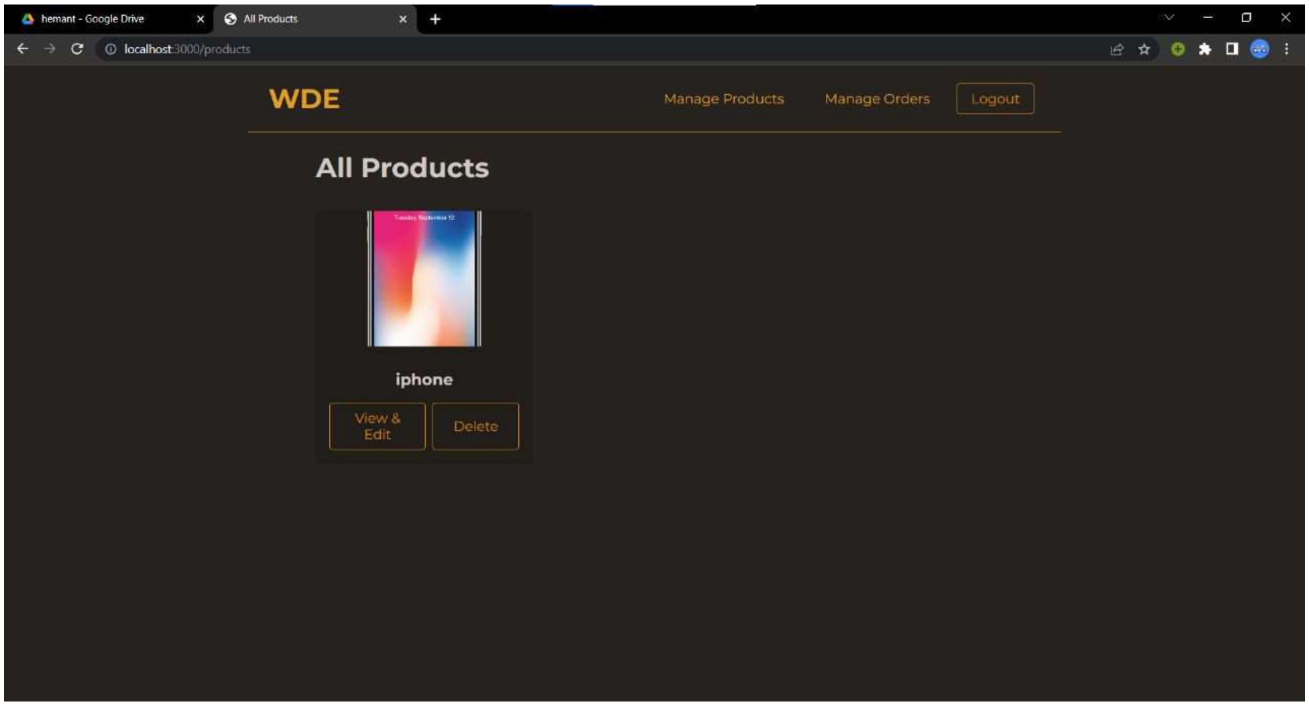
Analysis or the investigation made while working on the solution.

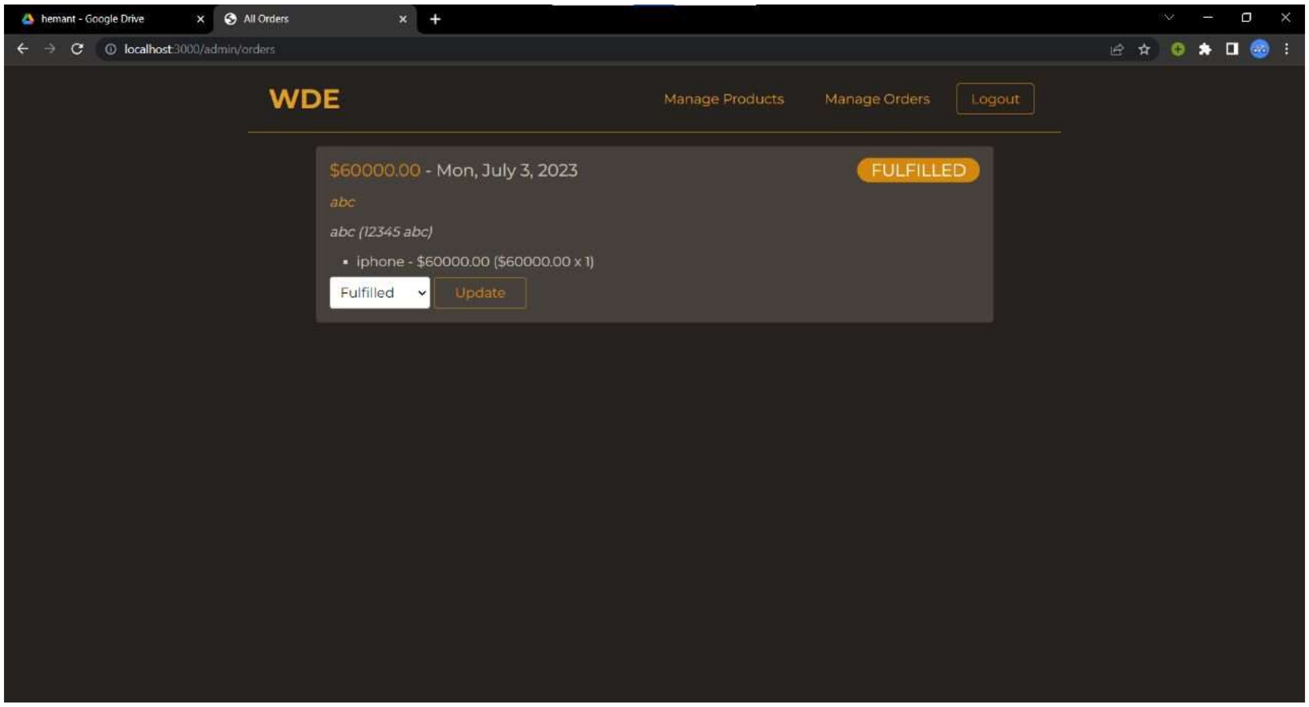
5. Flowchart



6. Result

Final findings (Output) of the project along with screenshots



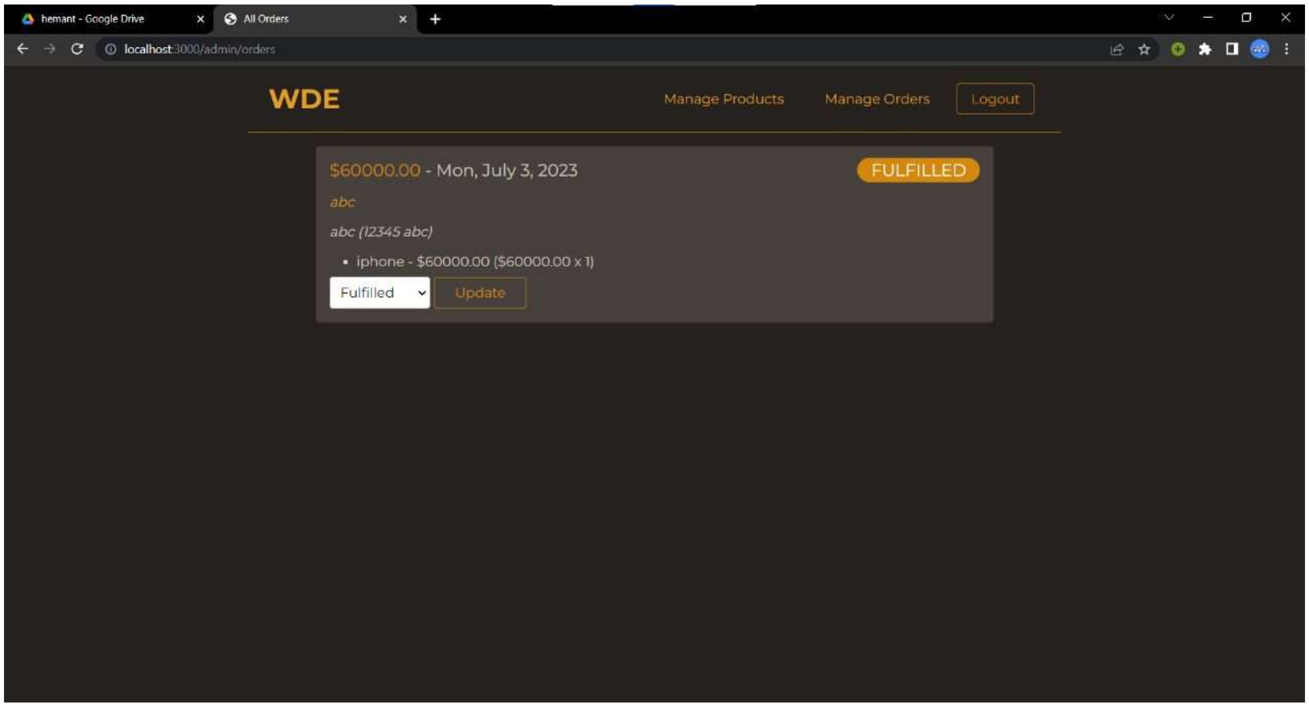


$60000.00 - Mon, July 3, 2023

abc

abc (12345 abc)

- iphone - $60000.00 ($60000.00 x 1)



$60000.00 - Mon, July 3, 2023

abc

abc (12345 abc)

- iphone - $60000.00 ($60000.00 x 1)

7. Advantages & Disadvantages

The key advantages of our platform include:

i. Convenience: Users can access a wide range of services and functionalities from a single platform, eliminating the need to use multiple apps or websites.

ii. Time Savings: Users save time by avoiding the hassle of navigating between different platforms or searching for specific services. Everything they need is conveniently available in one place.

iii. Seamless Experience: One-Stop Shop applications provide a cohesive and consistent user experience. Users can navigate easily, as the design, interface, and interactions remain familiar across different services.

iv. Simplified Management: Users can manage multiple accounts, transactions, and preferences from a centralized location. This simplifies their overall management and makes it easier to track and control various activities.

v. Personalization: These applications often offer personalized recommendations based on user preferences and behavior. Users receive tailored suggestions and content that align with their interests and needs.

Disadvantages of the platform:

I. Dependence on a Single Platform: Users become reliant on a single application for multiple services. If there are technical issues or downtime with the application, it can disrupt access to all the services integrated within it.

II. Integration Challenges: Integrating different services from various providers into a single platform can be complex. Incompatibility issues, data sharing restrictions, or resistance from service providers can hinder the seamless integration of services.

III. Privacy and Security Concerns: Consolidating multiple services into one application raises concerns about data privacy and security. Users may worry about the safety of their personal information, especially if there is a data breach or unauthorized access to the consolidated data.

IV. User Interface and Experience Limitations: One-Stop Shop applications often aim for a standardized user interface and experience across different services. However, this can result in a less tailored experience for each specific service, potentially sacrificing some of the unique features or functionalities that users may prefer.

V. User Adaptation and Learning Curve: Users accustomed to using separate apps or websites for different services may find it challenging to adapt to a new One-Stop Shop application. Adjusting to a different interface and learning how to navigate the consolidated services may require some time and effort.

8. Applications

The application of a One-Stop Shop app is diverse and can be implemented in various industries and sectors. Here are some examples of how One-Stop Shop apps can be applied:

- ✧ E-commerce: A One-Stop Shop app can bring together multiple online stores, allowing users to browse and purchase products from various retailers within a single platform. It provides a convenient shopping experience with access to a wide range of products and brands.

- ✧ Travel and Tourism: A One-Stop Shop app can integrate flight bookings, hotel reservations, car rentals, and travel itineraries. Users can plan and manage their entire travel experience, from booking flights to exploring local attractions, all in one application.

- ✧ Financial Services: One-Stop Shop apps can combine banking, investment, insurance, and payment services. Users can perform various financial transactions, access account information, and manage their personal finances from a single platform.

- ✧ Food Delivery: A One-Stop Shop app can aggregate multiple food delivery services, offering users a variety of restaurants and cuisines to choose from. Users can browse menus, place orders, and track deliveries, simplifying the food ordering process.

- ✧ Healthcare: One-Stop Shop apps in healthcare can integrate features like booking doctor appointments, accessing medical records, ordering medications, and receiving telemedicine consultations. It provides users with comprehensive healthcare services and convenient access to healthcare providers.

- ✧ Government Services: One-Stop Shop apps can bring together different government services, such as applying for identification documents, paying taxes, accessing public services, and submitting official forms. Users can access and interact with various government services through a single application.

9. Conclusion

A One-Stop Shop app project aims to provide users with a comprehensive platform that integrates multiple services and functionalities into a single application. The project offers numerous advantages, including convenience, time savings, a seamless user experience, simplified management, personalization, and cost savings. It caters to users' diverse needs and enhances their overall satisfaction by providing a centralized hub for various services.

However, the project also faces challenges such as Slimited-service options, integration complexities, privacy concerns, user adaptation, and competition from established platforms. Overcoming these challenges requires strategic collaborations, robust data security measures, user education, continuous improvement based on feedback, and differentiation through unique value propositions.

By implementing these proposed solutions and considering the potential drawbacks, a successful One-Stop Shop app project can effectively meet the demands of users and establish itself as a reliable and preferred platform in various industries and sectors.

10.Future Scope

The future holds immense potential for enhancing the functionality and user experience of our platform. By incorporating advanced recommendation algorithms, we can further personalize product suggestions, offering users tailored recommendations based on their preferences and behaviour.

To improve product discovery, we plan to implement advanced search and filtering options. This will allow users to find product based on specific criteria such as genre, quality, brand, manufactured date, or user ratings. Additionally, user profile customization will enable users to curate their own product lists, highlight favourite genres, and receive personalized recommendations based on their preferences.

Recognizing the importance of mobile accessibility, we aim to develop mobile applications for both Android and iOS platforms. This will empower users to access the platform and receive product recomendations on the go, ensuring a seamless and convenient experience.

Furthermore, integrating external review sources, such as professional critics' reviews , will provide a diverse range of opinions and perspectives on product. This will enrich the platform's content and enable users to make more informed decisions. Additionally, we plan to introduce multi-language support, enabling users from different regions to access product information and reviews in their preferred language.

11. Bibliography

Filip, Petr, and Lukáš Čegan. "Comparison of mysql and mongodb with focus on performance." *2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*. IEEE, 2020.

Ma, Meng, et al. "Light-weight and scalable hierarchical-MVC architecture for cloud web applications." *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*. IEEE, 2019.

Shah, Jay, and Dushyant Dubaria. "Building modern clouds: using docker, kubernetes & Google cloud platform." *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2019.

Manek, Asha S., P. Deepa Shenoy, and M. Chandra Mohan. "Aspect term extraction for sentiment analysis in large movie reviews using Gini Index feature selection method and SVM classifier." *World wide web* 20 (2017): 135-154.

Banker, Kyle, et al. *MongoDB in action: covers MongoDB version 3.0*. Simon and Schuster, 2016.

Luksa, Marko. *Kubernetes in action*. Simon and Schuster, 2017.

APPENDIX

A. Source Code

```
const path = require('path');

const express = require('express');
const csrf = require('csurf');
const expressSession = require('express-session');

const createSessionConfig = require('./config/session');
const db = require('./data/database');
const addCsrfTokenMiddleware = require('./middlewares/csrf-token');
const errorHandlerMiddleware = require('./middlewares/error-handler');
const checkAuthStatusMiddleware = require('./middlewares/check-auth');
const protectRoutesMiddleware = require('./middlewares/protect-routes');
const cartMiddleware = require('./middlewares/cart');
const updateCartPricesMiddleware = require('./middlewares/update-cart-prices');
const notFoundMiddleware = require('./middlewares/not-found');
const authRoutes = require('./routes/auth.routes');
const productsRoutes = require('./routes/products.routes');
const baseRoutes = require('./routes/base.routes');
const adminRoutes = require('./routes/admin.routes');
const cartRoutes = require('./routes/cart.routes');
const ordersRoutes = require('./routes/orders.routes');

const app = express();

app.set('view engine', 'ejs');
app.set('views', path.join(__dirname, 'views'));

app.use(express.static('public'));
app.use('/products/assets', express.static('product-data'));
app.use(express.urlencoded({ extended: false }));
app.use(express.json());

const sessionConfig = createSessionConfig();

app.use(expressSession(sessionConfig));
app.use(csrf());

app.use(cartMiddleware);
app.use(updateCartPricesMiddleware);

app.use(addCsrfTokenMiddleware);
app.use(checkAuthStatusMiddleware);

app.use(baseRoutes);
app.use(authRoutes);
app.use(productsRoutes);
app.use('/cart', cartRoutes);
app.use('/orders', protectRoutesMiddleware, ordersRoutes);
app.use('/admin', protectRoutesMiddleware, adminRoutes);

app.use(notFoundMiddleware);

app.use(errorHandlerMiddleware);

db.connectToDatabase()
  .then(function () {
    app.listen(3000);
  })
  .catch(function (error) {
    console.log('Failed to connect to the database!');
    console.log(error);
  });
```

```
{
  "name": "one-stop-shop",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "nodemon app.js"
  },
  "keywords": [],
  "author": "Hemant Kumar",
  "license": "ISC",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "connect-mongodb-session": "^3.0.0",
    "csurf": "^1.11.0",
    "ejs": "^3.1.6",
    "express": "^4.17.1",
    "express-session": "^1.17.2",
    "mongodb": "^4.1.0",
    "multer": "^1.4.3",
    "uuid": "^8.3.2"
  },
  "devDependencies": {
    "nodemon": "^2.0.12"
    }
  }
```