A Mini Project Report
On
ShopCo E-Commerce WebApp

Submitted in partial fulfillment of the requirement of

University of Mumbai for

Internet Programming Mini Project
In
Information Technology
Submitted By

Student Name	Roll No
Vatsal Balar	01
Dhruv Ghori	16
Jenil Jain	19

Department of Information Technology

K.J Somaiya Institute of Engineering and Information Technology

Ayurvihar, Sion, Mumbai -400022

UNIVERSITY OF MUMBAI

2020-2021(Odd SEM)

K. J. Somaiya Institute of Engineering and Information Technology

CERTIFICATE

This is to certify that the requirements for the report entitled 'Name of the Project' have been successfully completed by the following students:

Student Name	Roll No
Vatsal Balar	01
Dhruv Ghori	16
Jenil Jain	19

In partial fulfillment of Internet Programming mini Project in the Department of Information Technology,**K.J Somaiya Institute of Engineering and Information** during the Academic Year 2020 – 2021.

Date:_____

_____

(Prof. Nasim Shah)

Guide & Project Coordinator

_____

(Dr. Radhika Kotecha)

Head of Department

Table of Content

AIM

Costs are escalating for businesses that only deal with their markets through physical channels like retail outlets and sales people. Organisations need to find new ways to deliver products and services while maintaining a direct link with customers – this is where the Internet has had a major impact.

E-commerce websites enable the distribution, buying, selling, marketing, and servicing of products or services over the Internet and helps to reduce costs while reaching a wider market.

Many businesses opt for an off-the-shelf E-commerce website to meet their needs. But they soon find themselves restricted by the inherent limitations of such an approach. This inflexibility prevents the solution from being adapted to the very processes that give your business its competitive edge. While the inability to integrate with other applications, that your business runs, can lead to inefficiencies and errors.

INTRODUCTION

An E-Commerce portal which will allow formal and informal merchants in developing countries to advertise and sell their goods on the internet. This would permit rural communities to make their wares available to the rest of the world via the World Wide Web. The objective of this project is to create an e-commerce web portal with a content management system which would allow product information to be updated securely using a mobile device. The web portal will have an online interface in the form of an e-commerce website that will allow users to buy goods from the merchants.

ABSTRACT

In this era of internet, e-commerce is growing by leaps and bounds keeping the growth of brick-and-mortar businesses in the dust. In many cases, brick-and-mortar businesses are resorting to having a counterpart which is internet or e-commerce driven. People in the developed world and a growing number of people in the developing world now use e-commerce websites on a daily basis to make their everyday purchases. Still the proliferation of e-commerce in the under-developed world is not that great and there is a lot to desire for. This paper outlines different aspects of developing an e-commerce website and the optimum solution to the challenges involved in developing one. It consists of the planning process, which starts with determining the use case, domain modeling and architectural pattern of the web application. The entire development process is primarily divided into two parts: the front-end development and the back end development. The database design is also discussed with an emphasis on its relational connectivity. This no-nonsense method of developing an e-commerce website can be easily replicated and followed in developing e-commerce websites in the developing and under-developed countries where computing resources are scarce and expensive because of their socio-economic condition.

The basis of this project is to create a system where the customer can receive products easily and without any inconvenience. The software and techniques we used to create this forum: firebase, HTML, CSS, ReactJS, NODE JS, Bootstrap, ExpressJS and Chrome browser for testing purposes. This website forum is responsive therefore it can accessible in any wearable devices.

PROPOSED SYSTEM

This System provides variety of functionality to the user i.e. customer. The functionality includes

- Promotion and discount code tools
- An easy-to-use checkout
- Search engine optimized code and layout
- Reporting tools and custom report features
- Email marketing features or integration
- Multiple payment options (Credit card, Debit Card, etc.)
- The ability to scale and add new features

EXISTING SYSTEM

E-Commerce has become one of the most popular mediums of transactions in the recent years. While it does offer quite a lot of benefits to both buyers and sellers, it is not totally free from disadvantages. By having an idea of these limitations, we can address them and come up with a solution

LIMITATIONS OF EXISTING SYSTEM

By analyzing the existing system, some of its drawbacks are listed.

There can be lack of system security, reliability or standards owing to poor implementation of e-commerce.

- The software development industry is still evolving and keeps changing rapidly.
- In many countries, network bandwidth might cause an issue.
- Special types of web servers or other software might be required by the vendor, setting the e-commerce environment apart from network servers.
- Sometimes, it becomes difficult to integrate an e-commerce software or website with existing applications or databases.
- There could be software/hardware compatibility issues, as some e-commerce software may be incompatible with some operating system or any other component.

REQUIREMENT ANALYSIS

There are mainly 7 modules in this website namely like Home, Orders, About Us, Cancellation and Refund Policy, Shipping Policy, Privacy Policy and User Registration.

Home: This module gives an overview of the website. It consists of the interface which gives user the knowledge of latest services available for booking.
Orders: This Page gives the order history of the user along with the date and time when the orders were placed.
About Us: This Page describes the motive of this application and what services are offered.
Cancellations and Refund Policy: This page where a user can check the cancellation and refunds policy of the website.
Shipping Policy: Here a user can find how we ship the products and how their orders are packaged, how they can track their orders and also check the expected time for delivey.
Privacy Policy: Here user can find how we use user's data, what information do we collect from the customers.
User Registration: It is a registration form with Proper Validation and whatever user enters those values. Also there is a Login Page for existing users. As soon as the user Logs in the Logout option is set.

RESPONSIVE WEB DESIGN FRAMEWORK

What is a framework?

A framework is a standardized set of concepts, practices and criteria for dealing with a common type of problem, which can be used as a reference to help us approach and resolve new problems of a similar nature.

In the world of web design, to give a more straightforward definition, a framework is defined as a package made up of a structure of files and folders of standardized code (HTML, CSS, JS documents etc.) which can be used to support the development of websites, as a basis to start building a site.

Most websites share a very similar (not to say identical) structure. The aim of frameworks is to provide a common structure so that developers don't have to redo it from scratch and can reuse the code provided. In this way, frameworks allow us to cut out much of the work and save a lot of time.

ReactJS

ReactJS is JavaScript library used for building reusable UI components. According to React official documentation, following is the definition −

React is a library for building composable user interfaces. It encourages the creation of reusable UI components, which present data that changes over time. Lots of people use React as the V in MVC. React abstracts away the DOM from you, offering a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using React Native. React implements one-way reactive data flow, which reduces the boilerplate and is easier to reason about than traditional data binding.

Bootstrap

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive pre built components, and powerful plugins built on jQuery.

NodeJS

Node.js is an open-source server side runtime environment built on Chrome's V8 JavaScript engine. It provides an event driven, non-blocking (asynchronous) I/O and cross-platform runtime environment for building highly scalable server-side application using JavaScript.
Node.js can be used to build different types of applications such as command line application, web application, real-time chat application, REST API server etc. However, it is mainly used to build network programs like web servers, similar to PHP, Java, or ASP.NET.
Node.js was written and introduced by Ryan Dahl in 2009.

ExpressJS

Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some of the core features of Express framework –

- Allows to set up middlewares to respond to HTTP Requests.
- Defines a routing table which is used to perform different actions based on HTTP Method and URL.
- Allows to dynamically render HTML Pages based on passing arguments to templates.

TYPES OF CSS

CSS can be implemented in three ways:

- In a separate file (external)
- At the top of a web page document (internal)
- Right next to the text it decorates (inline)

External style sheets are separate files full of CSS instructions (with the file extension .css). When any web page includes an external style sheet, its look and feel will be controlled by this CSS file (unless you decide to override a style using one of these next two types). This is how you change a whole website at once. And that's perfect if you want to keep up with the latest fashion in web pages without rewriting every page!

Internal styles are placed at the top of each web page document, before any of the content is listed. This is the next best thing to external, because they're easy to find, yet allow you to 'override' an external style sheet -- for that special page that wants to be a nonconformist!

Inline styles are placed right where you need them, next to the text or graphic you wish to decorate. You can insert inline styles anywhere in the middle of your HTML code, giving you real freedom to specify each web page element. On the other hand, this can make maintaining web pages a real chore!

CSS CODE

External CSS:

```
.Navbar {
background: white;
box-shadow: 0px 0px 4px rgba(0, 0, 0, 0.25);
font-family: Montserrat;
font-style: normal;
font-weight: 600;
color: #444444;
}

.navbar-light .navbar-nav .nav-link {
color: #444444;
}
.Nav-mob-item {
font-weight: 700;
margin: 0.2rem 0;
}
.dropdown-divider {
border-top: 1px solid black;
}
.navbar-toggler:focus,
```

```
.navbar-toggler:active,
.navbar-toggler-icon:focus {
outline: none;
box-shadow: none;
}

.navbar-light .navbar-toggler {
color: rgba(0, 0, 0, 0.5);
border-color: rgba(0, 0, 0, 0.1);
margin-right: 0.5rem;
}

.header {
height: 60px;
display: flex;
align-items: center;
background-color: #131921;
position: sticky;
top: 0;
z-index: 100;
}

.header__logo {
font-family: Montserrat;
font-style: normal;
font-weight: 500;
font-size: 2rem;
color: #ffffff;
margin: 0 1rem;
}
.header__search {
display: flex;
flex: 1;
align-items: center;
border-radius: 24px;
}

.header__searchInput {
height: 12px;
padding: 10px;
border: none;
width: 100%;
}

.header__searchIcon {
padding: 5px;
height: 22px !important;
background-color: #cd9042;
}
```

```
.header__nav {
display: flex;
justify-content: space-evenly;
}

.header__option {
display: flex;
flex-direction: column;
margin: 0 10px;
color: white;
cursor: pointer;
}

.header__optionLineOne {
font-size: 10px;
}

.header__optionLineTwo {
font-size: 13px;
font-weight: 800;
}

.header__optionBasket {
display: flex;
align-items: center;
color: white;
}

.header__basketCount {
margin: 0 10px;
}

.caraousel-img {
  height: 700px;
  object-fit: cover;
}


.home {
  margin-left: auto;
  margin-right: auto;
  max-width: 1500px;
}
```

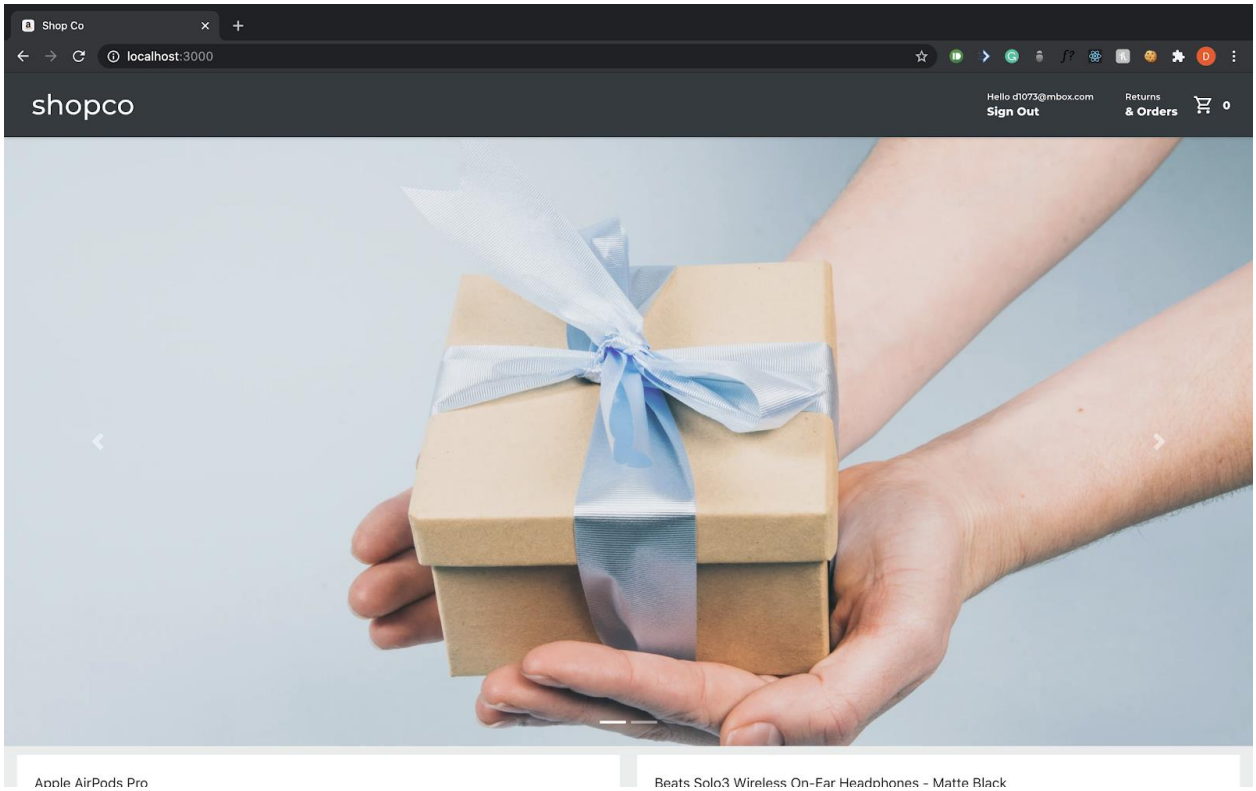
```
.home__row {
  z-index: 1;
  margin-left: 5px;
  margin-right: 5px;
}

.home__image {
  width: 100%;
  z-index: -1;
  margin-bottom: -250px;
  mask-image: linear-gradient(to bottom, rgba(0, 0, 0, 1), rgba(0, 0, 0, 0));
}
.product {
  display: flex;
  flex-direction: column;

  align-items: center;
  justify-content: flex-end;
  margin: 10px;
  padding: 20px;
  width: 100%;
  max-height: 500px;
  min-width: 100px;

  background-color: white;
  z-index: 1;
}

.product-mob {
  display: flex;
```

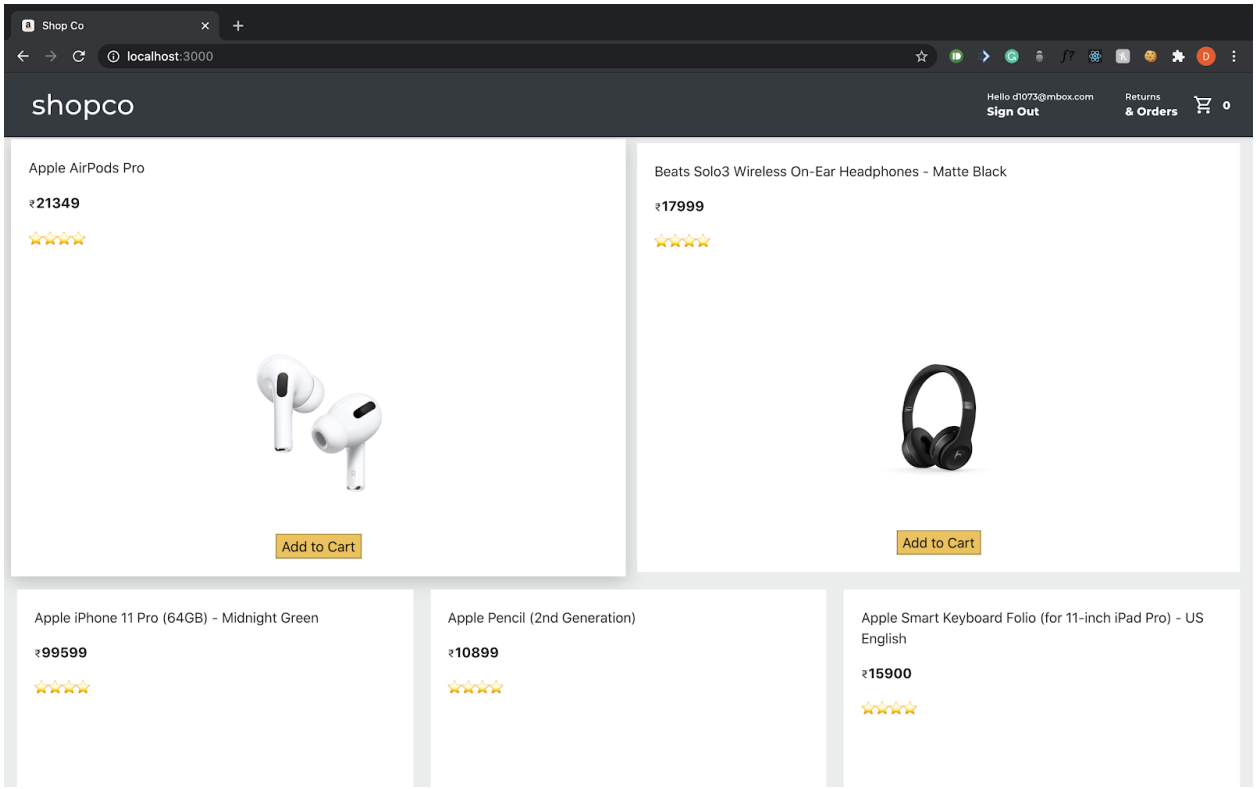
```
  flex-direction: column;

  align-items: center;
  justify-content: flex-end;
  margin: 10px;
  padding: 20px;
  max-height: 500px;
  min-width: 100px;

  background-color: white;
  z-index: 1;
}

.product:hover {
  transform: scale(1.02);
  box-shadow: 0px 5px 20px rgba(0, 0, 0, 0.2);
  border-color: rgba(255, 255, 255, 0.6);
  transition: transform 0.5s ease-in-out;
}

.product > img {
  max-height: 200px;
  width: 100%;
  object-fit: contain;
  margin-bottom: 15px;
}

.product-mob > img {
  max-height: 200px;
  width: 100%;
```

```
  object-fit: contain;
  margin-bottom: 15px;
}

.product__img:hover {
  transform: rotateY(180deg) !important;
}

.product__info {
  height: 200px;
  width: 100%;
  text-align: left;
}

.product__price {
  margin-top: 5px;
}

.product__rating {
  display: flex;
}

.product > button {
  background: #f0c14b;
  border: 1px solid;
  margin-top: 10px;
  border-color: #a88734 #9c7e31 #846a29;
  color: #111;
  cursor: pointer;
  position: relative;
```

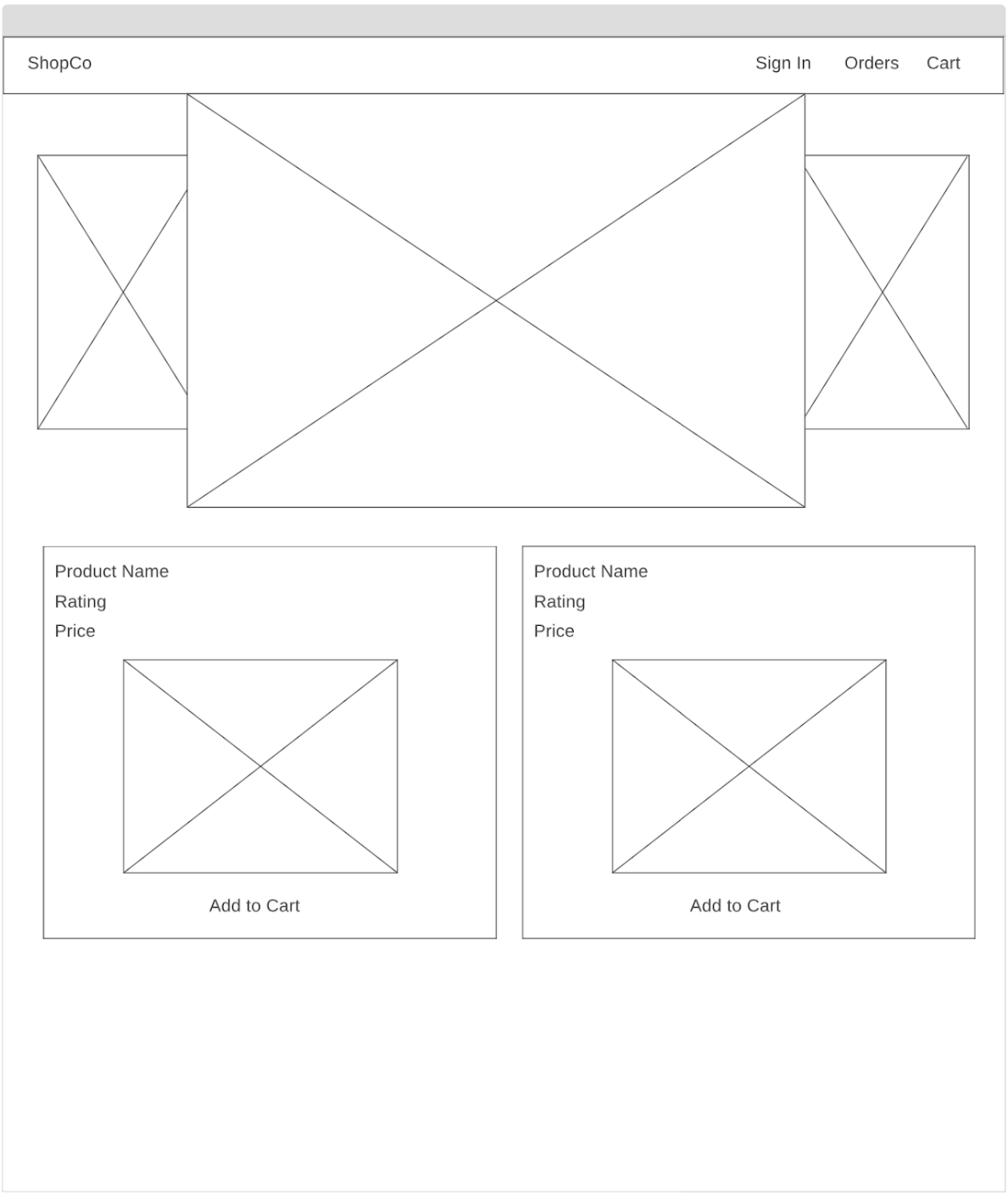
```
  left: 50%;
  transform: translateX(-50%);
}

.product-mob > button {
  background: #f0c14b;
  border: 1px solid;
  margin-top: 10px;
  border-color: #a88734 #9c7e31 #846a29;
  color: #111;
  cursor: pointer;
}
```

Output:




Shop Co
localhost:3000
shopco
Hello d1073@mbox.com
Sign Out
Returns
& Orders
0
Apple AirPods Pro
₹21349
Add to Cart
Beats Solo3 Wireless On-Ear Headphones - Matte Black
₹17999
Add to Cart
Apple iPhone 11 Pro (64GB) - Midnight Green
₹99599
Apple Pencil (2nd Generation)
₹10899
Apple Smart Keyboard Folio (for 11-inch iPad Pro) - US English
₹15900

SKELETON OF WEBSITE



RESPONSIVE DESIGN CODE

Code:

Header.js

```
<Navbar sticky='top' className='Navbar bg-dark' expand='lg'>
<Navbar.Toggle
style={{ border: "None" }}
aria-controls='basic-navbar-nav'
/>
<Navbar.Brand className='Navbar-brand' href='/'>
<p className='header__logo'>shopco</p>
</Navbar.Brand>

<Navbar.Collapse className='ml-auto' id='basic-navbar-nav'>
<Nav className='ml-auto'>
<Nav.Link className='Nav-item d-none d-lg-block'>
<Link to={!user && "/login"}>
<div onClick={handleAuthentication} className='header__option'>
<span className='header__optionLineOne'>
Hello {user ? user?.email : "Guest"}
</span>
<span className='header__optionLineTwo'>
{user ? "Sign Out" : "Sign in"}
</span>
</div>
</Link>
</Nav.Link>
<Nav.Link className='Nav-item d-none d-lg-block'>
<Link to="/orders">
<div className='header__option'>
<span className='header__optionLineOne'>Returns</span>
<span className='header__optionLineTwo'>& Orders</span>
</div>
</Link>
</Nav.Link>

<Nav.Link className='Nav-mob-item d-lg-none d-xl-none'>
<Link to={!user && "/login"}>
<div onClick={handleAuthentication} className='header__option'>
<span className='header__optionLineOne'>
Hello {user ? user?.email : "Guest"}
</span>
<span className='header__optionLineTwo'>
{user ? "Sign Out" : "Sign in"}
</span>
</div>
</Link>
```

```
</Nav.Link>
<NavDropdown.Divider />
<Nav.Link className='Nav-mob-item d-lg-none d-xl-none'>
<Link to="/orders">
<div className='header__option'>
<span className='header__optionLineOne'>Returns</span>
<span className='header__optionLineTwo'>& Orders</span>
</div>
</Link>
</Nav.Link>
</Nav>
</Navbar.Collapse>
<Link to='/checkout'>
<div className='header__optionBasket'>
<ShoppingCartOutlinedIcon className='header__optionBasketIcon' />
<span className='header__optionLineTwo header__basketCount'>
{basket?.length}
</span>
</div>
</Link>
</Navbar>
```

Product.js

```
{/* desk-product */}

    <div className='product d-none d-sm-none d-md-none d-lg-block'>

      <div className='product__info'>

        <p>{title} </p>

        <p className='product__price'>

        <small>₹</small>

        <strong>{price}</strong>

        </p>

        <div className='product__rating'>

          {Array(rating)

           .fill()

           .map((_, i) => (

             <p> ⋆ </p>

           ))}

        </div>
```

```
  </div>
  <img className='product__img' src={image} />
  <button onClick={addToCart}>Add to Cart</button>
 </div>
 {/* mob-product */}
 <div className='product-mob d-lg-none d-xl-none'>
  <div className='product__info'>
   <p>{title} </p>
   <p className='product__price'>
   <small>₹</small>
   <strong>{price}</strong>
   </p>
   <div className='product__rating'>
    {Array(rating)
     .fill()
     .map((_, i) => (
      <p>⋆ </p>
     ))}
   </div>
  </div>
  <img className='product__img' src={image} />
  <button onClick={addToCart}>Add to Cart</button>
 </div>
```

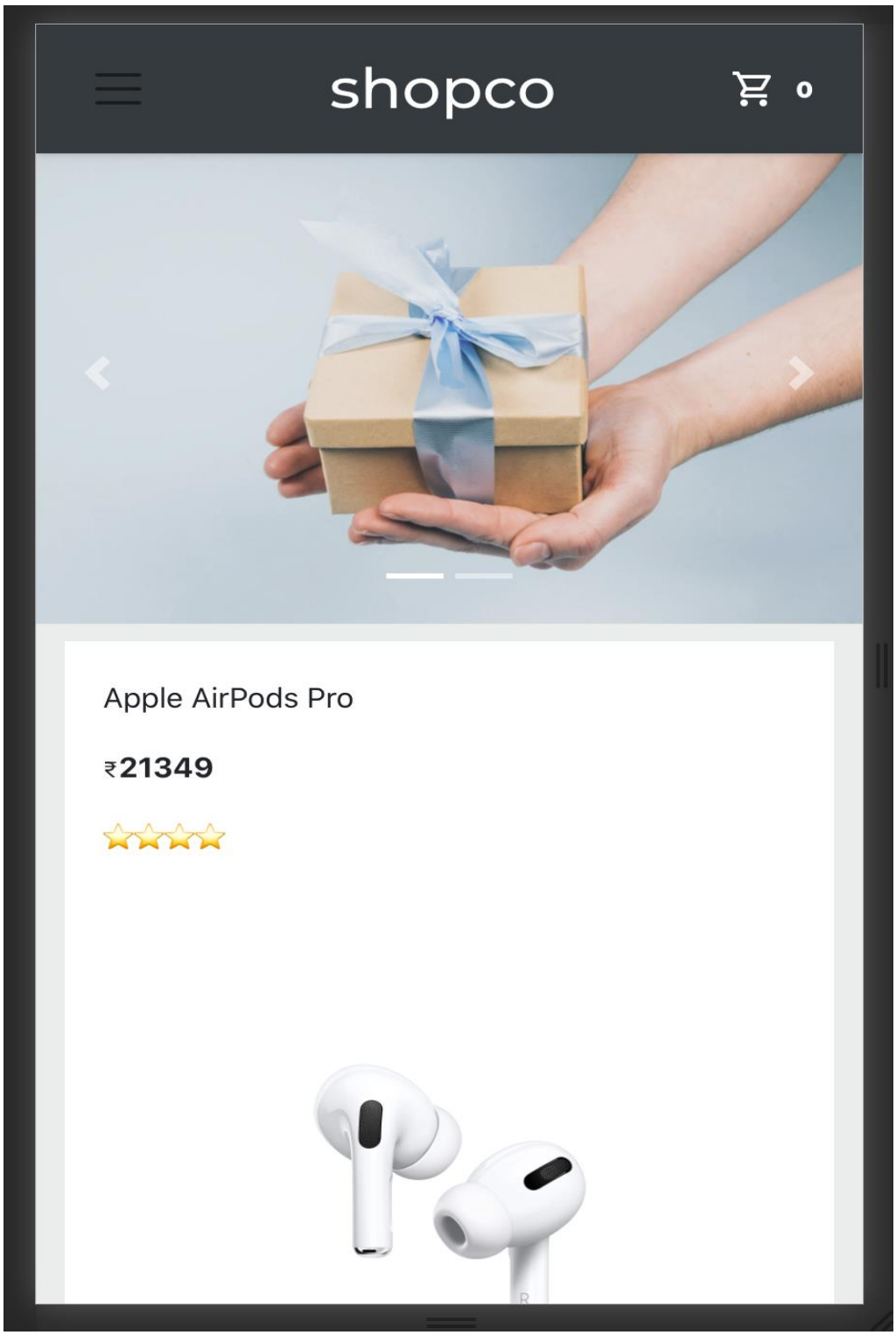
Footer.js

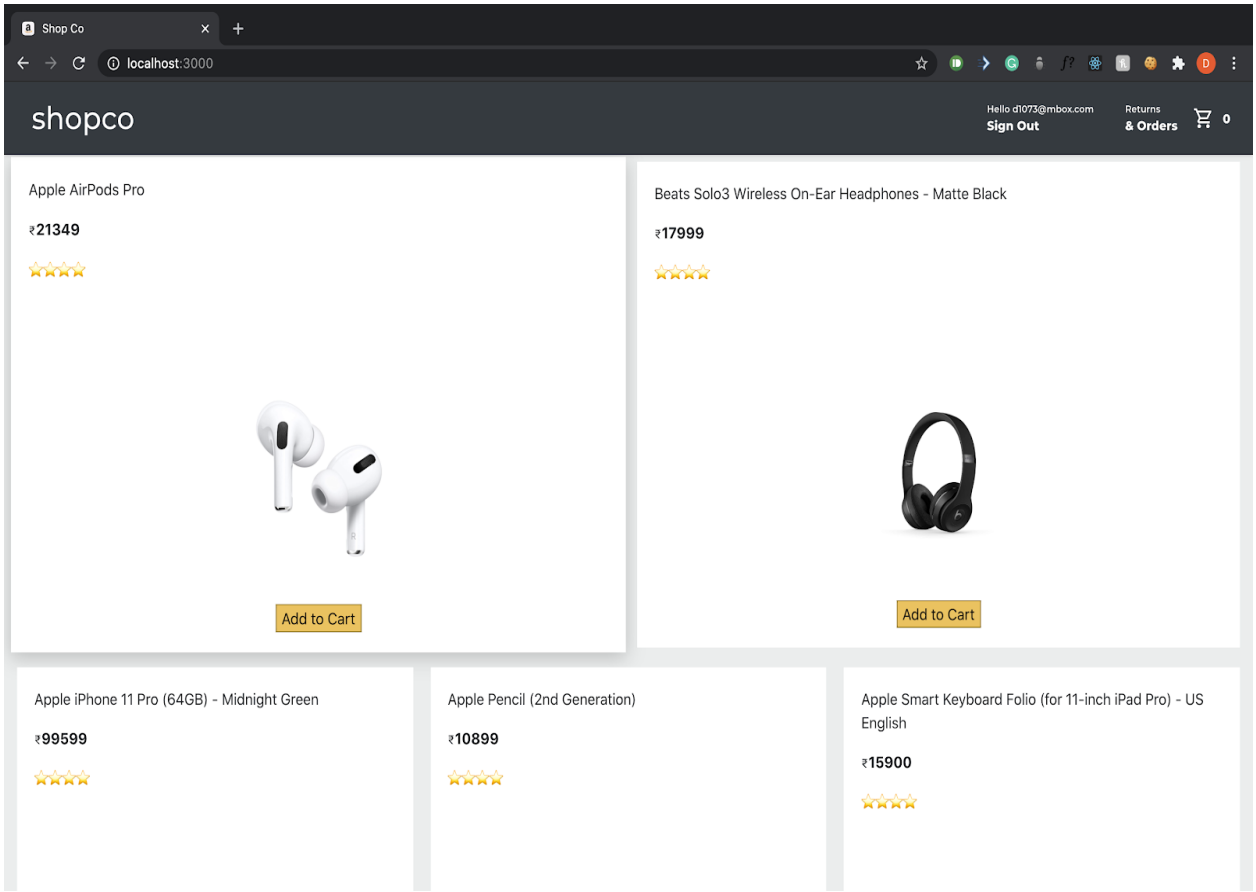
```
<div className='bg-dark px-5 py-4'>
<Container fluid>
<Row xs={1} sm={1} lg={3}>
<Col className='table-head-text'>
COMPANY
<ul className='table-content-text'>
```

```
<li className='text-white text-decoration-none'>
<Link
to='/about-us'
className='text-white text-decoration-none'
>
About Us
</Link>
</li>
</ul>
</Col>

<Col className='table-head-text'>
RESOURCES
<ul className='table-content-text'>
<li>
<Link
to='/cancellation-and-refund'
className='text-white text-decoration-none'
>
Cancellation and Refund Policy
</Link>
</li>
<li>
<Link to='/shipping-policy' className='text-white text-decoration-none'>
Shipping Policy
</Link>
</li>
<li>
<Link to='/privacy-policy' className='text-white text-decoration-none'>
Privacy policy
</Link>
</li>
</ul>
</Col>

<Col className='table-head-text'>
CONNECT WITH US
<ul className='table-content-text'>
<li>demo@demo.com</li>
<li>
Somaiya Ayurvihar Complex Eastern Express Highway, Sion East, Mumbai, Maharashtra
400022
</li>
</ul>
</Col>
</Row>
</Container>
<p className='copyright-text mb-5'>
shop.co 2020 - All rights reserved.
</p>
```

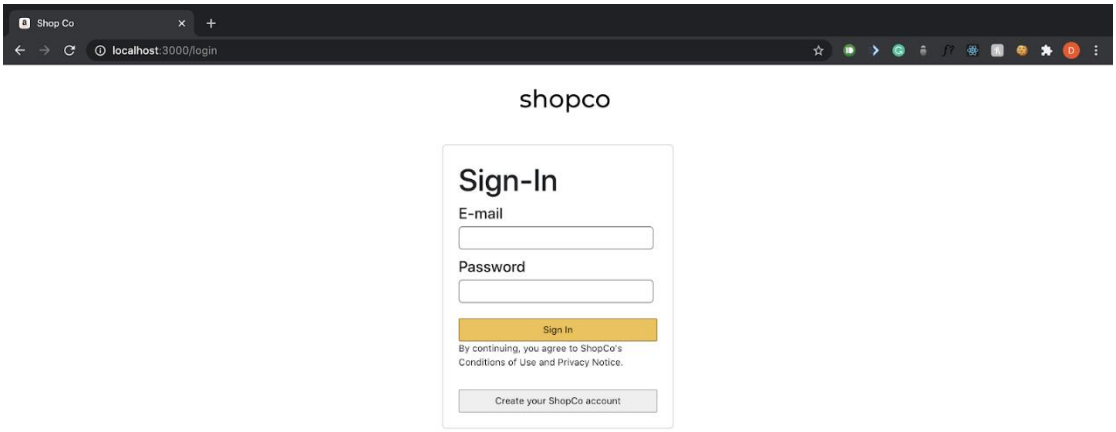
Output:-

Responsive Front Page:-

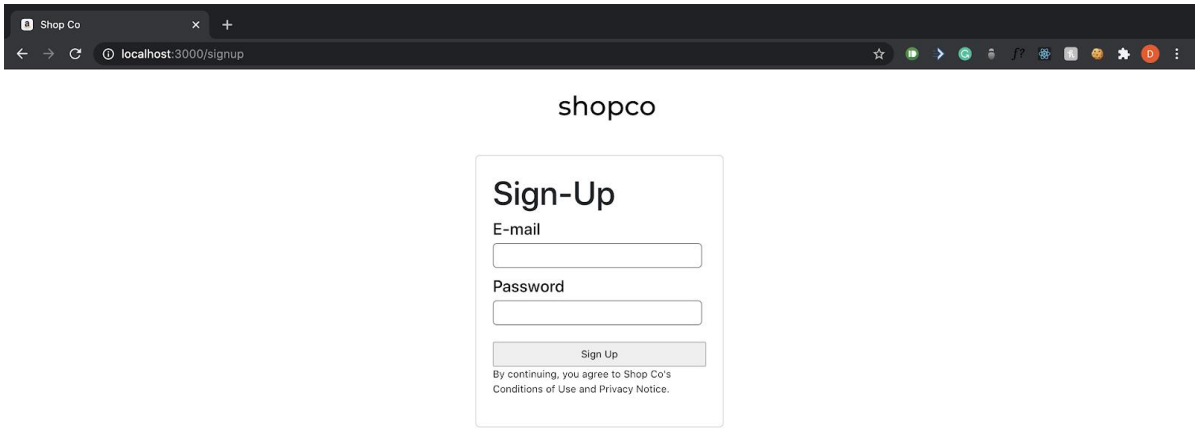



Shop Co
localhost:3000
shopco
Hello d1073@mbox.com
Sign Out
Returns
& Orders
0
Apple AirPods Pro
₹21349
Add to Cart
Beats Solo3 Wireless On-Ear Headphones - Matte Black
₹17999
Add to Cart
Apple iPhone 11 Pro (64GB) - Midnight Green
₹99599
Apple Pencil (2nd Generation)
₹10899
Apple Smart Keyboard Folio (for 11-inch iPad Pro) - US English
₹15900

Login Page:-



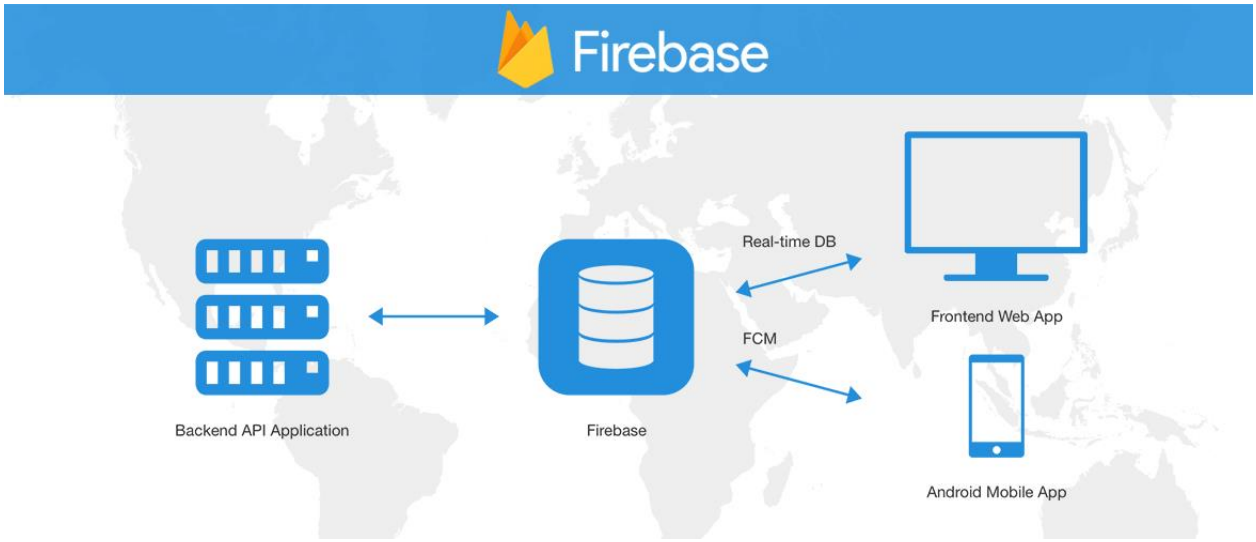
Sign Up Page:-



FireBase Cloud Firestore

CONNECT DATABASE

Firebase is a backend platform for building Web, Android and IOS applications. It offers real time database, different APIs, multiple authentication types and hosting platform. This is an introductory tutorial, which covers the basics of the Firebase platform and explains how to deal with its various components and sub-components.



Database: Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. Like Firebase Realtime Database, it keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud Platform products, including Cloud Functions.
Collection: A group of database documents can be called a collection. The RDBMS equivalent to a collection is a table. The entire collection exists within a single database. There are no schemas when it comes to collections. Inside the collection, various documents can have varied fields, but mostly the documents within a collection are meant for the same purpose or for serving the same end goal.
Document: A set of key–value pairs can be designated as a document. Documents are associated with dynamic schemas. The benefit of having dynamic schemas is that a document in a single collection does not have to possess the same structure or fields. Also, the common fields in a collection's document can have varied types of dat.

Advantages of using firebase cloud firestore:

- Combines the Benefits of Firebase with Google Cloud Platform.
- Cloud or Serverless Solution.
- Offers Excellent Data Handling Capabilities.

- Designed to Scale.
- Promises Robust Security.
- Enables Offline Support.
- Cost-Effective Pricing.

Code:
firebase.js :

```
import firebase from "firebase";

// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
    apiKey: "AIzaSyBeAyDxNLnHZMHnS2_q4a5rlEWrCGvClW4",
    authDomain: "clone-ef42d.firebaseapp.com",
    databaseURL: "https://clone-ef42d.firebaseio.com",
    projectId: "clone-ef42d",
    storageBucket: "clone-ef42d.appspot.com",
    messagingSenderId: "984665646017",
    appId: "1:984665646017:web:ed3df27184ad495f03c540",
    measurementId: "G-3BMNY5KFDP"
  };

const firebaseApp = firebase.initializeApp(firebaseConfig);

const db = firebaseApp.firestore();
const auth = firebase.auth();

export { db, auth };
```

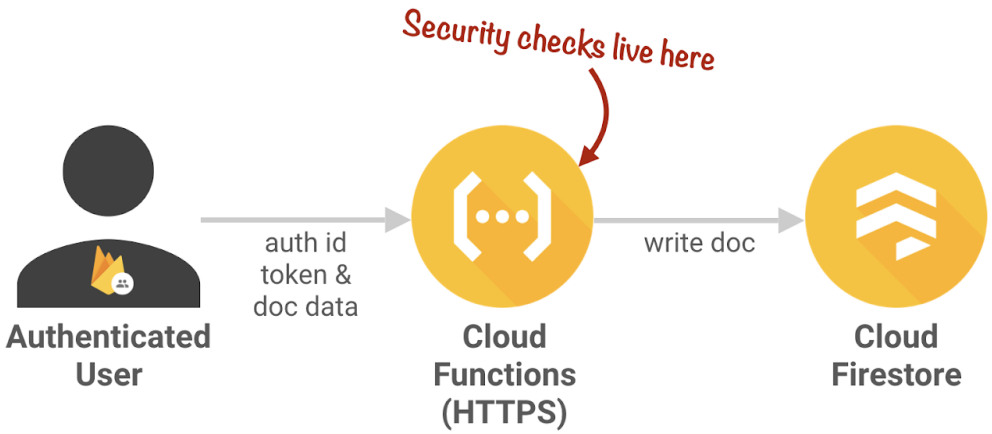
Payment.js :

```
const handleSubmit = async (event) => {
event.preventDefault();
// manage payment
setProcessing(true);
const payload = await stripe
.confirmCardPayment(clientSecret, {
payment_method: {
card: elements.getElement(CardElement),
},
})
.then(({ paymentIntent }) => {
// paymentIntent= payment confirmation
db.collection("users")
.doc(user?.uid)
.collection("orders")
.doc(paymentIntent.id)
.set({
basket: basket,
```

```
amount: paymentIntent.amount,
created: paymentIntent.created,
});
setSucceeded(true);
setError(null);
setProcessing(false);
dispatch({
type: "EMPTY_CART",
});
history.replace("/orders");
});
};
const handleChange = (event) => {
// Listen for changes in CardElement
// and display any errors as the customer types in their card details
setDisabled(event.empty);
setError(event.error ? event.error.message : "");
};
```

WEB SERVICES

Firebase Authentication



Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app, offering a more engaging experience across different platforms and between apps. It supports authentication using passwords, popular federated identity providers such as Google; Facebook; and Twitter, and more, making it easy for users to access your content and get into your apps quickly and securely.

Key features:

- **Integrates tightly** with other Firebase features.
- **Uses industry standards** such as OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.
- **Offers two development options** either FirebaseUI as a complete drop-in authentication solution or the Firebase Authentication SDK to manually integrate one or several sign-in methods into your app.
- **Provides secure authentication** that makes sign in easy for your users using their Google account, which they already use with Gmail, Google Play, and other Google services. It also supports authentication using passwords and popular federated identity providers such as Facebook and Twitter.
- **Enables a seamless app experience across devices and into your website**, securely from a one-time consent. This will help keep your users engaged, no matter what device they pick up or sit down at.

- **Connects users securely with Google services**. Share with Google contacts, save files to Drive, add events to Calendar, and more.

Code:

```
import React, { useState } from "react";
import "./Login.css";
import { Link, useHistory } from "react-router-dom";
import { auth } from "./firebase";

function Login() {
const history = useHistory();

const [email, setEmail] = useState("");
const [password, setPassword] = useState("");

const signIn = (e) => {
e.preventDefault();


auth
.signInWithEmailAndPassword(email, password)
.then((auth) => {
if (auth) {
history.push("/");
}
})
.catch((error) => alert(error.message));
//firebase login
};


return (
<div className='login'>
<Link to='/' style={{textDecoration: 'None',}}>
<p className='login__logo'>shopco</p>
</Link>

<div class='login__container'>
<h1>Sign-In</h1>
<form>
<h5>E-mail</h5>
<input
type='email'
value={email}
onChange={(e) => setEmail(e.target.value)}
/>

<h5>Password</h5>
<input
```

```
type='password'
value={password}
onChange={(e) => setPassword(e.target.value)}
/>

<button
type='submit'
onClick={signIn}
className='login__signInButton'
>
Sign In
</button>
</form>
<p>
By continuing, you agree to ShopCo's Conditions of Use and
Privacy Notice.
</p>
<Link to='/signup'>
<button className='login__registerButton'>
Create your ShopCo account
</button>
</Link>
</div>
</div>
);
}

export default Login;
```

Code:

reducer.js

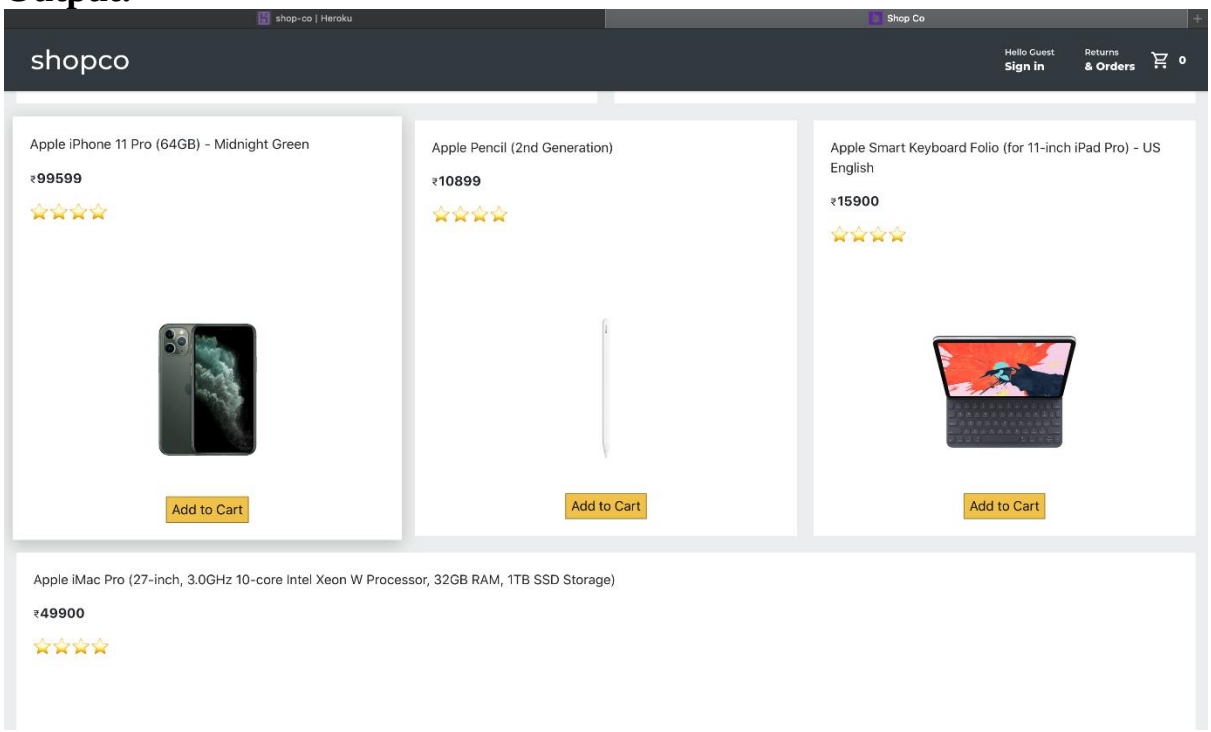
```
export const initialState = {
basket: [],
null: null,
};

// Selector
export const getCartTotal = (basket) =>
basket?.reduce((amount, item) => item.price + amount, 0);

const reducer = (state, action) => {
// console.log(action);
switch (action.type) {
case "ADD_TO_CART":
return {
...state,
basket: [...state.basket, action.item],
};
case 'EMPTY_CART':
return {
...state,
basket: []
}
case "REMOVE_FROM_CART":
const index = state.basket.findIndex(
(basketItem) => (basketItem.id === action.id)
);
let newBasket = [...state.basket];

if (index >= 0) {
newBasket.splice(index, 1);
} else {
console.warn(
`Can't remove the product (id: ${action.id}) as its not in the cart! `
);
}

return {
...state,
basket: newBasket,
};
case "SET_USER":
return {
...state,
user: action.user,
};
```

```
default:
return state;
}
};

export default reducer;
```

StateProvider.js

```
import React, { createContext, useContext, useReducer } from "react";

// Prepared the datalayer
export const StateContext = createContext();

// Wrap our app and provide Data layer
export const StateProvider = ({ reducer, initialState, children }) => (
<StateContext.Provider value={useReducer(reducer, initialState)}>
{children}
</StateContext.Provider>
);

// Pull information from the data layer
export const useStateValue = () => useContext(StateContext);
```

Output:



SECURITY

SSL certificates are what enable websites to move from HTTP to HTTPS, which is more secure. An SSL certificate is a data file hosted in a website's origin server. SSL certificates make SSL/TLS encryption possible, and they contain the website's public key and the website's identity, along with related information. Devices attempting to communicate with the origin server will reference this file to obtain the public key and verify the server's identity. The private key is kept secret and secure.

What is SSL?

SSL, more commonly called TLS, is a protocol for encrypting Internet traffic and verifying server identity. Any website with an HTTPS web address uses SSL/TLS. See What is SSL? and What is TLS? to learn more.

What information does an SSL certificate contain?

SSL certificates include:

- The domain name that the certificate was issued for
- Which person, organization, or device it was issued to
- Which certificate authority issued it
- The certificate authority's digital signature
- Associated subdomains
- Issue date of the certificate
- Expiration date of the certificate
- The public key (the private key is kept secret)

The public and private keys used for SSL are essentially long strings of characters used for encrypting and decrypting data. Data encrypted with the public key can only be decrypted with the private key, and vice versa.

Why do websites need an SSL certificate?

A website needs an SSL certificate in order to keep user data secure, verify ownership of the website, prevent attackers from creating a fake version of the site, and gain user trust.

Encryption: SSL/TLS encryption is possible because of the public-private key pairing that SSL certificates facilitate. Clients (such as web browsers) get the public key necessary to open a TLS connection from a server's SSL certificate.

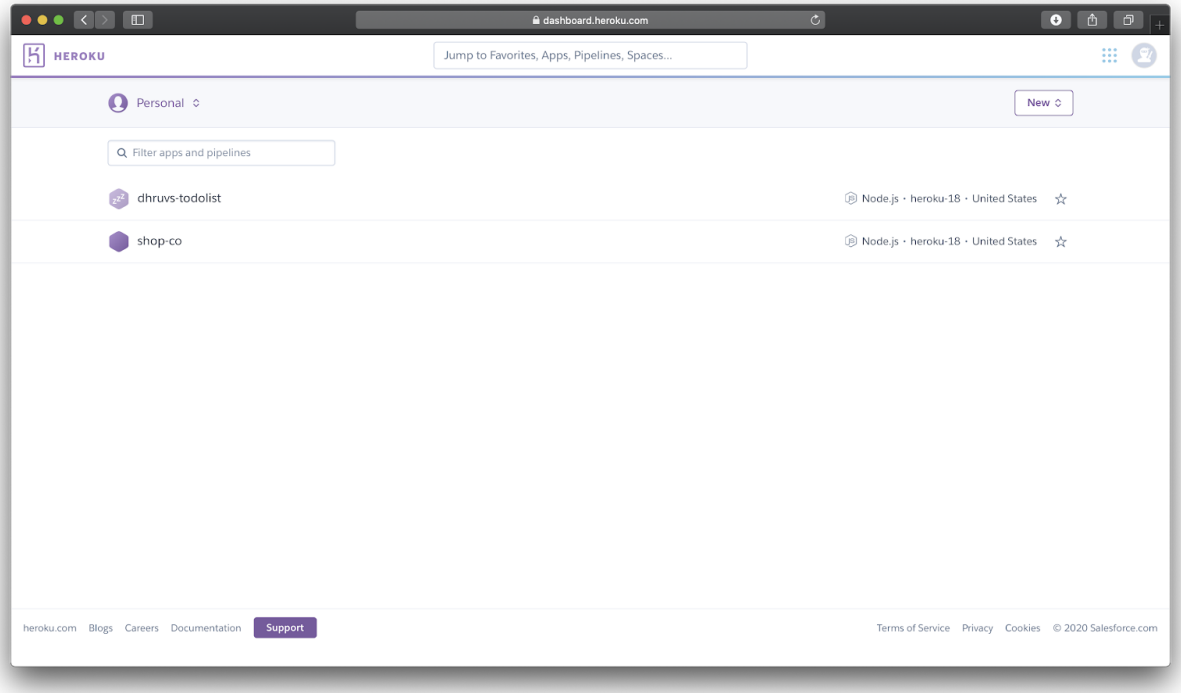
Authentication: SSL certificates verify that a client is talking to the correct server that actually owns the domain. This helps prevent domain spoofing and other kinds of attacks.

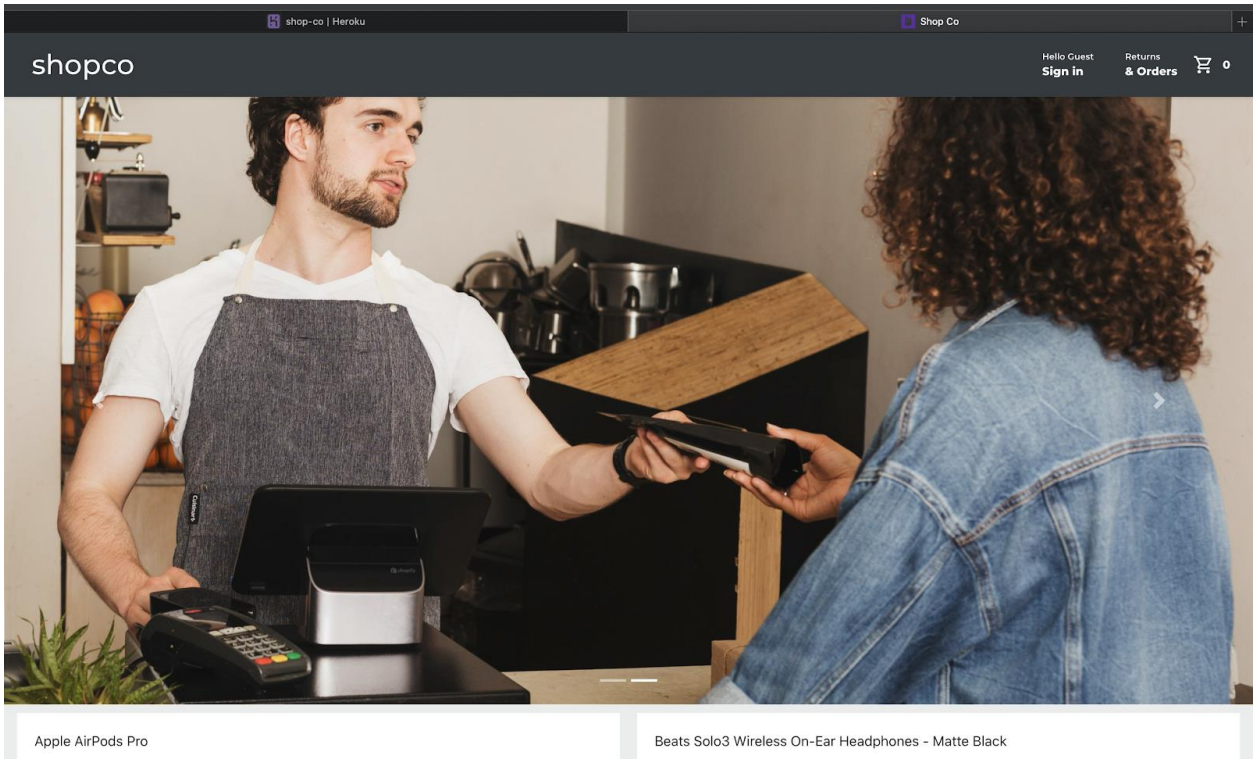
HTTPS: Most crucially for businesses, an SSL certificate is necessary for an HTTPS web address. HTTPS is the secure form of HTTP, and HTTPS websites are websites that have their traffic encrypted by SSL/TLS.

In addition to securing user data in transit, HTTPS makes sites more trustworthy from a user's perspective. Many users won't notice the difference between an http:// and an https:// web address, but most browsers have started tagging HTTP sites as "not secure" in more noticeable ways, attempting to provide incentive for switching to HTTPS and increasing security.

WEB HOSTING

Heroku's free cloud services begins with the apps - apps which can be deployed to dynos - our lightweight Linux containers that are at the heart of the Heroku platform. When you sign up with Heroku, you automatically get a pool of free dyno hours to use for your apps. When your app runs, it consumes dyno hours. When it idles (automatically, after 30 minutes of inactivity), or when you scale it down, your app will stop consuming dyno hours.You can deploy your free app.





CONCLUSION

E-Commerce is not just about conducting business transactions via the Internet. Its impact will be far-reaching, and more prominent then we know currently. This is because the revolution in information technology is happening simultaneously with other developments, especially the globalization of the business. The new age of global e-commerce is creating entirely new economy and that will tremendously change our lives, will reshape the competition in various industries, and alter the economy globally. As companies are gaining high profits, more and more other companies are developing their websites to increase their profits. Since more businesses are being held online resulting in high economy development and emergence of a more innovative and advanced technology. Due to this we learnt various latest Technologies like Nodejs, JavaScript. Also various new web frameworks.

REFERENCES

[1] https://www.w3schools.com/
[2] https://www.tutorialspoint.com
[3] https://www.guru99.com/
[4] https://github.com/
[5] https://www.javatpoint.com
[6] https://www.udemy.com/
[7] https://stackoverflow.com/
[8] https://portswigger.net/
[9] https://webgradients.com/
[10] https://www.geeksforgeeks.org/

ACKNOWLEDGEMENT

The work has successfully completed with the contribution of all the people. It gives us immense pleasure to give our heartily welcome to various people directly or indirectly related to our present project work. We sincerely believe that we were not enough to have come across a group of human whose mere mode of living can inspire someone to work. Our project guide Prof. Nasim Shah, Asst. Prof in Department of Information Technology. She not only encouraged us to pursue our work but also initiated the self-belief and confidences, which helps us to take on various trivial, serious and inevitable hassles of life.