

Chapter 5

IMPLEMENTATION

Implementation is the process of defining how the system should be built, ensuring that it is operational and meets quality standards. It is a systematic and structured approach for effectively integrating a software-based service or component into the requirements of end users.

5.1 Front-end and back-end used

The front-end is everything involved with what the user sees. The back-end, or the "server-side", is basically how the site works, updates and changes. This refers to everything the user can't see in the browser, like databases and servers.

5.1.1 Features of front-end

HTML5 code along with CSS3 is used for styling while JavaScript, is used for validation at frontend. React is an open-source JavaScript library used for frontend development. Its component-based library lets you build high-quality user-interfaces for web apps.

React is a library that defines the way apps are written. It does this by setting very clear rules about how data can flow through the app, and how the UI will adapt as a result of that changing data. There are other libraries that set similar boundaries, such as Angular and Vue.

Plain JavaScript code (that is, JavaScript written without libraries) on the other hand, can be thought of as a scripting language that doesn't set any rules about how data can be defined, or how the UI can be changed. That makes apps written without these libraries more freeform and customizable. But going this route can also lead to problems down the road.

5.1.2 Features of back-end

Firebase is a mobile- and web application development platform, backed by Google, to help developers deliver richer app experiences. Firebase manages its own infrastructure with a nice set of tools to simplify the workflow of the developer by providing them with development kits and an online dashboard. These toolkits are interconnected, scalable and integrable with third party software to overcome complex challenge

The platform consists of a great set of development tools. The Realtime Database and Cloud Firestore can stock document-structured data and synchronize the corresponding apps in milliseconds whenever a data transformation occurs. This means that both the app and its database listen to each other, providing the user with reactive app experiences. And Firebase Cloud Functions can even extend this functionality. These functions allow the developer to write backend code to respond to events happening in the Firebase platform without having to deal with any servers.

With Firebase **Authentication** you can verify users through both social and e-mail logins and manage them securely. Firebase **Hosting** can deploy both static and dynamic content to web apps and **Cloud Storage** can accumulate and serve a huge amount of files. Last but not least, there's ML Kit. This module brings machine learning to your app with ready-to-use APIs and custom models using TensorFlow Lite, an open source machine learning platform by Google.

5.2 Discussion of code segments

This section includes the segments of code used to provide various user functionalities.

5.2.1 Code segment to establish connection with the firebase

The below is the code segment to establish the connection with the firebase.

```
import firebase from "firebase";

const firebaseConfig = {
  apiKey: "AIzaSyB__jVz8KWP-iK6Fv8m3zOeW6ogRpEIIHs",
  authDomain: "challenge-525ac.firebaseio.com",
  projectId: "challenge-525ac",
  storageBucket: "challenge-525ac.appspot.com",
  messagingSenderId: "1029790599229",
  appId: "1:1029790599229:web:be1642b5372cdcc09c8254",
  measurementId: "G-S9E41CX0R8",
};
```

```
const firebaseApp = firebase.initializeApp(firebaseConfig);
```

```
const db = firebaseApp.firestore();  
const auth = firebase.auth();  
  
export { db, auth };
```

5.2.2 Code segment for login page.

Below is the code segment for login authentication.

```
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) => {  
        history.push("/");  
      })  
      .catch((error) => alert(error.message));  
  };  
  
  const register = (e) => {  
    e.preventDefault();  
  
    auth  
      .createUserWithEmailAndPassword(email, password)  
      .then((auth) => {
```

```
    if (auth) {
      history.push("/");
    }
  })
  .catch((error) => alert(error.message));
};

return (
  <div className="login">
    <Link to="/">
      
    </Link>

    <div className="login__container">
      <h1>Sign-in</h1>

      <form>
        <h5>E-mail</h5>
        <input
          type="text"
          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 signing-in you agree to the Webstore Conditions of Use &
  Sale. Please see our Privacy Notice, our Cookies Notice and our
  Interest-Based Ads Notice.
</p>

<button onClick={register} className="login__registerButton">
  Create your WebStore Account
</button>
</div>
</div>
);
}

export default Login;

```

5.2.3 Code segment of Home page

Below is the code segment:

```

import React from "react";
import "./Home.css";
import Product from "./Product";
function Home() {
  return (
    <div className="home">
      <div className="home__container">
        
  <div className="home__row">
    <Product
      id="12321341"
      title="The Lean Startup: How Constant Innovation Creates Radically Succ
essful Businesses Paperback"
      price={ 11.96 }
      rating={ 5 }
      image="https://images-na.ssl-images-
amazon.com/images/I/51Zymoq7UnL._SX325_BO1,204,203,200_.jpg"
    />
    <Product
      id="49538094"
      title="Kenwood kMix Stand Mixer for Baking, Stylish Kitchen Mixer with
K-beater, Dough Hook and Whisk, 5 Litre Glass Bowl"
      price={ 239.0 }
      rating={ 4 }
      image="https://images-na.ssl-images-
amazon.com/images/I/81O%2BGNdkzKL._AC_SX450_.jpg"
    />
  </div>
  <div className="home__row">
    <Product
      id="4903850"
      title="Samsung LC49RG90SSUXEN 49' Curved LED Gaming Monitor"
      price={ 199.99 }
      rating={ 3 }
      image="https://images-na.ssl-images-
amazon.com/images/I/71Swqqe7XAL._AC_SX466_.jpg"
    />
    <Product
      id="23445930"
      title="Amazon Echo (3rd generation) | Smart speaker with Alexa, Charcoal
Fabric"
      price={ 98.99 }
      rating={ 5 }
      image="https://media.very.co.uk/i/very/P6LTG_SQ1_0000000071_CHAR
COAL_SLf?$300x400_retinamobilex2$"
```

```

    />
    <Product
      id="3254354345"
      title="New Apple iPad Pro (12.9-inch, Wi-Fi, 128GB) -
Silver (4th Generation)"
      price={598.99}
      rating={4}
      image="https://images-na.ssl-images-
amazon.com/images/I/816ctt5WV5L._AC_SX385_.jpg"
    />
  </div>
  <div className="home__row">
    <Product
      id="90829332"
      title="Samsung LC49RG90SSUXEN 49' Curved LED Gaming Monitor -
Super Ultra Wide Dual WQHD 5120 x 1440"
      price={1094.98}
      rating={4}
      image="https://images-na.ssl-images-
amazon.com/images/I/6125mFrzr6L._AC_SX355_.jpg"
    />
  </div>
</div>
</div>
);
}

```

```
export default Home;
```

5.2.4 Code segment of Payment page

Below is the code segment of Payment page.

```

import React, { useState, useEffect } from "react";
import "../Payment.css";
import CheckoutProduct from "../CheckoutProduct";
import { useStateValue } from "../StateProvider";
import { Link, useHistory } from "react-router-dom";
import { CardElement, useElements, useStripe } from "@stripe/react-stripe-js";
import CurrencyFormat from "react-currency-format";

```

```
import { getBasketTotal } from "./reducer";
import axios from "./axios";

function Payment() {
  const [{ basket, user }, dispatch] = useStateValue();

  const history = useHistory();

  const stripe = useStripe();
  const elements = useElements();

  const [succeeded, setSucceeded] = useState(false);
  const [processing, setProcessing] = useState("");

  const [error, setError] = useState(null);
  const [disabled, setDisabled] = useState(true);
  const [clientSecret, setClientSecret] = useState(true);

  useEffect(() => {
    const getClientSecret = async () => {
      const response = await axios({
        method: "post",
        url: `/payments/create?total= ${getBasketTotal(basket) * 100}`,
      });
      setClientSecret(response.data.clientSecret);
    };
    getClientSecret();
  }, [basket]);

  console.log("The secret is >>>", clientSecret);

  const handleSubmit = async (event) => {
    event.preventDefault();
    setProcessing(true);

    //const payload=await stripe
    const payload = await stripe
      .confirmCardPayment(clientSecret, {
```



```
    payment_method: {
      card: elements.getElement(CardElement),
    },
  })
  .then(({ paymentIntent }) => {
    setSucceeded(true);
    setError(null);
    setProcessing(false);

    history.replaceState("/orders");
  });
};

const handleChange = (event) => {
  setDisabled(event.empty);
  setError(event.error ? event.error.message : "");
};

return (
  <div className="payment">
    <div className="payment__container">
      <h1>
        Checkout(<Link to="/checkout">{basket?.length} items</Link>)
      </h1>
      <div className="payment__section">
        <div className="payment__title">
          <h3>Delivery Address</h3>
          <div className="payment__address">
            <p>{user?.email}</p>
            <p>123 React Lane</p>
            <p>Los Angeles, CA</p>
          </div>
        </div>
      </div>
      <div className="payment__section">
        <div className="payment__title">
          <h3>Review items and delivery</h3>
        </div>
        <div className="payment__items">
```

```

    {basket.map((item) => (
      <CheckoutProduct
        id={item.id}
        title={item.title}
        image={item.image}
        price={item.price}
        rating={item.rating}
      />
    ))}
  </div>
</div>
<div className="payment__section">
  <div className="payment__items">
    <h3>Payment method</h3>
  </div>
  <div className="payment__details">
    <form onSubmit={handleSubmit}>
      <CardElement onChange={handleChange} />
      <div className="payment__priceContainer">
        <CurrencyFormat
          renderText={(value) => <h3>Order Total: {value}</h3>}
          decimalScale={2}
          value={getBasketTotal(basket)}
          displayType="text"
          thousandSeparator={true}
          prefix={"$"}
        />
        <button disabled={processing || disabled || succeeded}>
          <spam>{processing ? <p>Processing</p> : "Buy Now"}</spam>
        </button>
      </div>

      {error && <div>{error}</div>}
    </form>
  </div>
</div>
</div>
</div>
</div>

```

```
);  
}
```

```
export default Payment;
```

5.2.5 Code segment to checkout page

Below is the code segment for checkout page.

```
import React from "react";  
import "./Checkout.css";  
import Subtotal from "./Subtotal";  
import { useStateValue } from "./StateProvider";  
import CheckoutProduct from "./CheckoutProduct";  
  
function Checkout() {  
  const [{ basket, user }, dispatch] = useStateValue();  
  return (  
    <div className="checkout">  
      <div className="checkout__left">  
          
        <div>  
          <h3>Hello, {user?.email}</h3>  
          <h2 className="checkout__title">Your shopping Basket</h2>  
  
          {basket.map((item) => (  
            <CheckoutProduct  
              id={item.id}  
              title={item.title}  
              image={item.image}  
              price={item.price}  
              rating={item.rating}  
            />  
          )
```

```
    )})  
  </div>  
</div>  
  
  <div className="checkout__right">  
    <Subtotal />  
  </div>  
</div>  
);  
}  
  
export default Checkout;
```

Chapter 6

TESTING AND RESULTS

6.1 Testing

Software testing is conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation

6.1.1 Unit testing

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

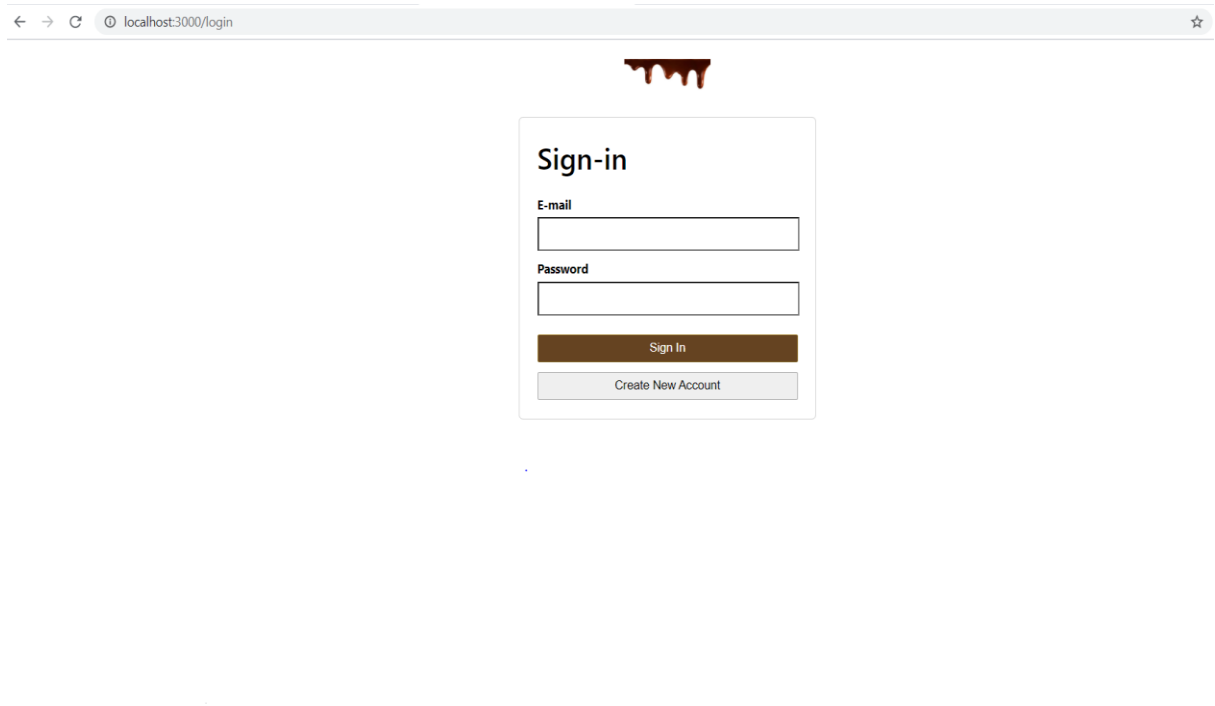
6.1.1.1 Case 1

This test case checks to see if the customer is able to Sign-in or not, by entering the correct credentials as shown in Table 6.1 and Figure 6.1


Table 6.1 Unit test case for customer Sign-in check

Sl No of test case:	1
Name of test:	Login Check
Feature being tested:	Customer Login
Sample Input:	Enter username and password fields and click on login.
Expected Output:	Redirect to Home page
Actual Output:	Home page is displayed
Remarks	Pass

Figure 6.1 Unit test case for Customer login check



← → ↻ localhost:3000/login ☆



Sign-in

E-mail

Password

Figure 6.1 Unit test case for Customer login check

6.1.2 Integration testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

6.1.1.2 Case 1

This test case checks to see if the particular item is added or not, to the basket which is as shown in Table 6.2 and Figure 6.2.

Table 6.2 Integration test case for add item check

Sl No of test case:	1
Name of test:	Add item Check
Feature being tested:	Add item button
Sample Input:	Click on any item that you want to buy by clicking on Add to cart button
Expected Output:	Basket gets updated and items are shown in cart page
Actual Output:	Basket gets updated and items are shown in Cart page

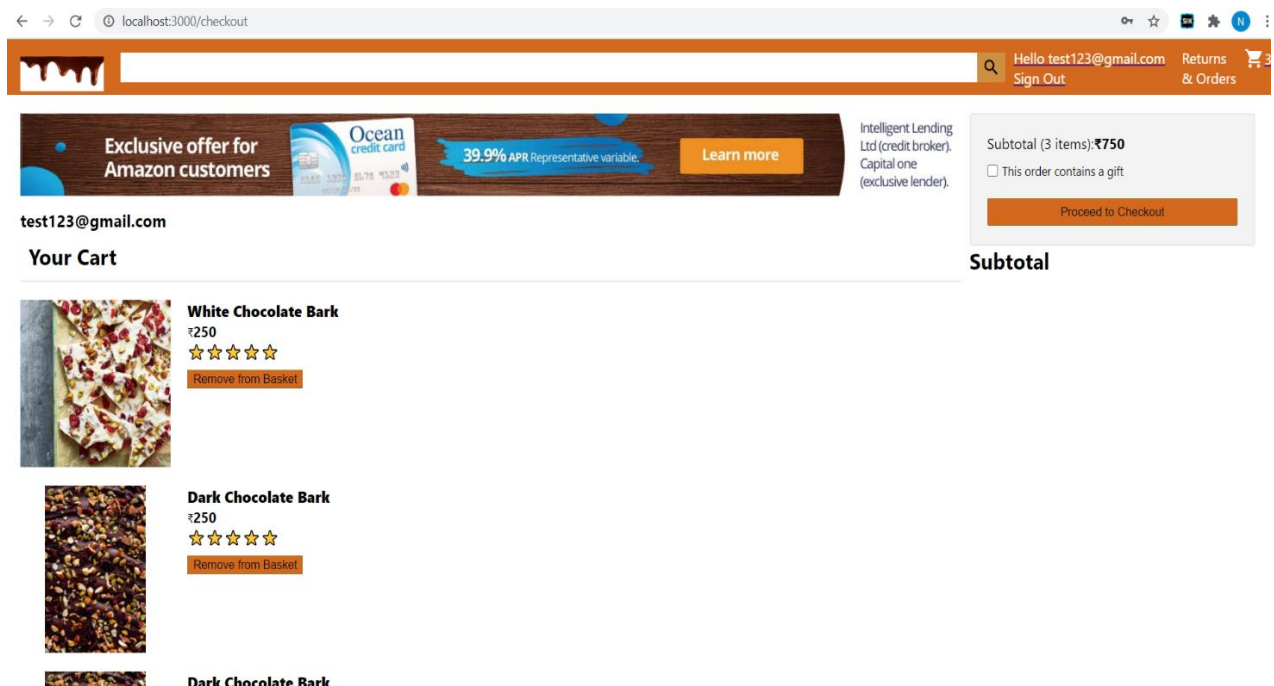


Figure 6.2 Add Employee snapshot

6.1.3 System Testing

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.

6.1.1.3 Case 1

This test case checks to see if statistics of the site is displayed or not. This checks to see whether all the items present in the basket will be present in the checkout page or not which is shown in the table 6.3 and figure 6.3 below.

Table 6.3 System test case for checkout page

SI No of test case:	1
Name of test:	System testing for checkout page.
Feature being tested:	Proceed to checkout button
Sample Input:	Click on proceed to checkout to buy the items, that are in the cart
Expected Output:	Redirects to checkout page having all buying items
Actual Output:	Redirects to checkout page having all buying items
Remarks	Pass

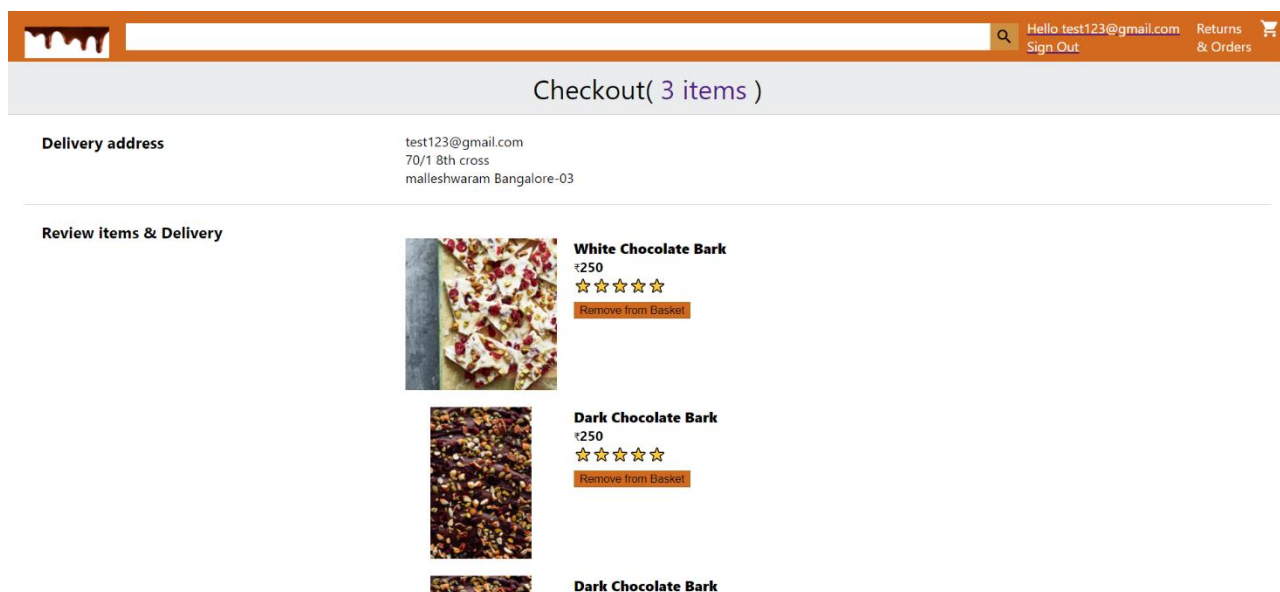


Figure 6.3 checkout page displayed successfully

6.2 Discussion of results

The outcomes of test results for a variety of user interactions with the application are discussed in the following sections of the chapter.

6.2.1 Home page

Figure 5.6 shows the home page which is displayed after login.

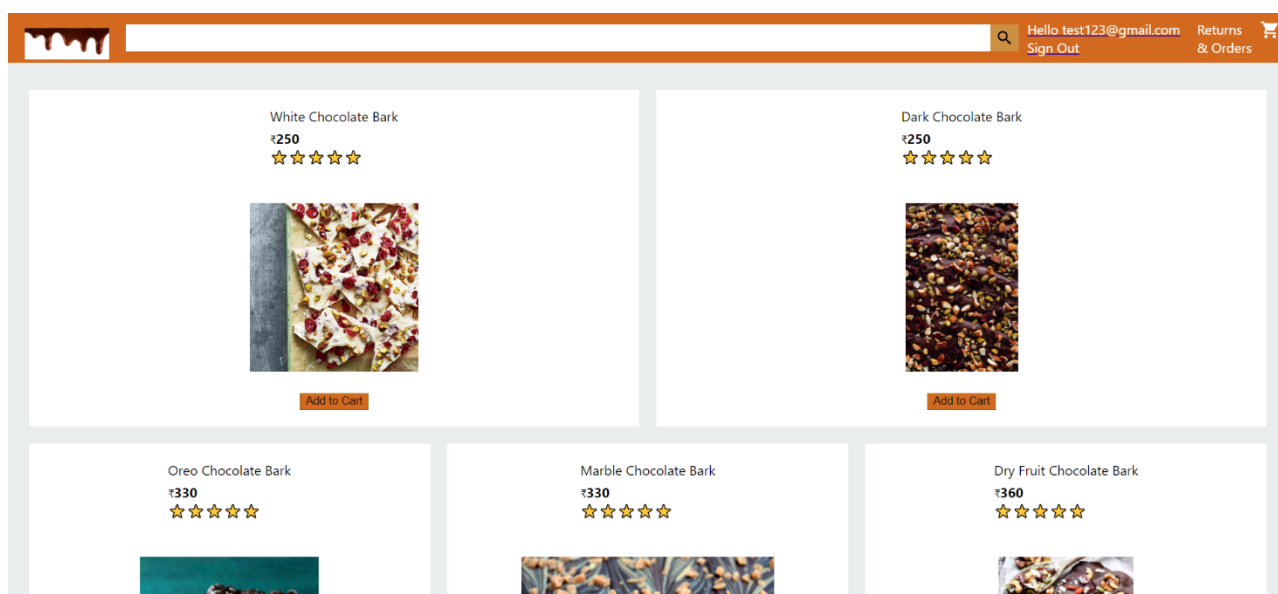


Figure 5.4 Home page

6.2.2 Admin Login Page

Figure 5.7 shows customer login page which is displayed for the customer to login.

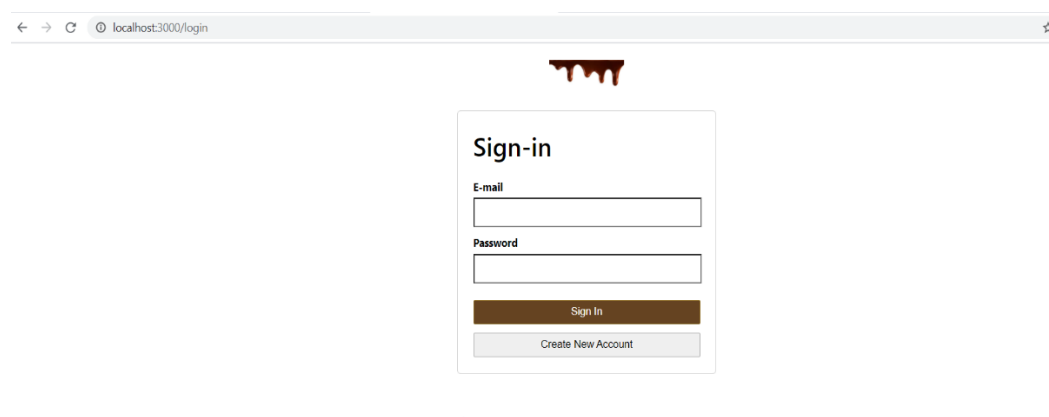


Figure 5.5 Customer login page

6.2.3 Cart

Figure 5.8 shows your cart after adding items to the basket.

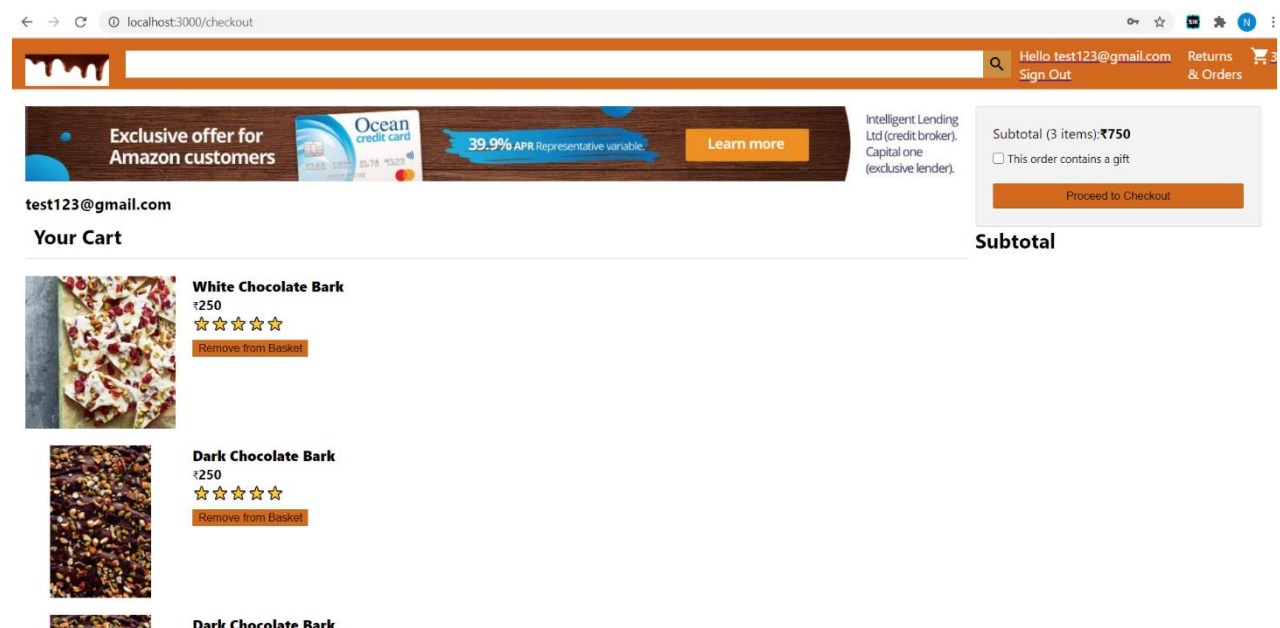


Figure 5.6 Cart

6.2.4 Checkout page

Figure 5.9 shows the checkout page

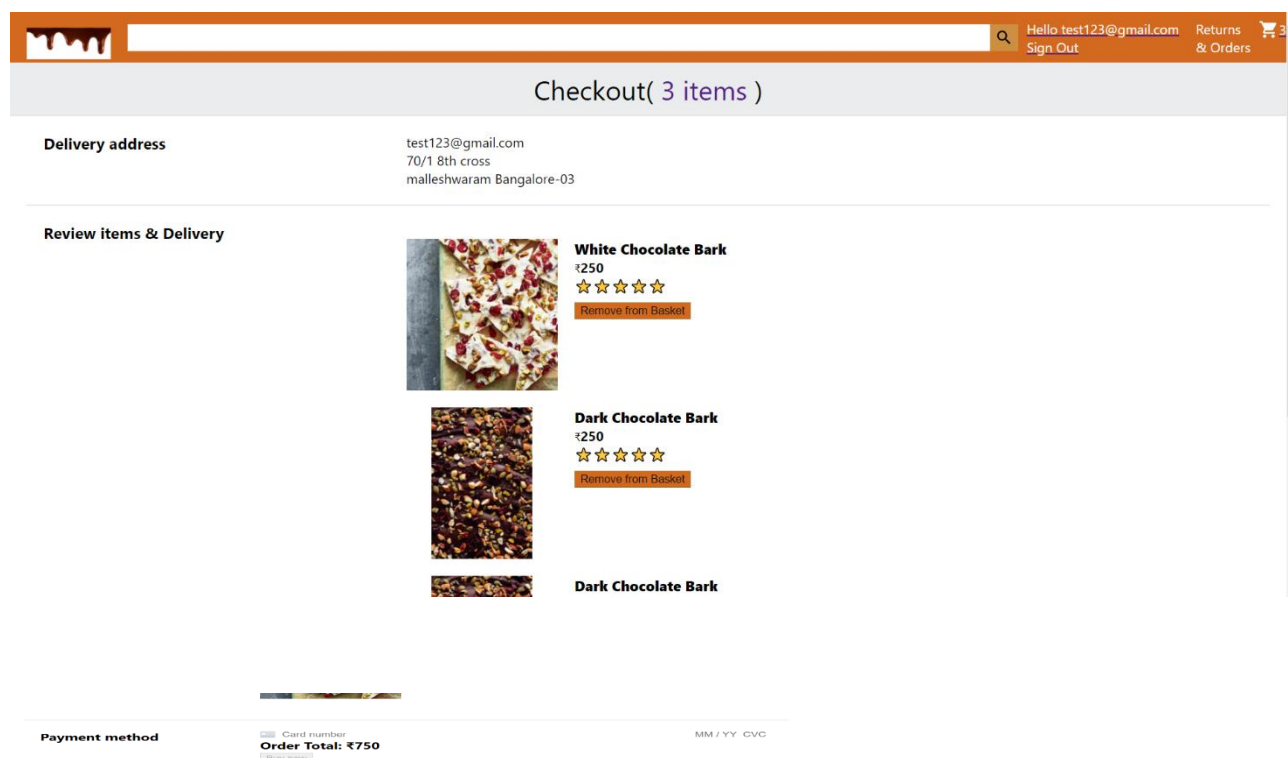


Figure 5.7 Checkout page

Chapter 7

INSTALLATION INSTRUCTIONS

This chapter provides step-by-step instructions to install firebase into your project

7.1 Adding Firebase

Firebase provides the tools and infrastructure you need to develop, grow, and earn money from your app. This package supports web (browser), mobile-web, and server (Node.js) clients.

- **Firebase Realtime Database** - The Firebase Realtime Database lets you store and query user data, and makes it available between users in realtime.
- **Cloud Firestore** - Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.
- **Firebase Storage** - Firebase Storage lets you upload and store user generated content, such as files, and images.
- **Firebase Cloud Messaging** - Firebase Cloud Messaging is a cross-platform messaging solution that lets you reliably deliver messages at no cost.
- **Firebase Authentication** - Firebase helps you authenticate and manage users who access your application.
- **Create and setup your account** - Get started using Firebase for free.

This SDK is intended for end-user client access from environments such as the Web, mobile Web (e.g. React Native, Ionic), Node.js desktop (e.g. Electron), or IoT devices running Node.js. If you are instead interested in using a Node.js SDK which grants you admin access from a privileged environment (like a server), you should use the **Firebase Admin Node.js SDK**.

.

Include Firebase in your web application via a `<script>` tag:

```
<script  
src="https://www.gstatic.com/firebasejs/${JSCORE_VERSION}/firebase.js"></script>
```

```
<script>  
  
var app = firebase.initializeApp({  
  apiKey: '<your-api-key>',  
  authDomain: '<your-auth-domain>',  
  databaseURL: '<your-database-url>',  
  projectId: '<your-cloud-firestore-project>',  
  storageBucket: '<your-storage-bucket>',  
  messagingSenderId: '<your-sender-id>'  
});  
  
// ...  
</script>
```

Install the Firebase npm module:

```
$ npm init
```

```
$ npm install --save firebase
```

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENTS

Our Web Store System is not just about conducting business transactions via the Internet. Its impact will be far-reaching, and more prominent than 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. For the future enhancement :

- We can implement real time database in this project to handle processing of workloads whose states is constantly changing.
- We can implement cloud functions which is a serverless framework that lets you automatically run backend code in response to events triggered by Firebase features and HTTPS requests.
- We can add recommendation feature to recommend items to the customer of their choice.

REFERENCES

- [1] Raghu Ramakrishnan and Johannes Gehrke , Database Management Systems , McGRAW HILL , 3rd Edition.
- [2] Ramez Elmasri and Shamkant B. Navathe , Fundamentals of Database Systems , Pearson , 7th Edition
- [3] Randy Connolly, Ricardo Hoar, "Fundamentals of Web Development", 1stEdition, Pearson Education India. .
- [4] Robin Nixon, "Learning PHP, MySQL &JavaScript with jQuery, CSS and HTML5", 4thEdition, O'Reilly Publications, 2015.
- [5] Learning React : Fundamental web development with react and redux.
- [6] Nicholas C Zakas, "Professional JavaScript for Web Developers", 3rd Edition, Wrox/Wiley India, 2012.
- [7] www.php.net
- [8] www.javascript.com
- [9] www.w3schools.com