GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

E-COMMERCE AUTOMOTIVE

A Report for the Evaluation 3 of Project 2

Submitted by

AMAN SHARMA

(1613101107)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Under the Supervision of

Mr. SUBHASH CHANDRA GUPTA

Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"E-COMMERCE"** is the bonafide work of **"AMAN SHARMA (1613101107)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD

Dr. MUNISH SHABARWAL,
PhD(Management), PhD(CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Mr. Subhash Chandra Gupta
M.Tech(SE),B.E(CSE)
Professor,
School of Computing Science & Engineering

ABSTRACT

The market globalization and production globalization of the automotive industry make the customers, marketing network, suppliers and other related resources also included in the scope of management, not only the enterprise itself. Then manufacturing companies use the Internet to create a virtual company's extended supply chain to meet these demands. Electronic commerce technologies represent one mechanism to manage boundary spanning activities in the supply chain, and it promotes the company's supply chain process reengineering, also is conducive to improve organization structure refinement and operational performance. This paper constructs the automotive e-commerce supply chain model, and analyzes the functions of this model. Finally, a case about automotive e-commerce supply chain management is given to prove this model. The case study indicates that the given model is effective and is suitable for the automotive industry. This paper contributes to the e-commerce supply chain management practices in the automotive industry.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The term e-commerce was coined back in the 1960s, with the rise of electronic commerce – the buying and selling of goods through the transmission of data – which was made possible by the introduction of the electronic data interchange. Fast forward fifty years and e-commerce has changed the way in which society sells goods and services.

E-commerce has become one of the most popular methods of making money online and an attractive opportunity for investors. For those interested in buying an e-commerce business, this article serves to provide an introduction to e-commerce, covering the reasons for its popularity, the main distribution models and a comparison of the major e-commerce platforms available.

E-commerce has transformed the way business is done in India. The Indian e-commerce market is expected to grow to US$ 200 billion by 2026 from US$ 38.5 billion as of 2017. Much growth of the industry has been triggered by increasing internet and smartphone penetration. The ongoing digital transformation in the country is expected to increase India's total internet user base to 829 million by 2021 from 636.73 million in FY19. India's internet economy is expected to double

from US$ 125 billion as of April 2017 to US$ 250 billion by 2020, majorly backed by ecommerce. India's E-commerce revenue is expected to jump from US$ 39 billion in 2017 to US$ 120 billion in 2020, growing at an annual rate of 51 per cent, the highest in the world.

During April-June quarter 2019, smartphone shipment in India grew 9.9 per cent year-on-year to 36.9 million shipments. It is expected to reach 160 million in 2019.

In 2017, Forrester Research predicted that the B2B e-commerce market will top $1.1 trillion in the U.S. by 2021, accounting for 13% of all B2B sales in the nation.

Business-to-consumer (B2C) is the retail part of e-commerce on the internet. It is when businesses sell products, services or information directly to consumers. The term was popular during the dot-com boom of the late 1990s, when online retailers and sellers of goods were a novelty.

Today, there are innumerable virtual stores and malls on the internet selling all types of consumer goods. The most recognized example of these sites is Amazon, which dominates the B2C market.

Consumer-to-consumer (C2C) is a type of e-commerce in which consumers trade products, services and information with each other online. These transactions are

generally conducted through a third party that provides an online platform on which the transactions are carried out.

Online auctions and classified advertisements are two examples of C2C platforms, with eBay and Craigslist being two of the most popular of these platforms. Because eBay is a business, this form of e-commerce could also be called C2B2C -- consumer-to-business-to-consumer.

Consumer-to-business (C2B) is a type of e-commerce in which consumers make their products and services available online for companies to bid on and purchase. This is the opposite of the traditional commerce model of B2C.

A popular example of a C2B platform is a market that sells royalty-free photographs, images, media and design elements, such as iStock. Another example would be a job board.

Business-to-administration (B2A) refers to transactions conducted online between companies and public administration or government bodies. Many branches of government are dependent on e-services or products in one way or another, especially when it comes to legal documents, registers, social security, fiscals and employment. Businesses can supply these electronically. B2A services have grown considerably in recent years as investments have been made in e-government capabilities.

1.1 What is a website?

A website is a group of globally accessible, interlinked web pages which have a single domain name. It can be developed and maintained by an individual, business or organization. The website aims to serve a variety of purposes. Example: Blogs.A website is hosted on a single or multiple web server.

1.2 What is a Web Application?

A web application is a software or program which is accessible using any web browser. Its frontend is usually created using languages like HTML, CSS, Javascript, which are supported by major browsers. While the backend could use any programming stack like LAMP, MERN, etc. Unlike mobile apps, there is no specific SDK for developing web applications.

1.3 Why do we need a Web Application?

Web applications are more popular because of the following reasons:

- Compared to desktop applications, web applications are easier to maintain as they use the same code in the entire application. There are no compatibility issues.
- Web applications can be used on any platform: Windows, Linux, Mac… as they all support modern browsers.
- Mobile App store approval not required in web applications.

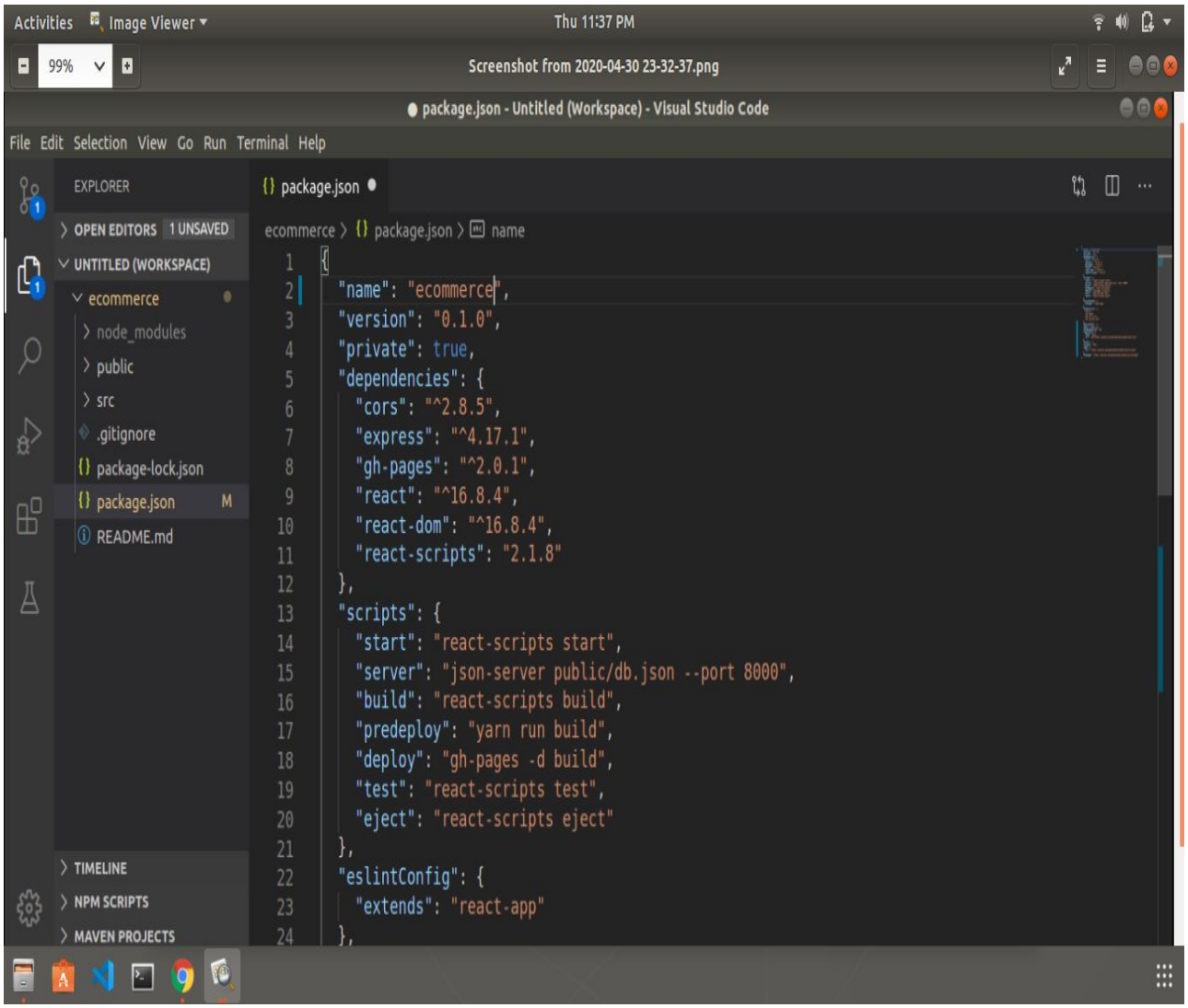
- Released any time and in any form. No need to remind users to update their applications.
- You can access these web applications 24 hours of the day and 365 days a year from any PC.
- You can either make use of the computer or your mobile device to access the required data.
- Web applications are a cost-effective option for any organization. Seat Licenses for Desktop software are expensive where SasS are generally, pay as you go.
- Web-Based Apps are Internet-enabled apps that are accessed through the mobile's web browser. Therefore, you don't required to download or install them.

1.4 Purpose of choosing this project

I chose this project to help the tourists all around the world, I know what problems tourists face when he goes to a new place, so I have added many features that can help them in getting some information prior going that place like nearest railway station, bus stand, hotels, tickets fare etc, I have also added features so that people can upload photos, give ratings ,reviews, do comments about a visited place. I have used different technologies like html, css, javascript ,bootstrap for the frontend and nodeJS for backend ,mongodb as database and heroku for deployment.

I have divided this app in many modules so that it follows the standards of web development and also followed the REST(representational state transfer) convention for routing this app, and also made basic CRUD(create, read, update, destroy) functionalities, i have also used features like associations and authentication.

Fig 1: List of modules, files and npm packages used in this project



1.5 Existing System

Some of the major developments in the Indian e-commerce sector are as follows.In August 2019, Amazon acquired 49 per cent stake in a unit of Future Group.Reliance to invest Rs 20,0000 crore (US$ 2.86 billion) in its telecom business to expand its broadband and E-commerce presence and to offer 5G services.In September 2019, PhonePe launched super-app platform 'Switch' to provide a one stop solution for customers integrating several other merchants' apps.

In November 2019, Nykaa opened its 55th offline store marking success in tier II and tier III cities.

1.6 Proposed System

Our web app have all the basic features that a website contains, like login, signup, profile of the user, authentication features, search bar, filter add the product to basket, add the product description, signin and signup And most important functionality which is not provided by other websites like, do the comparing between the other item in the list and they can add photos of that place, edit them, do comments, rate etc.

1.7 Problem Statement:

I also faced many problems when I was doing this project like which tools I should use, which technologies I should use, which programming language for backend, which framework for frontend, from where to learn to use these technologies. What features I should add in this webapp.

1.8 List of modules and usage:

Views: This module will contain all the express files for different routes.

Node Modules: This will contain all the node packages that we will need to develop our app.

Public: This will contain all the stylesheets to style our app.

Routes: This will contain all the different routes we will need in our app, Ex-user, add places,

comments, login, signup, update, show info etc.

Middleware: This will contain all the code that we need to run when some event occurs on

our app.

Our end product is a fully functional web application that can be accessed by the user from anywhere on any device and on any network, this app can also be used on mobiles

2. SYSTEM REQUIREMENTS

2.1 Hardware Requirements:

Processor(Core 2 Duo or above).

RAM(2gb or above).

HDD(100gb or above).

2.2 Software Requirements:

Environment used: VS Code.

Languages and Technologies Used to build this application:-

Frontend: ReactJs.

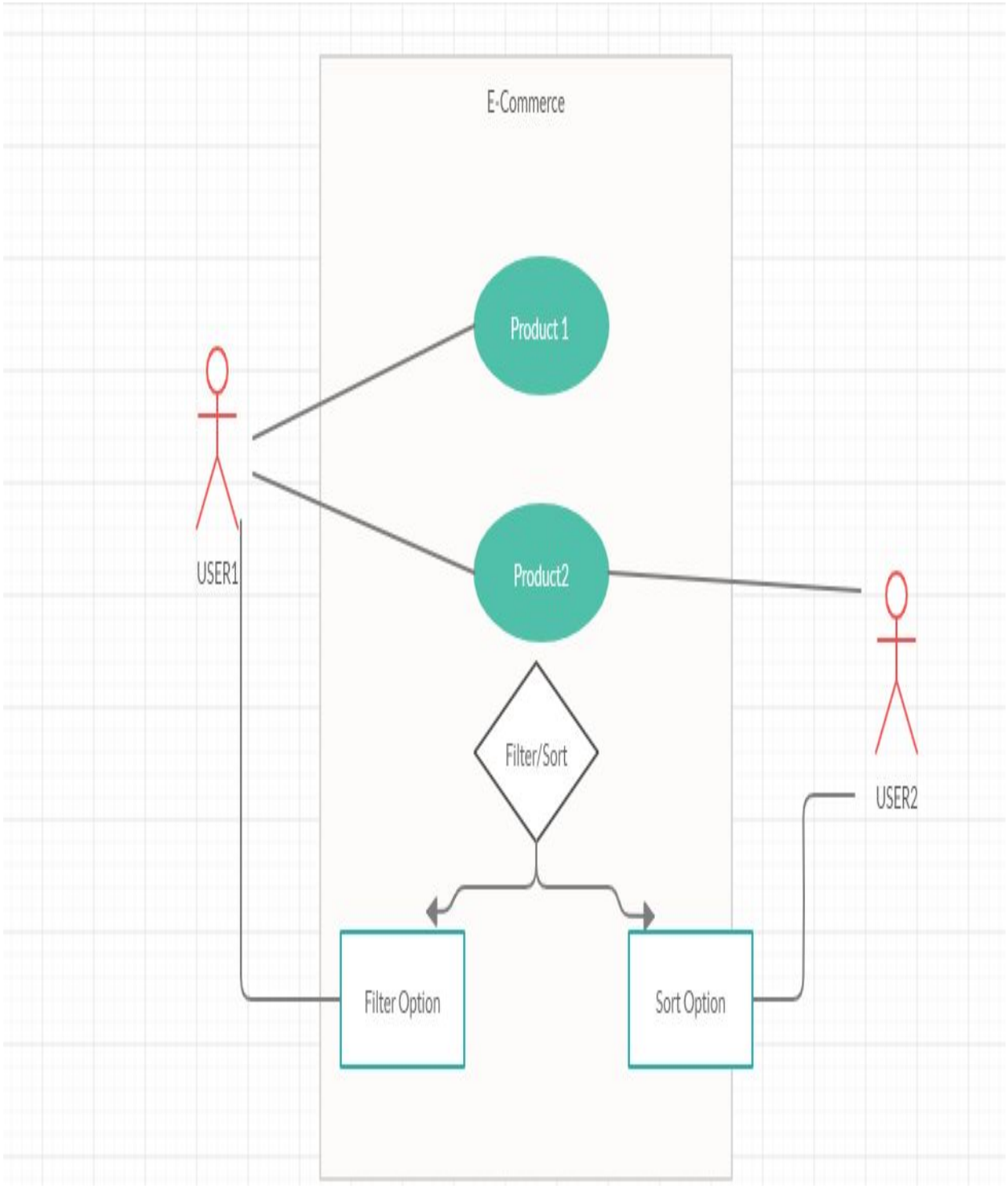
Backend: Node JS, Express, Mongoose.

Database: MongoDB.

Version Control System: Git.

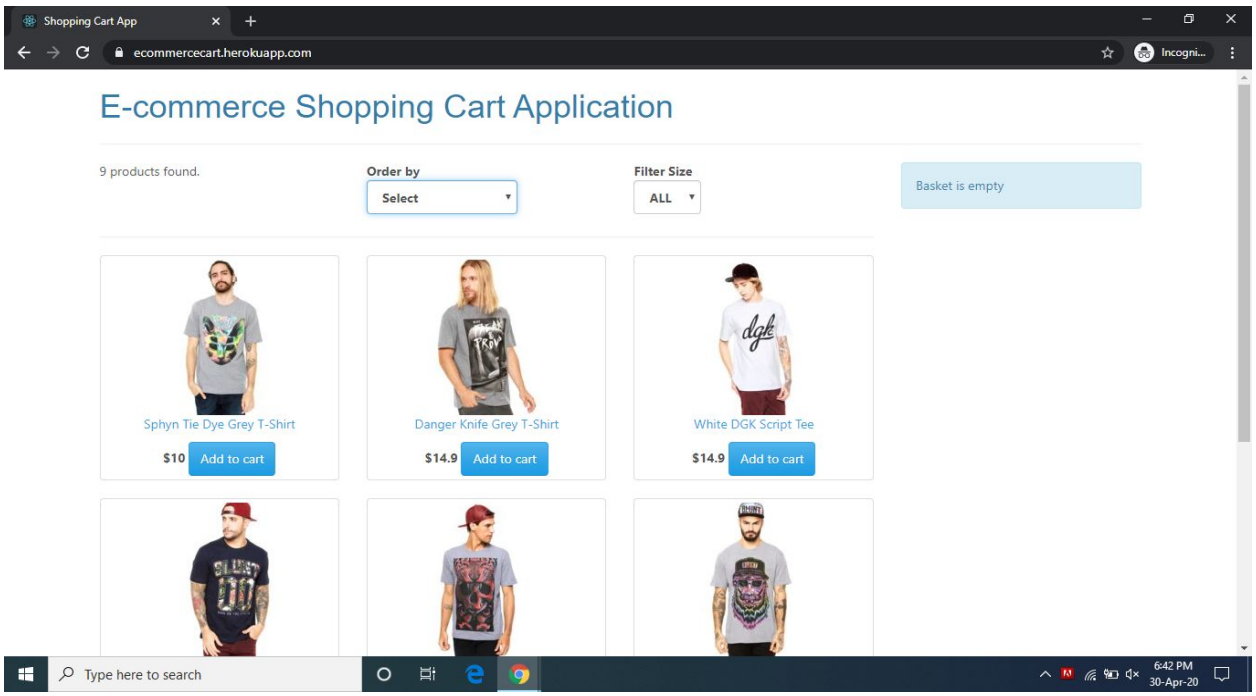
Heroku: For deployment.

3. IMPLEMENTATION DIAGRAM

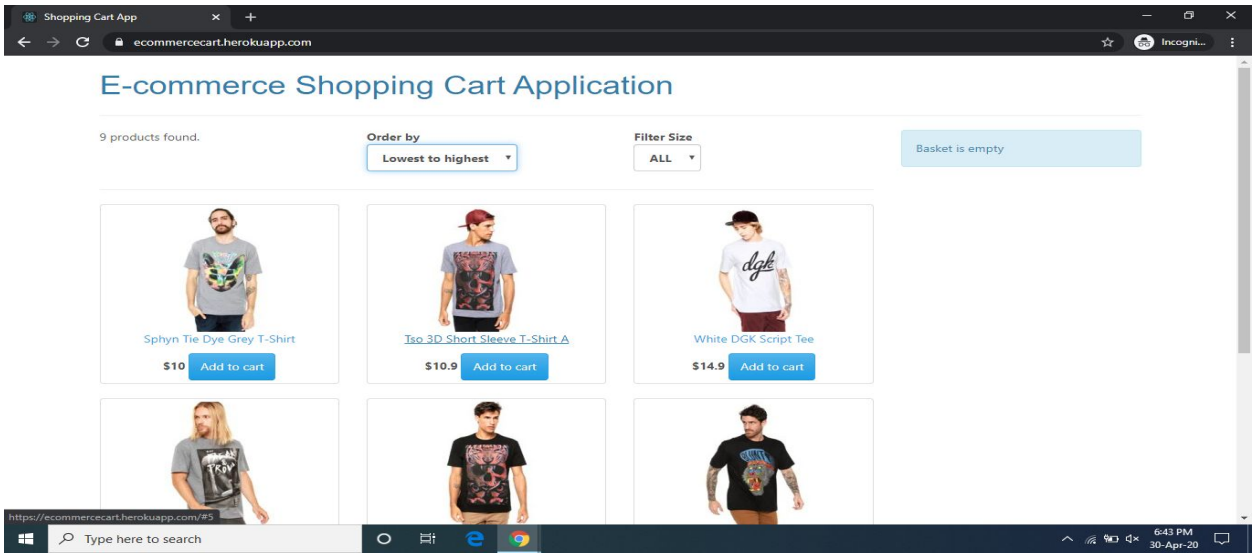


4. OUTPUTS

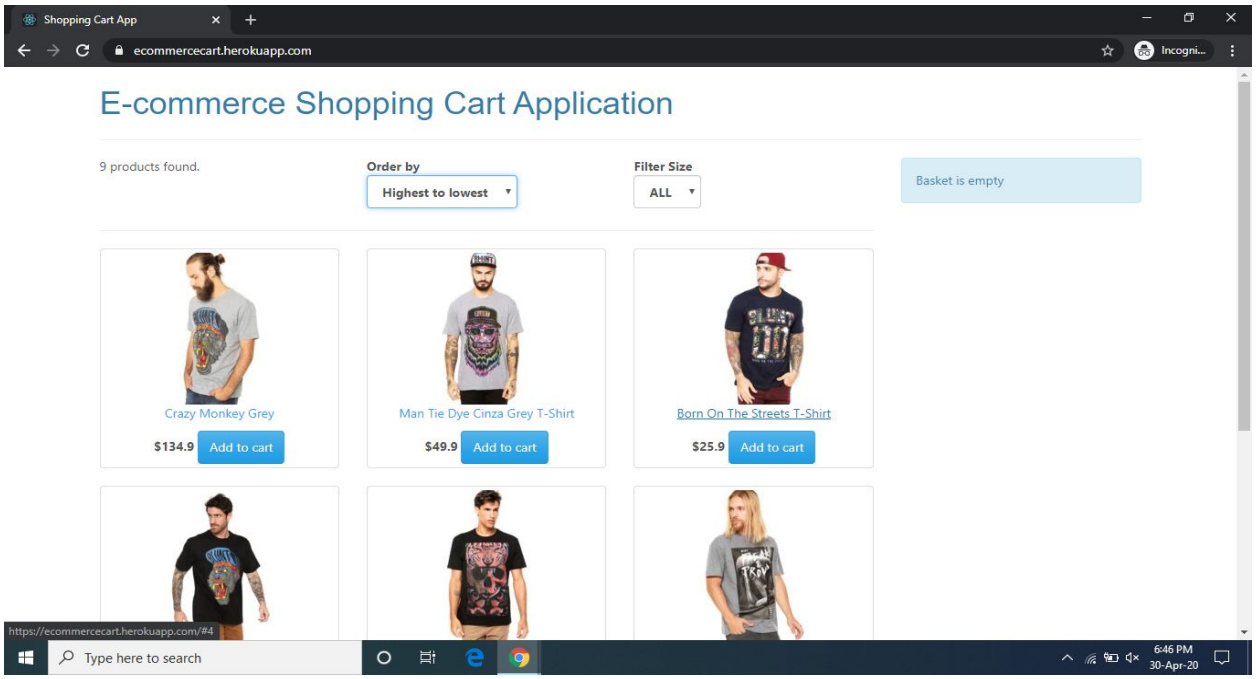
When the cart is empty.



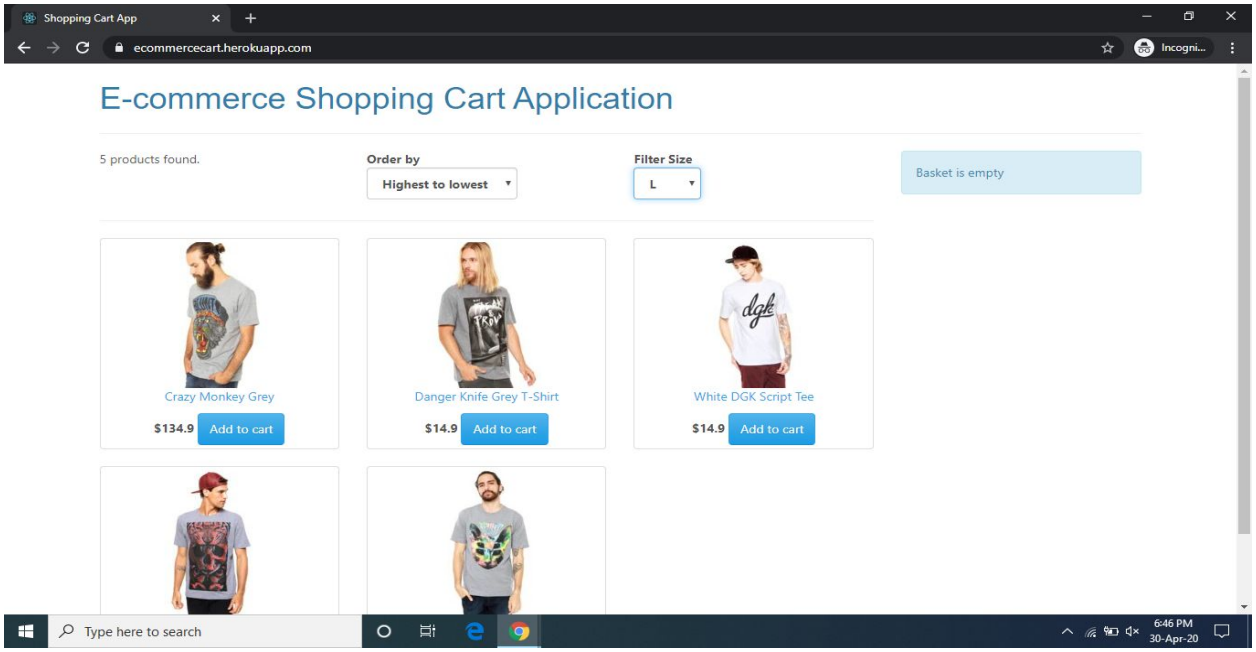
Sort from Lowest To highest



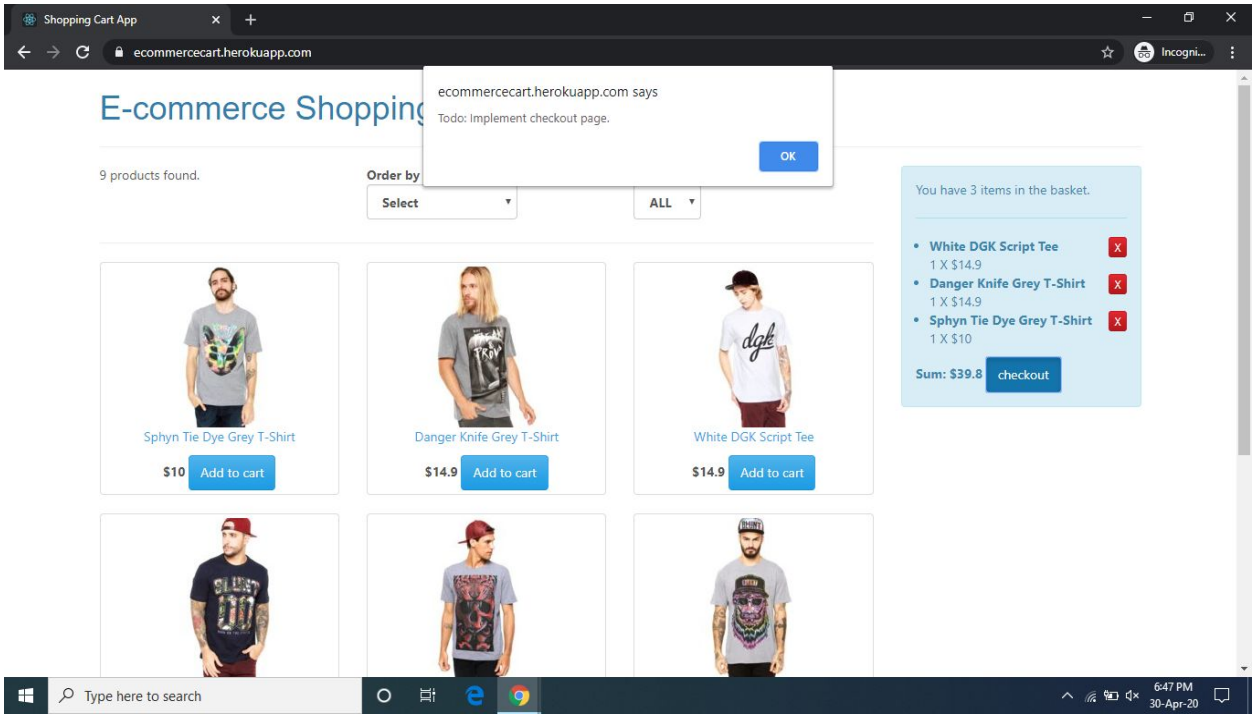
Sort from highest to lowest



Filter the Large-Size



Adding to the cart and Checkout



5. SOURCE CODE

This chapter contains the source code for the whole web app, our web app has many modules and each module contains code for different purposes. App.js is the file from where our app will start because this file binds all the modules together. Different modules that present in our web app are: Middleware, Public, Routes, Views and this module further contains different sub modules namely headers,filters,baskets components.

5.1 App.js

```
import React, { Component } from "react";
import Products from "./components/Products";
import Filter from "./components/Filter";
import Basket from "./components/Basket";

import "./App.css";
import Copyright from "./components/Copyright";

class App extends Component {
  constructor() {
    super();
    this.state = {
      size: "",
      sort: "",
      cartItems: [],
      products: [],
      filteredProducts: []
    };
  }
  componentWillMount() {
    if (localStorage.getItem("cartItems")) {
      this.setState({
        cartItems: JSON.parse(localStorage.getItem("cartItems"))
```

```
    });
  }

  fetch("http://localhost:8000/products")
    .then(res => res.json())
    .catch(err =>
      fetch("db.json")
        .then(res => res.json())
        .then(data => data.products)
    )
    .then(data => {
      this.setState({ products: data });
      this.listProducts();
    });
}

handleRemoveFromCart = (e, product) => {
  this.setState(state => {
    const cartItems = state.cartItems.filter(a => a.id !== product.id);
    localStorage.setItem("cartItems", JSON.stringify(cartItems));
    return { cartItems: cartItems };
  });
};

handleAddToCart = (e, product) => {
  this.setState(state => {
    const cartItems = state.cartItems;
    let productAlreadyInCart = false;

    cartItems.forEach(cp => {
      if (cp.id === product.id) {
        cp.count += 1;
        productAlreadyInCart = true;
```

```
      }
    });

    if (!productAlreadyInCart) {
      cartItems.push({ ...product, count: 1 });
    }
    localStorage.setItem("cartItems", JSON.stringify(cartItems));
    return { cartItems: cartItems };
  });
};

listProducts = () => {
  this.setState(state => {
    if (state.sort !== "") {
      state.products.sort((a, b) =>
        state.sort === "lowestprice"
          ? a.price > b.price
            ? 1
            : -1
          : a.price < b.price
          ? 1
          : -1
      );
    } else {
      state.products.sort((a, b) => (a.id > b.id ? 1 : -1));
    }
    if (state.size !== "") {
      return {
        filteredProducts: state.products.filter(
          a => a.availableSizes.indexOf(state.size.toUpperCase()) >= 0
        )
      };
    }
```

```
    return { filteredProducts: state.products };
  });
};
handleSortChange = e => {
  this.setState({ sort: e.target.value });
  this.listProducts();
};
handleSizeChange = e => {
  this.setState({ size: e.target.value });
  this.listProducts();
};

render() {
  return (
    <div className="container">
      <h1>E-commerce Shopping Cart Application</h1>
      <hr />
      <div className="row">
        <div className="col-md-9">
          <Filter
            count={this.state.filteredProducts.length}
            handleSortChange={this.handleSortChange}
            handleSizeChange={this.handleSizeChange}
          />
          <hr />
          <Products
            products={this.state.filteredProducts}
            handleAddToCart={this.handleAddToCart}
          />
        </div>
        <div className="col-md-3">
          <Basket
            cartItems={this.state.cartItems}
```

```
            handleRemoveFromCart={this.handleRemoveFromCart}
          />
          <Copyright />
        </div>
      </div>
    </div>
  );
 }
}

export default App;
```

5.2 App.css

```
.App {
  text-align: center;
}

.App-logo {
  animation: App-logo-spin infinite 20s linear;
  height: 40vmin;
  pointer-events: none;
}

.App-header {
  background-color: #fb7b6b;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
```

```
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```

Util.js

```
export default {
    formatCurrency: function (num) {
        return '$' + Number(num.toFixed(1)).toLocaleString() + ' ';
    }
}
```

Components

Basket.js

```
import React, { Component } from 'react';
import util from '../util'
export default class Basket extends Component {
    render() {
        const { cartItems } = this.props;
```

```
    return (
      <div className="alert alert-info">
        {cartItems.length === 0
          ? "Basket is empty" :
          <div>You have {cartItems.length} items in the basket. <hr /></div>
        }
        {cartItems.length > 0 &&
          <div>
            <ul style={{ marginLeft: -25 }}>
              {cartItems.map(item => (
                <li key={item.id}>
                  <b>{item.title}</b>
                  <button style={{ float: 'right' }} className="btn btn-danger
btn-xs"
                    onClick={(e) => this.props.handleRemoveFromCart(e,
item)}>X</button>
                  <br />
                  {item.count} X {util.formatCurrency(item.price)}
                </li>))
              }
            </ul>

            <b>Sum: {util.formatCurrency(cartItems.reduce((a, c) => (a +
c.price * c.count), 0))}
            </b>
            <button onClick={() => alert('Todo: Implement checkout page.')}
className="btn btn-primary">checkout</button>
          </div>
        }
      </div>
    )
  }
```

```
}
```

5.3 Filter.js

```
import React, { Component } from 'react';
export default class Products extends Component {

   render() {


       return (
          <div className="row">
             <div className="col-md-4">
                {`${this.props.count} products found.`}
             </div>
             <div className="col-md-4">
                <label>Order by
            <select className="form-control" value={this.props.sort}
onChange={this.props.handleSortChange}>
                      <option value="">Select</option>
                      <option value="lowestprice">Lowest to highest</option>
                      <option value="highestprice">Highest to lowest</option>
                   </select>
                </label>
             </div>
             <div className="col-md-4">
                <label > Filter Size
            <select className="form-control" value={this.props.size}
onChange={this.props.handleSizeChange}>
                      <option value="">ALL</option>
                      <option value="x">XS</option>
                      <option value="s">S</option>
                      <option value="m">M</option>
```

```
                    <option value="l">L</option>
                    <option value="xl">XL</option>
                    <option value="xxl">XXL</option>
                  </select>
                </label>
              </div>
            </div>
        )
    }
}
```

5.4 Product.js

```
import React, { Component } from 'react';
import util from '../util'
export default class Products extends Component {

    render() {
        const productItems = this.props.products.map(product => (
            <div className="col-md-4" key={product.id}>
                <div className="thumbnail text-center">
                    <a
href={`#${product.id}`}onClick={(e)=>this.props.handleAddToCart(e, product)}>
                        <img src={`products/${product.sku}_2.jpg`} alt={product.title} />
                        <p>{product.title}</p>
                    </a>
                    <b>{util.formatCurrency(product.price)}</b>
                    <button className="btn btn-primary"
onClick={(e)=>this.props.handleAddToCart(e, product)}>Add to cart</button>
                </div>
            </div>
        ));
```

```
        return (
            <div className="row">
                {productItems}
            </div>
        )
    }
}
```

5.5 Todo.json

```
{
  "products": [
    {
      "id": 1,
      "sku": 18644119330491312,
      "title": "Sphyn Tie Dye Grey T-Shirt",
      "description": "Sphynx Tie Dye Grey",
      "availableSizes": ["X", "L", "XL", "XXL"],
      "price": 10,
      "isFreeShipping": true
    },

    {
      "id": 2,
      "sku": 11854078013954528,
      "title": "Danger Knife Grey T-Shirt",
      "description": "Danger Knife Grey",
      "availableSizes": ["X", "M", "L"],
      "price": 14.9,
      "isFreeShipping": true
    },

    {
      "id": 3,
```

```
  "sku": 876661122392077,
  "title": "White DGK Script Tee",
  "description": "White DGK Script",
  "availableSizes": ["X", "M", "L"],
  "price": 14.9,
  "isFreeShipping": true
},

{
  "id": 4,
  "sku": 9197907543445677,
  "title": "Born On The Streets  T-Shirt",
  "description": "Born On The Streets",
  "availableSizes": ["XL"],
  "price": 25.9,
  "isFreeShipping": false
},

{
  "id": 5,
  "sku": 10547961582846888,
  "title": "Tso 3D Short Sleeve T-Shirt A",
  "description": "Tso 3D Short Sleeve",
  "availableSizes": ["X", "L", "XL"],
  "price": 10.9,
  "isFreeShipping": false
},

{
  "id": 6,
  "sku": 6090484789343891,
  "title": "Man Tie Dye Cinza Grey T-Shirt",
  "description": "Man Tie Dye Cinza Grey",
```

```
  "availableSizes": ["XL", "XXL"],
  "price": 49.9,
  "isFreeShipping": false
},

{
  "id": 7,
  "sku": 18532669286405342,
  "title": "Crazy Monkey Black T-Shirt",
  "description": "1977 Infantil",
  "availableSizes": ["S"],
  "style": "Preto com listras brancas",
  "price": 22.5,
  "isFreeShipping": true
},

{
  "id": 8,
  "sku": 5619496040738316,
  "title": "Tso 3D Black T-Shirt",
  "description": "",
  "availableSizes": ["XL"],
  "style": "Azul escuro",
  "price": 18.7,
  "isFreeShipping": false
},
{
  "id": 9,
  "sku": 11600983276356165,
  "title": "Crazy Monkey Grey",
  "description": "",
  "availableSizes": ["L", "XL"],
  "price": 134.9,
```

```
    "isFreeShipping": true
   }
  ]
}
```

5.6 Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1, shrink-to-fit=no"
    />
    <meta name="theme-color" content="#000000" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See
https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
```

```
    <link rel="stylesheet"
href="https://bootswatch.com/3/cerulean/bootstrap.min.css">
    <title>Shopping Cart App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

6. CONCLUSION

I have successfully completed the project under the guidance of Mr.Subhash Chandra Gupta, he helped me at every point where I needed him. This project E-Commerce Automotive not only helps users to take know about where price product , products can be sort. According to the price and size according to the customer requirement, user also add the item in the cart.To build this Web-App I have used technologies and languages like ReactJS, bootstrap, javascript, nodeJs, express, mongoose, mongoDB, git, heroku. This project is all about the user to know the different price of an item so that they can compare from the other website. This web-app is scalable to much more functionalities and can handle the request of thousands of customers.

The automotive parts distributors are joining hands in order to serve their customers with highest supply and diversified options. Do not even try to do the maths when it comes to count the number of parts bought and sold every day. It is impossible to channelize every demand through an individual network, unless one wants to be at the back seat of the competition.

7. REFERENCES

1. www.google.com
2. www.wikipedia.com
3. www.w3schools.com
4. https://www.udemy.com/
5. https://www.youtube.com/
6. www.stackoverflow.com