



Tran Nham

Developing an E-commerce Application Prototype with ReactJS and Firebase

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology - Mobile Solutions

Bachelor's Thesis

15 April 2022

Abstract

Author:	Tran Nham
Title:	Developing an E-commerce application prototype with ReactJS and Firebase
Number of Pages:	32 pages + 1 appendices
Date:	15 April 2022
Degree:	Bachelor of Engineering
Degree Programme:	Information Technology
Professional Major:	Mobile Solutions
Supervisors:	Janne Salonen

In the context of the current epidemic, e-commerce platforms have become increasingly popular and essential for people's daily needs. It is crucial to continuously research and develop modern technologies for supporting these platforms. In this thesis, we aim to explain the process of developing an e-commerce application with the help of ReactJS and Firebase services.

Besides giving a brief description about E-commerce, the thesis also helps studying the fundamental aspects of ReactJS and Firebase. The core of the process is to establish a Firebase conceptual framework that makes use of its tools and services.

The outcome of the project based on this thesis is an e-commerce webstore prototype. The web app will allow people to buy and sell items that are relevant to their needs. The website also provides useful tools to help the owners managing their e-commerce store in handling orders, payments, and logistic. By implementing the serverless backend, the maintenance has become easier and easier, especially for the developers compared to the traditional backend technologies.

Keywords: E-commerce, Web Application, ReactJS, Firebase

Contents

Introduction	1
1 E-commerce	2
1.1 Definition	2
1.2 Categories	2
1.3 Advantages	2
1.4 Disadvantages	4
2 ReactJS	4
2.1 What is React?	4
2.2 React Features	5
3 Firebase	8
3.1 Overview	8
3.2 Firebase Outstanding services	9
3.2.1 Firebase Authentication	10
3.2.2 Firebase Realtime Database	11
3.2.3 Firebase Cloud Storage	12
3.2.4 Firebase Hosting	14
3.2.5 Performance Monitoring	16
4 Web application project	17
4.1 Overview and project flow	17
4.2 Setting up the develop environment	18
4.3 Structuring the project	18
4.4 Application elements	20
4.4.1 Index.html	20
4.4.2 Styling with CSS	20
4.4.3 JavaScript and components	22
4.4.4 Redux	24
4.5 Configuring Firebase	26
4.6 User interface	27
4.7 Deploying the project with Firebase Hosting	30
4.8 Testing	30

4.9 Further Discussion	31
5 Conclusion	32
References	32

List of Abbreviations

API:	Application Programming Interfaces, make software development and innovation simpler by making it easy and safe for programs to share data and functions.
DOM:	Document Object Model is a programming interface for HTML and XML documents. It shows what the page looks like so that programs can change the document's structure, style, and content. It shows the document with nodes and objects.
JSON:	JavaScript Object Notation format is used in data storing and transmitting
JSX:	JavaScript XML is an extended syntax that allows programmers to write HTML in React easily.
NPM:	Node package manager is a Node.js tool to construct and administer JavaScript programming libraries.
REST:	Representational State Transfers is an architectural style applied to networked applications. It exists as a series of constraints applied to implementations of network elements, allowing for unified interface semantics, rather than application-specific syntax and implementations.
UI:	User Interface is everything a user interacts with when using a digital product or service.

Introduction

Modern technology has improved the quality of human life in many aspects. Internet has become a crucial aspect of our daily lives, especially in this current pandemic status and crisis. By the early 2022, there were an estimated 5.25 billion active internet users worldwide, or 66.2 percent of the global population, according to (www.broadbandsearch.net). Throughout history, the convenience of ordering food or other goods online has grown in importance. Online shopping can be done via a website or a mobile app.

In 2020, 56.8% of all internet traffic was generated by mobile devices. However, a mobile landing page takes an average of 25 seconds to load. You can't argue with the fact that 55 percent of people will close the website if the loading time is more than 3 seconds. A large amount of web users has shifted to use native applications for their day-to-day activities on the internet. e-commerce has seen a 62 percent growth in this sector in 2019. As a result, web users are spending less time on their browsers, and web developers are scrambling to keep up with native apps in terms of user experience.

E-commerce web applications for businesses, especially small ones, are the focus of the thesis. The outcome is to build the prototype of an e-commerce web application. It's possible to create dynamic websites with JavaScript, which responds to user requests and improves the user experience. React, a free-to-use JavaScript library is utilized to construct a friendly and efficient user interface.

One of the thesis aims is to develop a functional responsive e-commerce web application for an online retailer by combining ReactJS and the Firebase cloud service. The application will provide all basic features of an E-commerce webstore: authentication, viewing and purchasing products, handling orders and admin panel to help managing the products and orders

1 E-commerce

1.1 Definition

Ecommerce, often known as electronic commerce or online commerce, refers to the buying and selling of things and services through the internet. The term "e-commerce" was initially used in the 1960s. With the rising popularity of mobile devices, social media has become a powerful affirmation of the strength and growth of the web page after years of development. Commercial launchers help to expedite the growth of trade (E-commerce). [1]

1.2 Categories

There are now several types of e-commerce:

Business to Business refers to transactions between businesses, corporations, and organizations. Over three quarters of current e-commerce operations is in this group.

Business to Consumer Also known as retail ecommerce, this business model comprises sales between online firms and customers.

Consumer to Company refers to an individual who sells goods and services to a business or organization.

Consumer to Consumer refers to transactions among consumers.

There are more variations, such as G2C, G2B, etc., although they are less common than these four.

1.3 Advantages

Elevating Buying Process: E-commerce has accelerated the entire purchasing procedure for consumers. They can purchase things from the comfort of their own

homes, without the need to visit actual stores. It saves enormous amounts of time and expedites transactions.

Minimizing the cost: Businesses do not need to create physical stores during conducting E-commerce model. There are substantial costs associated with operating a store, including rent, utilities, other bills, and employee compensation. It eliminates the physical costs and conducts all commercial operations via an internet platform.

Personalizing Shopping Experiences: Consumers could have a customized shopping experience. Consumers can browse for a large range of products based on their preferences and needs without restriction. On e-commerce websites, customers are presented products depending on their interests and geography.

24/7 Availability: The option to shop online is available all the time, 24 hours a day, 7 days a week. One of the best things about e-commerce is that customers can buy things online at any time. Unlike traditional stores, this site doesn't have set times for when it opens and when it closes.

Global connections: Without geographical restrictions, online firms can reach and engage with customers in remote locations. People can place orders from any location and receive delivery to their location.

Clarified product information: Customers can discover a comprehensive product description. It gives detailed information so that customers can easily compare it to other products and select the best one.

.

Retargeting Customers: Online purchasing has simplified the process of retargeting customers for businesses. While clients are shopping online, the e-commerce enterprise collects a large amount of information about them. Periodically, customers can be contacted by sending them personalized emails, messages, promotions, and discounts.

1.4 Disadvantages

In the case of online purchasing, customers lack the ability to touch and feel. Customers are sometimes happier with in-person purchases since they may inspect the product before to purchase.

Customers are unable to obtain assurance regarding the quality of the products. They may be deceived by businesses and acquire defective goods.

Security Concerns: During online purchasing online, customers have the risk of being stolen crucial credentials. Hackers may steal client information and lead to serious financial lost.

Another significant disadvantage of e-commerce is that buyers must wait longer to receive their packages. With offline shopping, clients receive immediate delivery of their purchases.

Customers are not able to test out a product before making a purchase when they shop online. This is a disadvantage. They cannot negotiate prices and acquire additional information about the product's usage and qualities, as they would in a real store with a salesperson.

2 ReactJS

2.1 What is React?

React.js is an open-source JavaScript package that is used to make user interfaces for single-page apps. It takes care of the view layer for apps that run online and on phones. React also lets us make user interface parts that can be used more than once. React was first made by a software engineer at Facebook named Jordan Walke. React was first used in 2011 on Facebook's news feed and in 2012 on Instagram.com.

With React, developers can make large web apps that can change data without having to reload the page. React's main goal is to be fast, easy to use, and scalable. It only works on the user interfaces of the app. The view in the MVC template is the same as this. It can work with other libraries or frameworks written in JavaScript, like Angular JS in MVC. React is the shorter form of ReactJS.

2.2 React Features

These are the features that make the React library so good and strong for building modern apps, as well as what distinguishes React from competing frameworks and libraries:

2.2.1 Virtual DOM

DOM (Document Object Model) is the most essential component of web applications; it describes the HTML-based structure of a web page document.

DOM manipulation is frequently utilized nowadays since contemporary applications demand a lot of state changes, animations, effects, and so on.

Consider the following scenario: you have an application, and you want to update just the areas of your DOM tree that are affected by state changes. You don't want to re-render your whole UI from start because it would be expensive in terms of performance and user experience. React employs a feature known as virtual DOM, which is a virtual version of the real DOM tree. It's simply a tree data structure of basic JavaScript objects that is maintained in memory synchronized as depicted in figure 1. Because the virtual DOM will never be presented to the user, it will only exist in memory, rendering it is quicker.

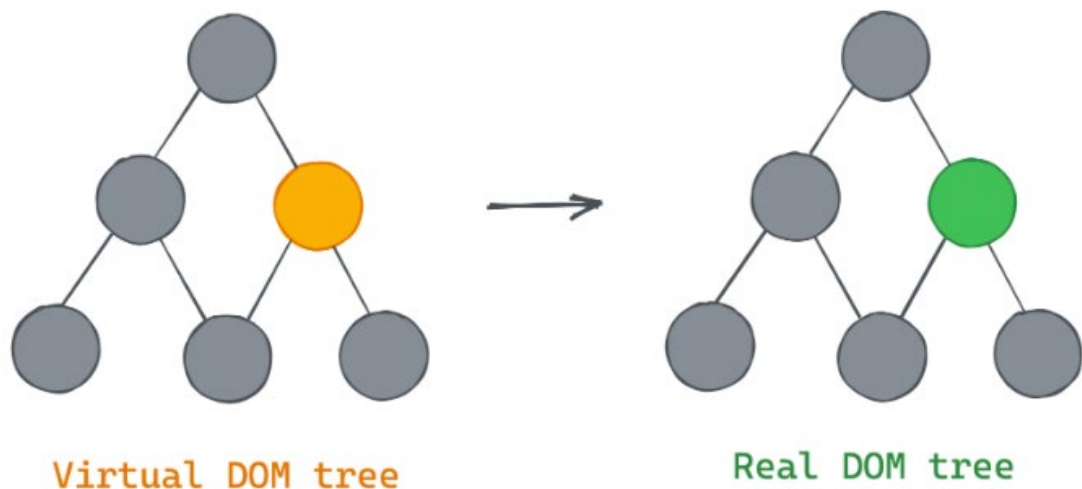


Figure 1. DOM Working flow in React

React builds a duplicate of your real DOM tree when your React app loads. Instead of re-rendering the whole real DOM tree whenever a state change occurs in your application, React first updates its virtual DOM with the changed state. React then compares its virtual DOM to your actual DOM tree to determine what needs to be modified. Then it modifies your actual DOM tree, but it just changes

the components that need to be altered. The virtual DOM is one of the aspects that contribute to the React framework's speed and dependability.

2.2.2 JSX

JSX is an extension of JavaScript used by React. We utilize it to construct "elements" in React. Preprocessors (such as Babel) employ JSX to turn HTML-like content in JavaScript files into parsable JavaScript objects. React does not mandate the use of JSX, however many developers find it beneficial for working with the UI within the JavaScript code. JSX is an essential component of React since it is used to generate React components.

2.2.3 Components

React components are responsible for the reusability of our code and the segmentation of our UI into distinct parts. React components operate in the same method as JavaScript functions. A React component takes arbitrary inputs, called props, and must always return a React element indicating what should be presented to the user. Defining a JavaScript function that returns a React element is the simplest approach to construct a React component.

```
1 function Welcome(props) {  
2   return  
3     <h1>My first React component</h1>  
4 }  
You, seconds ago • Uncommitted changes
```

Figure 2. Defining a React component

In the figure, Welcome is a React component that receives a single prop named props and returns a React element, in this example a basic h1 element. If a React component fails to return a React element, it will issue an error. React components adhere to the separation of concerns design concept, which dictates that a computer program (in this example, our application) should be divided into distinct pieces, with each section addressing distinct issues.

React components are extremely effective because they enable us to write code that is clearer, more resilient, and more succinct throughout the whole application. We can have an unlimited number of React components.

3 Firebase

In 2011, Firebase was born under the name Envolv by James Tamplin and Andrew Lee. The purpose Envolv provides developers with APIs is to integrate live chat functionality into websites. However, more than just chatting, users have expanded the usability of Envolv. Developers have taken advantage of Envolv to transfer application data such as online games, contacts, calendars, etc.

Therefore, the two founders of Envolv split the online messaging system and real-time data synchronization into two separate parts. In April 2012, Firebase was created as a separate company Backend-as-a-Service with Realtime functionality. In 2014, Google acquired Firebase. After that, Firebase quickly evolved into the versatile application of today's mobile and web platforms.

3.1 Overview

After Google acquisitions, Firebase quickly developed into the multipurpose mobile and website application development platform it is today. system, Firebase focus is on 2 main objects:

- Developing and testing
- Analysing data and optimizing experience.

Firebase gives us simple and powerful APIs that can be used cross-platform in managing the database, so now we just need to call the API and the server part has Firebase taking care of it. See figure 3 below.

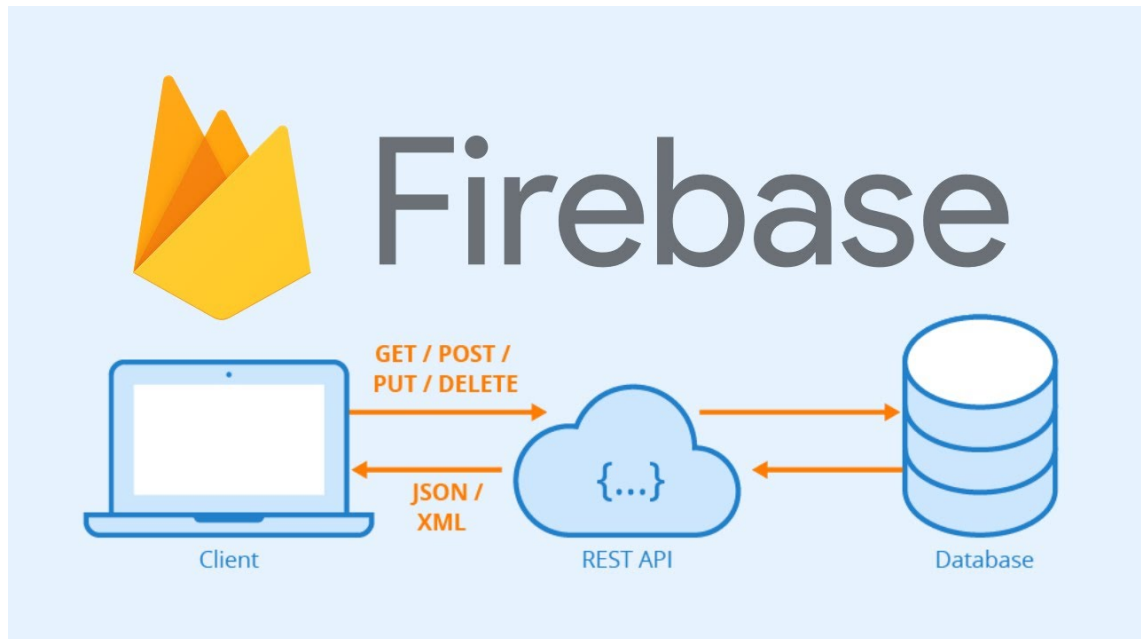


Figure 3. Firebase concept

Firebase is a database service that operates in the cloud. Google's server system is incredibly powerful. Its primary purpose is simplifying database operations.

In simple API, the objective is to raise the amount of users and maximize profitability.

Additionally, the service is highly flexible and secure. Firebase supports both Android and iOS. It is not surprising that many developers chose Firebase as their initial platform for creating apps for millions of people globally.

3.2 Firebase Outstanding services

Firebase is a versatile platform that provides a lot of different services to its users. But when it comes to this platform, people still think of the following outstanding services.

3.2.1 Firebase Authentication

The most prominent activity of Firebase is building user authentication steps through Email, Facebook, Twitter, GitHub or Google. In addition, the Firebase Authentication operation also supports anonymous authentication for applications. Firebase's authentication can help keep users' personal information safer. This also ensures the user's account and personal information is not stolen.

Firebase Authentication can be seen as an intermediary that will handle all 3rd party logins as can be seen in figure 4.

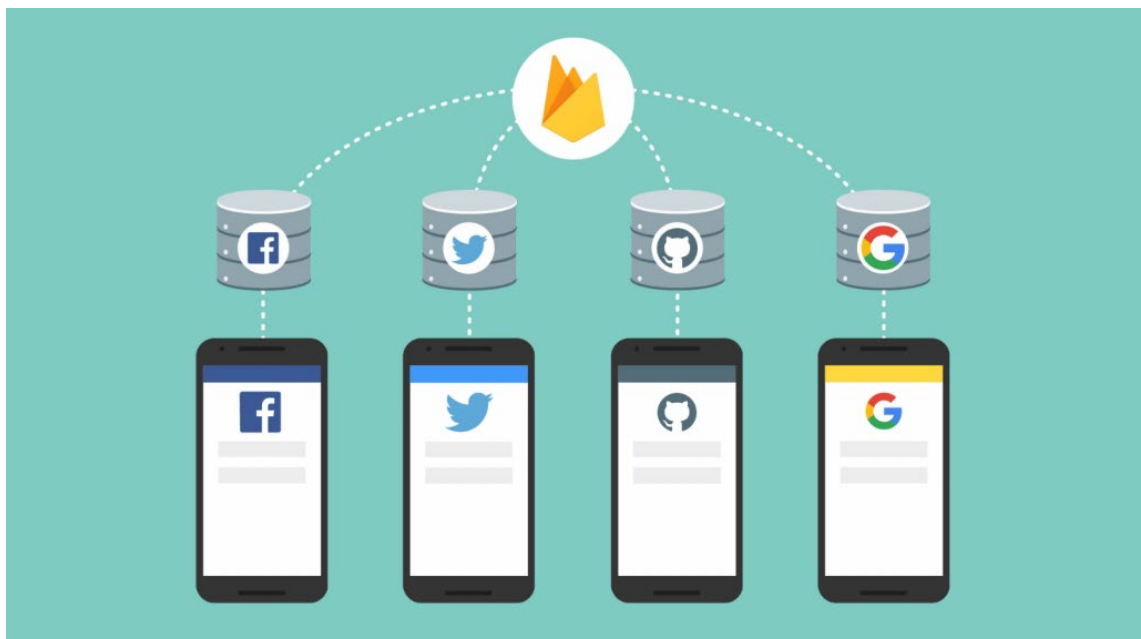


Figure 4. Firebase-supported 3rd party logins

Firebase's user interface provides access to both password recovery and user verification. Without disclosing their true identity, customers can set up an anonymous temporary account that is able to link to 3rd party provider later. In addition, it features a technique known as Smart Lock for automatically remembering and logging in with credentials. Email address is required as a sign-in credential by the Firebase authentication SDK.

3.2.2 Firebase Realtime Database

Firebase Realtime Database is a NoSQL data type stored in the cloud that allows you to store and synchronize user data in real time. In practical, your data is stored as a JSON object, which the developer can manage in real time. Realtime Database Service



Figure 5. Data stored in real-time database.

As can be seen from the illustration in figure 5 above, with only a single API, you will get both the latest data and its updates.



Figure 6. Realtime syncing illustration

Realtime syncing helps users access their data from any device. One advantage of Realtime Database is that Firebase will provide you with an SDK for you to easily build mobile and web applications without a server. When the device is offline, the Realtime Database SDK uses the device's memory. The application still interacts with the user as usual. When the device is back online, it automatically syncs to the server.

Using a few client-side codes, it specifies who can view specific information. Thanks to real-time database (NoSQL-based database) hosting, operating and maintaining server is no longer compulsory. The Real-time database is compatible with Android, iOS, Web, C++, Unity Platform, and JavaScript (Google 2019). Real-time Database supports up to one hundred thousand concurrent connections and one thousand per second per database.

Due to the different database, the data set is modified and improved, and a real-time database is integrated using Cocoa Pods or Gradle to improve client access time. This approach additionally provides offline writing capabilities while maintaining client security. These are the fundamental stages for creating a Firebase Real-time Database: Integrate the Firebase Realtime Database SDKs, build Realtime Database References, enable Offline Persistence, and safeguard data (Google 2019).

3.2.3 Firebase Cloud Storage

The following figure illustrates how Firebase Cloud Storage is the data storage space. If you are planning to work with Google Cloud Platform, specifically Google Cloud Storage, then Firebase Cloud Storage is the top choice in the list of services that you need to refer to.

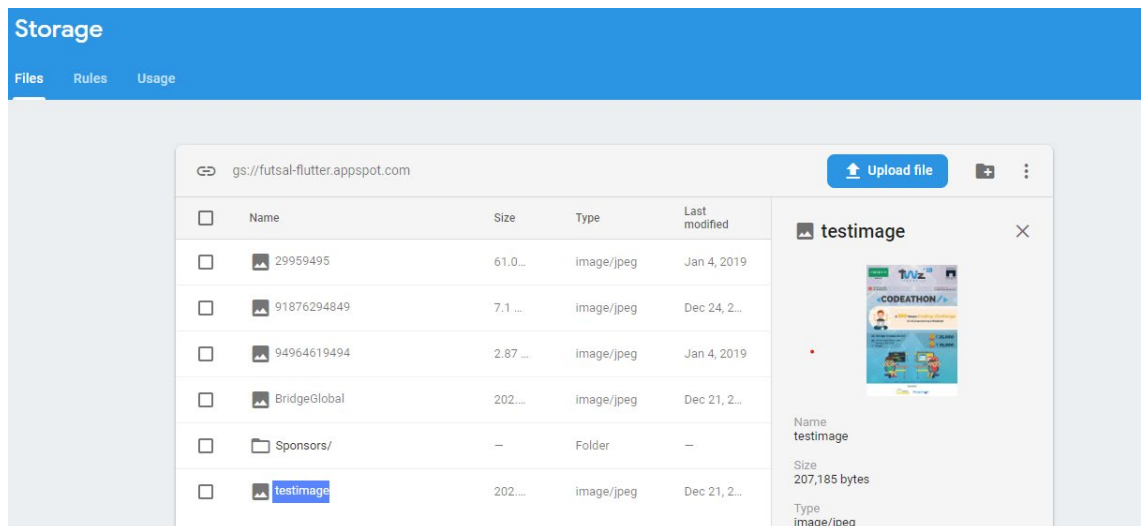


Figure 7. Firebase Cloud Storage example

In short, Firebase Cloud Storage is the data storage space. The data here has no limit at all. You can contain any file(s) you want, like photos, music, videos or text files, zip or even a file with your own data type of your own design. You might be thinking, “Hey, how does that sound like Google Drive?” – Same, but the purpose of Firebase Cloud Storage is for programming activities, as a foundation for you to build software products on, not for personal purposes. In fact, it's like the storage space on your Web host.

If you already have a Realtime Database, why to have Cloud Storage? The answer is very simple. Realtime Database is more "as a database" than Cloud Storage - which is a place to store separate files. From a dynamic perspective, the Realtime Database is "dynamic", that is, when there is a change in the database, the Firebase server will immediately send Events to the devices, while Cloud Storage does not. From a capacity perspective, with the Spark package, you only have 1GB for Realtime Database, while you have up to 5GB for Cloud Storage. Cloud Storage has a set of libraries for you to simply apply to your projects. You don't need to write any code to make APIs, to handle events when users send requests, upload or download files at the server unless that is necessary. And if you are planning to work with Google Cloud Platform, specifically Google Cloud Storage, then Firebase Cloud Storage is the top choice in the list of services that you need to refer to.

3.2.4 Firebase Hosting

Firebase Hosting is an API platform that acts like a real database hosting service on the Cloud platform - cloud computing. Integrated and used at the same time with Google's own powerful server system. You may know that the main function of Firebase Hosting is to help programmers simplify many database operations when setting up the Website structure. When Firebase Hosting is integrated into the system, you can save a lot of time and effort to design the backend for your website. Particularly, Firebase in general and Firebase Hosting is also considered a versatile database storage service with excellent data security capabilities. In the world, Firebase Hosting is one of the top choices, it can be said that Trend Hosting has many integrated Web sites for its website system. For example, when you want to develop an instant synchronous online chat function like messaging Facebook, the programmer only needs to develop on the user side the remaining functions when integrated. to the Website. You just need to call the properties of the API and you can connect and use it right away.

How Firebase Hosting Works

You can simply understand that Firebase Hosting provides hosting and data sharing services to users through SSL security technology standards from the CDN network development system right on the Cloud platform - cloud computing to users. real use. With this security standard, you can simply understand how it works through a CDN that helps Firebase Hosting to optimize data more with a system of cloud servers located around the world. End users can easily access with faster speed and better stability and use services on your website when Firebase Hosting is integrated.

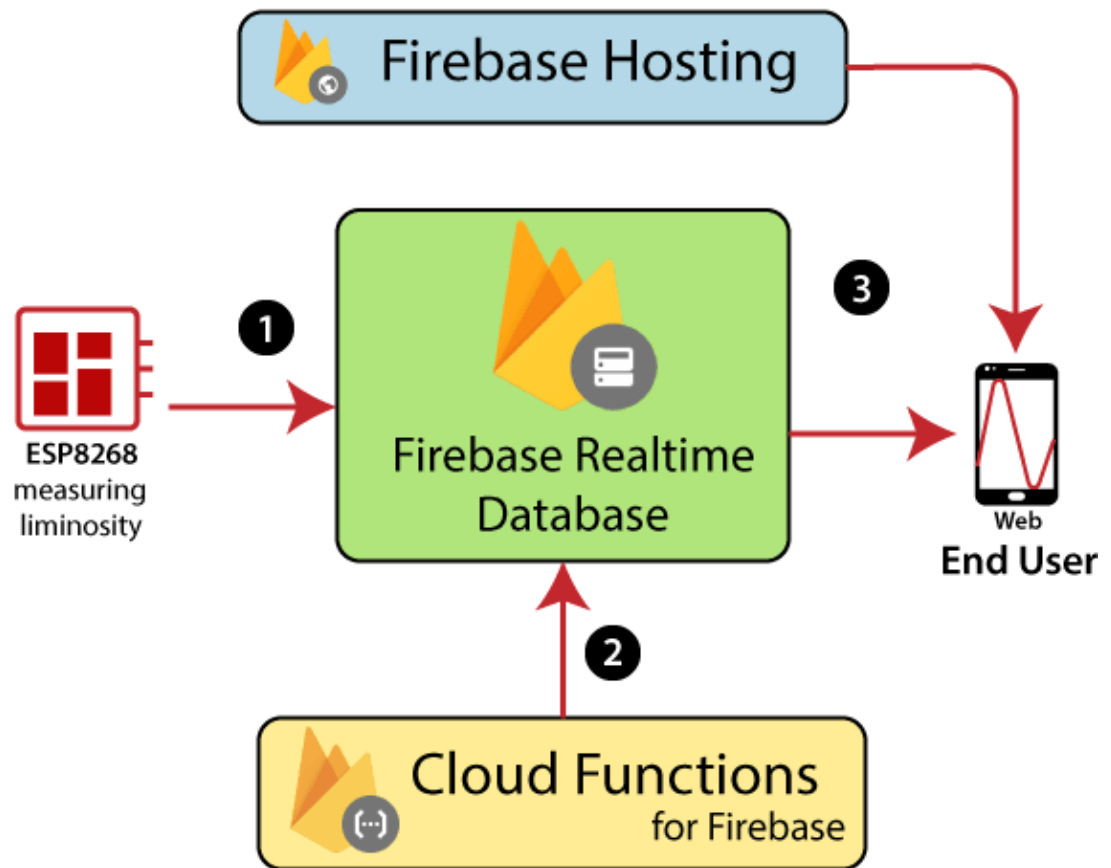


Figure 8. Firebase Hosting usage flow

As illustrated in figure 8, Firebase Hosting usage can deploy files to our hosting server from a local directory on our PC by using the Firebase CLI command-line interface. In addition to delivering static material, we can host Microsoft on our sites using cloud runs or cloud functions. The nearest edge server on the global CDN is used to distribute all material via an SSL connection. Firebase's lightweight hosting choices allow us to construct complex PWAs with ease. Custom headers and URL rewriting for client-side routing are simple to implement.

Firebase provides a variety of domain and subdomain choices for serving our content:

- By default, free subdomains are provided for all Firebase projects on the firebaseapp.com and web.app domains. There is no difference in the content and configuration of the two.

- If we have numerous apps and sites that use the same Firebase project resources, we can establish multiple sites (for example, if we have an admin panel, blog, and public app).
- A Firebase-hosted site can be linked to our own domain name.

Firebase automatically configures SSL certificates for all our domains, ensuring that all our content is presented in a secure manner.

3.2.5 Performance Monitoring

Performance Monitoring detects and automatically corrects problems with system performance. Network latencies affect the app's performance; therefore, this feature analyses HTTP/S requests and evaluates other elements such as waiting time to load and response times or the size of the payload. Adding several codes allows developers to keep tabs on the original source of the problem. Issues have been broken down by nations, devices, and versions, allowing the developer to get to the heart of the problem. Using the Performance Monitoring SDK, the time it takes for the app to open to the point where it responds automatically is recorded. Custom trace is another report that links codes from the client. Below figure could show you more about what the Firebase Performance Monitoring looks like

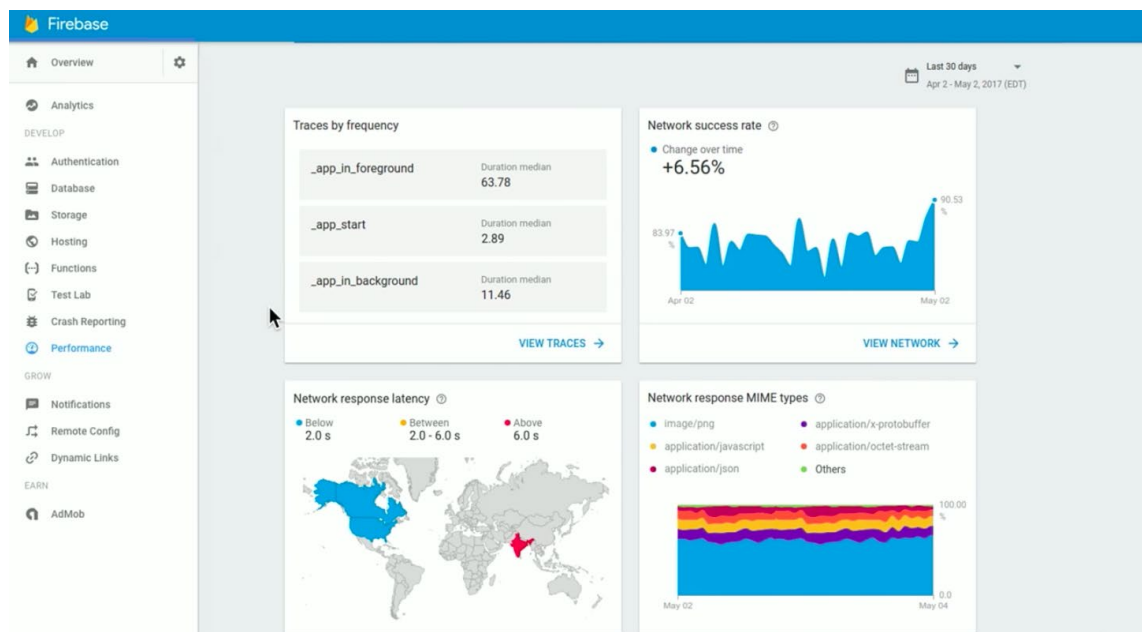


Figure 9. Example of Firebase Performance Monitoring

Performance monitoring SDK is integrated into the application at no cost. Then, features like as metrics and custom traces are selected for background execution. App performance can be improved by utilizing Firebase's Performance Monitoring SDK and custom traces and metrics defined for your app, as well as monitoring performance statistics in the Firebase console during the app's development cycle.

4 Web application project

4.1 Overview and project flow

The project is a full stack e-commerce application using React, Redux and Firebase.

This chapter explains the fundamental steps of developing process, and the outcome of the project – a high-performance e-commerce webstore consists of five primary pages:

The Authentication page is the initial page of the whole website system, it requires the user to provide the login credentials, email and password as authenticators. Additionally, if the credentials are not yet registered to the system yet, users are needed to sign up for their information. Authentication is a must in order to access the entire site.

The Homepage is the next destination after the authentication was conducted. Homepage allows users to perform multiple tasks such as seeing thumbnail pictures of the products, adding a product to the shopping cart or seeing the product details in subpages. In Homepage, the users can also navigate to other pages like Cart and Orders, or even logging out the shopping system.

The Cart page will display all the selected products by the user and provide the total cost of those items. Customer could also delete unwanted items from the cart. After checking the cart, placing the order is the ultimate destination of the whole shopping process, you need to provide your address in order to place the order.

Order page summarizes all order places by the current user.

The last page would be the Admin Panel page, it can only be accessed by the administrators. With this panel, the admin users have the right to add new product, edit product's details or even delete a product. Beside managing the products, the panel also shows all orders placed of all the system in Order tab.

4.2 Setting up the develop environment

Before developing the program, the environment must be established. The first tool is NodeJS version 13.12.0. NodeJS could be used on both the client and server. We choose Visual Studio Code is one as our "source-code editor" due to its graphical user interface and snippets that make writing easier and more convenient, such as ES7 React/Redux. In the other hand, Firebase CLI 10.5.0 helps viewing, managing and deploying the project with several tools. Finally, developer can establish the connection between Firebase Cloud and the local.

4.3 Structuring the project

The below figure will show how my project was structured

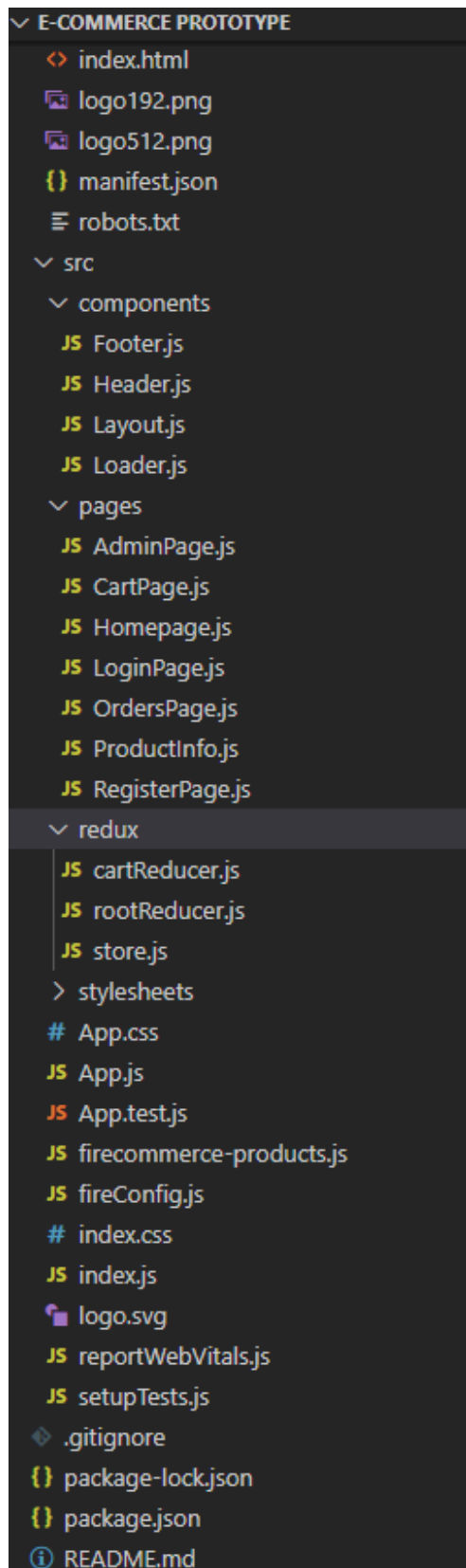


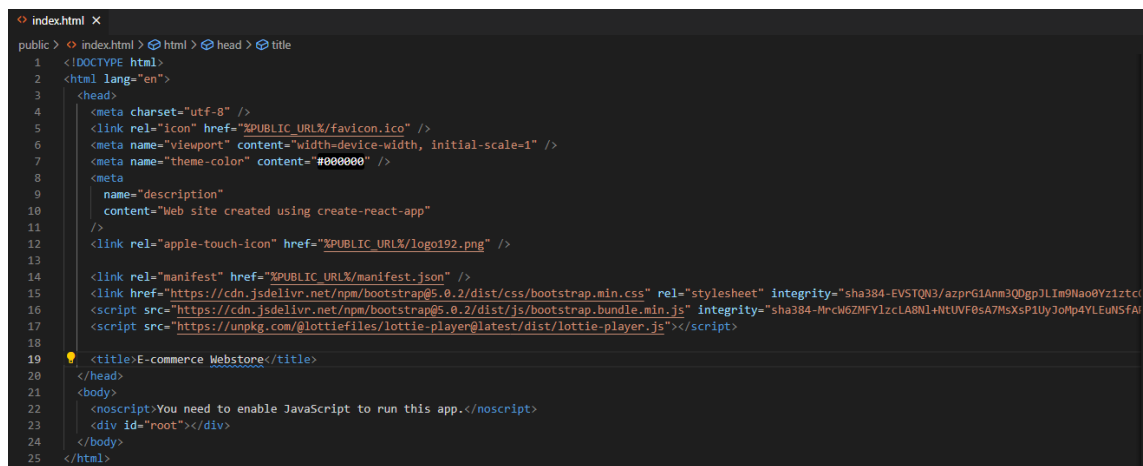
Figure 10. Project structure

There are four main parts in the project structure, which can be shown in the Figure 10.

- HTML: index.html.
- CSS: index.css and App.css.
- JavaScript: App.js, other components and redux.
- Firebase: fireConfig.js.

4.4 Application elements

4.4.1 Index.html



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6   <meta name="viewport" content="width=device-width, initial-scale=1" />
7   <meta name="theme-color" content="#000000" />
8   <meta
9     name="description"
10    content="Web site created using create-react-app"
11  />
12  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13
14  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
15  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgp7LIm9Nao8Yz1ztc" />
16  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MmCWZMFY1zCL88N1+HtUUVF8sA7MsXsP1Uy3oM4YLEUNSF" />
17  <script src="https://unpkg.com/@lottiefiles/lottie-player@latest/dist/lottie-player.js"></script>
18
19  <title>E-commerce Webstore</title>
20 </head>
21 <body>
22   <noscript>You need to enable JavaScript to run this app.</noscript>
23   <div id="root"></div>
24 </body>
25 </html>

```

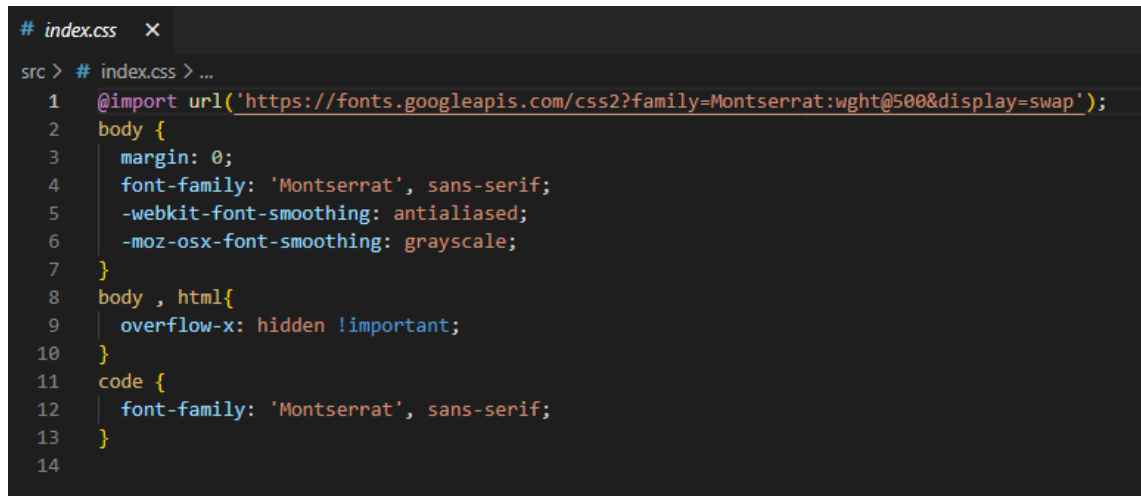
Figure 11. Screenshot of index.html

Figure 11 represents the HTML file's core component. ".js files" include definitions for all the major functions. The external CSS sources used in this project are stored in this file. "root" is the id for the web app.

4.4.2 Styling with CSS

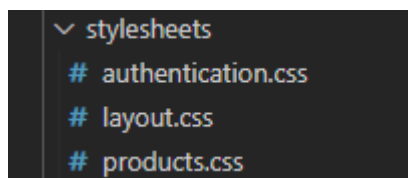
A cascading style sheet (CSS) file is used to format the information on a webpage. It includes configurable global properties for displaying HTML components. For instance, CSS files can specify the size, colour, font, line spacing, indentation, borders, and positioning of HTML elements. This project

includes following styling file: index.css, layout.css, authentication.css and product.css.

A screenshot of a code editor showing the content of the index.css file. The editor has a dark background with light-colored text. The file name 'index.css' is visible in the top left corner. The code includes an @import statement for a Google Fonts link, followed by CSS rules for the body and code elements, all using the Montserrat font family.

```
# index.css X
src > # index.css > ...
1  @import url('https://fonts.googleapis.com/css2?family=Montserrat:wght@500&display=swap');
2  body {
3      margin: 0;
4      font-family: 'Montserrat', sans-serif;
5      -webkit-font-smoothing: antialiased;
6      -moz-osx-font-smoothing: grayscale;
7  }
8  body , html{
9      overflow-x: hidden !important;
10 }
11 code {
12     font-family: 'Montserrat', sans-serif;
13 }
14
```

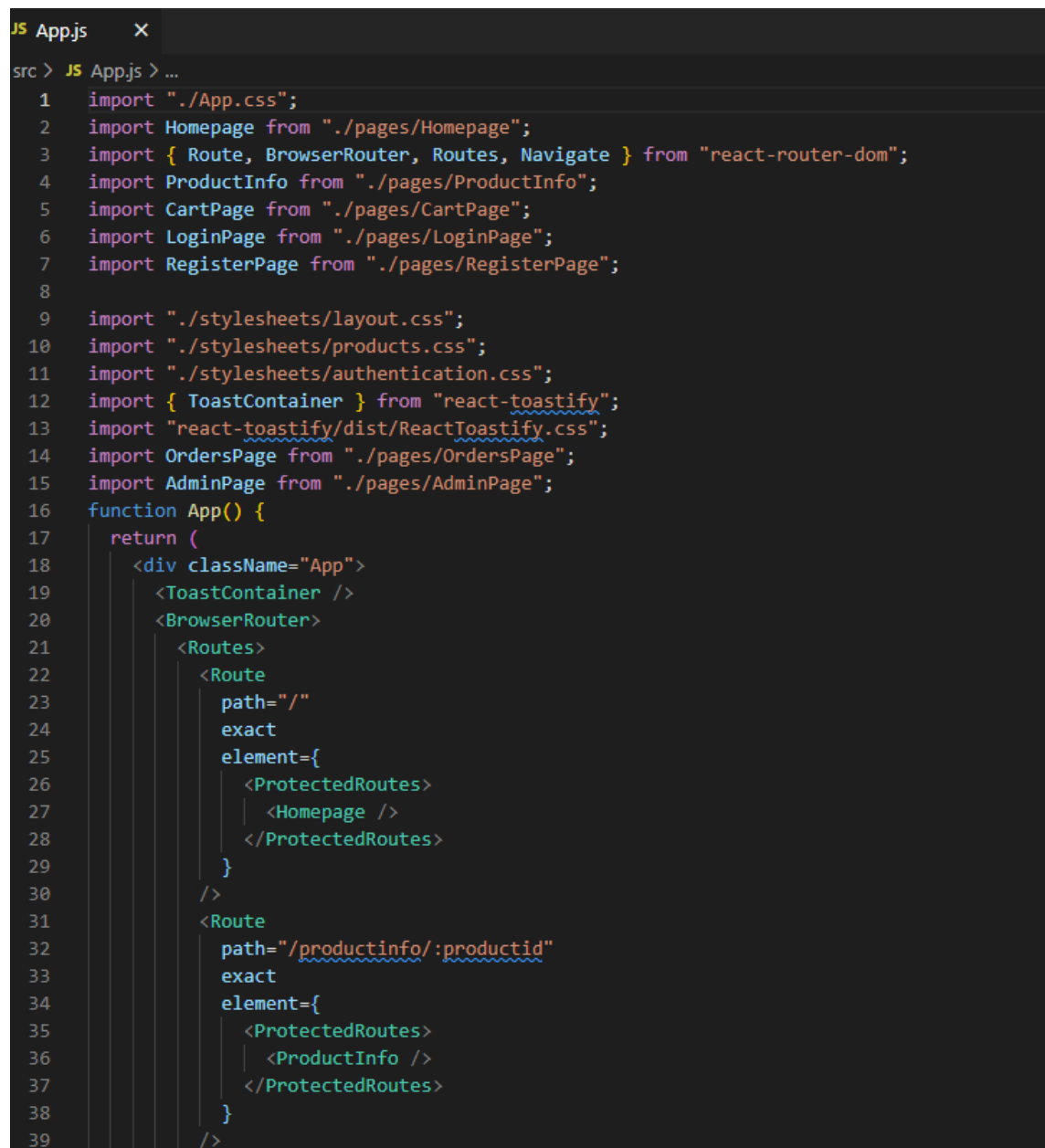
Figure 12. Screenshot of index.css file

A screenshot of a file explorer showing a folder named 'stylesheets'. The folder is expanded, revealing three files: authentication.css, layout.css, and products.css. Each file name is preceded by a hash symbol (#).

```
▼ stylesheets
# authentication.css
# layout.css
# products.css
```

Figure 13. Screenshot of stylesheets folder

4.4.3 JavaScript and components



```

JS App.js  X
src > JS App.js > ...
1  import './App.css';
2  import Homepage from './pages/Homepage';
3  import { Route, BrowserRouter, Routes, Navigate } from 'react-router-dom';
4  import ProductInfo from './pages/ProductInfo';
5  import CartPage from './pages/CartPage';
6  import LoginPage from './pages/LoginPage';
7  import RegisterPage from './pages/RegisterPage';
8
9  import './stylesheets/layout.css';
10 import './stylesheets/products.css';
11 import './stylesheets/authentication.css';
12 import { ToastContainer } from 'react-toastify';
13 import 'react-toastify/dist/ReactToastify.css';
14 import OrdersPage from './pages/OrdersPage';
15 import AdminPage from './pages/AdminPage';
16 function App() {
17   return (
18     <div className="App">
19       <ToastContainer />
20       <BrowserRouter>
21         <Routes>
22           <Route
23             path="/"
24             exact
25             element={
26               <ProtectedRoutes>
27                 <Homepage />
28               </ProtectedRoutes>
29             }
30           />
31           <Route
32             path="/productinfo/:productid"
33             exact
34             element={
35               <ProtectedRoutes>
36                 <ProductInfo />
37               </ProtectedRoutes>
38             }
39           />

```

Figure 14. Screenshot of App.js

Figure 14 shows that the application consists of various parts such as Homepage, Authentication including Login and Register pages, Product Info, Cart, Orders and Admin page.

React-router-dom helps navigating more accurate and faster.

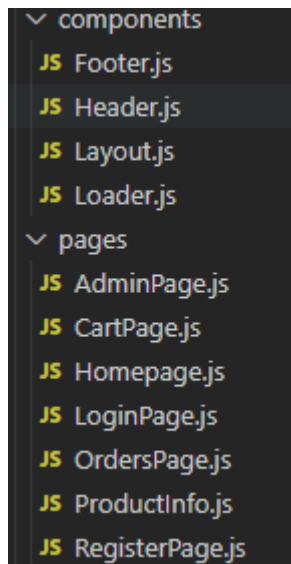


Figure 15. Screenshot of components and pages

Figure 15 displays the subsequent list of additional components. Regarding the website's layout, there are three major components: Footer, Header, and Layout. The page's header is shown by using Header component. Accordingly, the page's footer can be displayed by using Footer component. Layout component helps combining the Header and Footer, also content that is provided in the Layout. Loader component is basically the animated spinner that is shown when loading assets. CartPage component provides the list of ready-to-checkout products. Using Homepage component, customers can see a list of things with accompanying photographs, as well as filter products by name and category. OrdersPage is the page where users can view their placed orders. By instance, the ProductInfoPage views zoomed pictures and detailed of the product, including its name, category, and price. Login is the initial page of the website, requiring a valid email and password. Users can redirect to Register page to create a new account by providing an email valid address, password, and re-entered password.

4.4.4 Redux

For JavaScript applications, Redux is a reliable state management solution. It aids in the development of programs that are reliable, run in a variety of settings (client, server, and native), and are simple to test. The Elm language and Facebook's Flux architecture were the inspiration for Redux. As a result, Redux is frequently used in collaboration with React.

The working way of Redux is very simple. There is a central "store" that holds the entire state of the application. Each component can access the stored state without having to send it from one component to another. There are three things need to be constructed: actions, store, and reducers.

In this project, Redux is mainly used to manage the loading, adding and deleting state of the products in store and cart. Here are some illustrations of the way Redux was used in the project.

A screenshot of a code editor window titled 'JS store.js 1 X'. The editor shows the following code:

```
src > redux > JS store.js > ...
1  import { createStore, applyMiddleware } from "redux";
2  import { composeWithDevTools } from "redux-devtools-extension";
3  import rootReducer from "../rootReducer";
4
5  const composeEnhancers = composeWithDevTools({});
6
7
8  const initailStore = {
9    cartReducer : {
10      cartItems : JSON.parse(localStorage.getItem('cartItems')) ?? []
11    }
12  }
13
14  export const store = createStore(rootReducer, initailStore , composeEnhancers());
15
```

Figure 16. Screenshot of store.js redux file

```
JS cartReducer.js X
src > redux > JS cartReducer.js > ...
1  const initialState = {
2    cartItems: ['item1'],
3  };
4
5  export const cartReducer = (state = initialState, action) => {
6    switch (action.type) {
7
8      case 'ADD_TO_CART' : {
9        return{
10          ...state ,
11          cartItems : [...state.cartItems , action.payload]
12        }
13      }
14      case 'DELETE_FROM_CART' : {
15        return{
16          ...state ,
17          cartItems : state.cartItems.filter(obj=>obj.id !== action.payload.id)
18        }
19      }
20      default:
21        return state;
22    }
23  };
24
```

Figure 17. Screenshot of cartReducer.js redux file

```
JS rootReducer.js X
src > redux > JS rootReducer.js > ...
1  import { combineReducers } from "redux";
2  import { cartReducer } from "./cartReducer";
3  const rootReducer = combineReducers({
4    cartReducer : cartReducer,
5  })
6
7  export default rootReducer
```

Figure 18. Screenshot of rootReducer.js redux file

4.5 Configuring Firebase

Firebase cloud services including Authentication, Firestore and Hosting was integrated in our project. In order to add Firebase to your project, you need to follow these steps:

- Create a Firebase project and register your app
- Install the SDK and initialize Firebase
- Access Firebase in your app

The configuration in your local files is well illustrated in Figure 19.

```
1  import { initializeApp } from "firebase/app";
2  import { getFirestore } from "firebase/firestore";
3
4  const firebaseConfig = {
5    apiKey: process.env.REACT_API_KEY,
6    authDomain: process.env.REACT_AUTH_DOMAIN,
7    projectId: process.env.REACT_PROJECT_ID,
8    storageBucket: process.env.REACT_STORAGE_BUCKET,
9    messagingSenderId: process.env.REACT_MESSAGING_SENDER_ID,
10   appId: process.env.REACT_APP_ID,
11   measurementId: process.env.REACT_MEASUREMENT_ID
12 };
13
14 // Initialize Firebase
15 const app = initializeApp(firebaseConfig);
16
17 const fireDB = getFirestore(app)
18
19 export default fireDB
```

Figure 19. Screenshot of fireConfig.js

4.6 User interface

The following figures illustrate the outcome of the project – a webstore prototype with simple user interface.

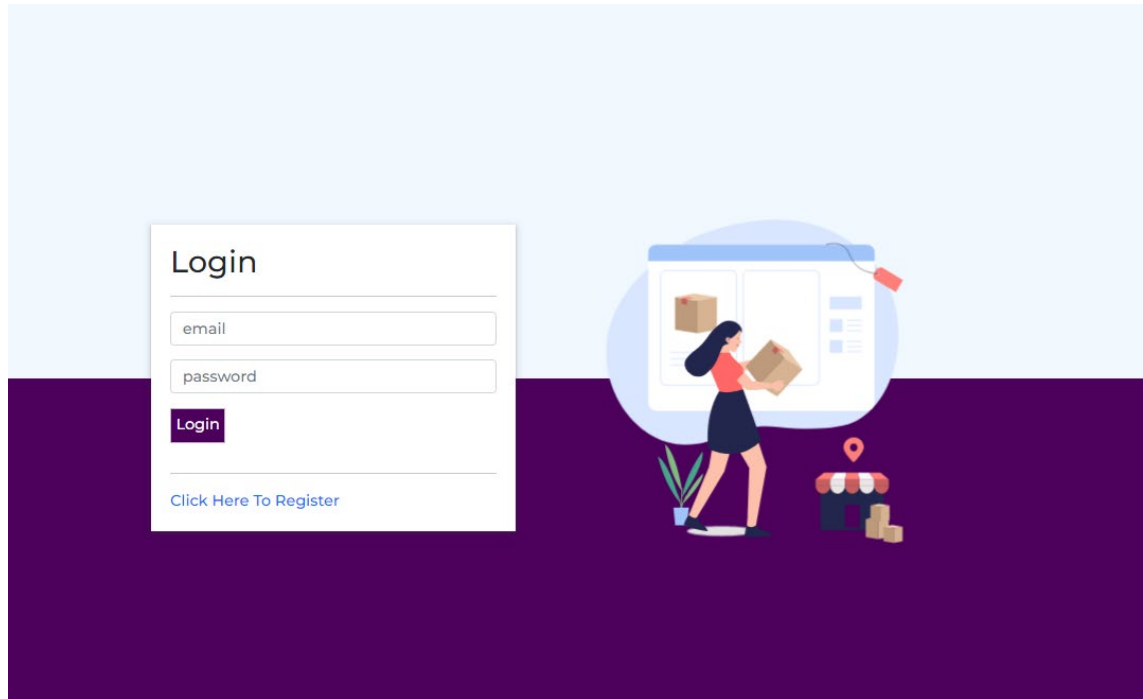


Figure 20. Screenshot of Login page

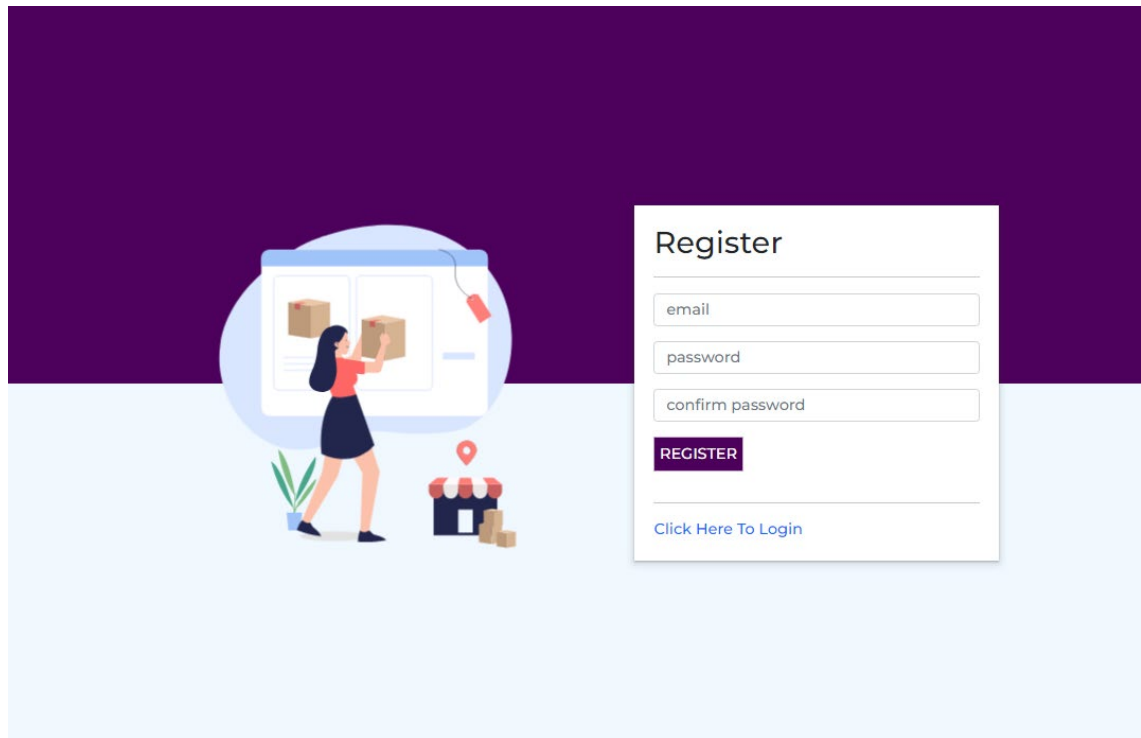


Figure 21. Screenshot of Register page

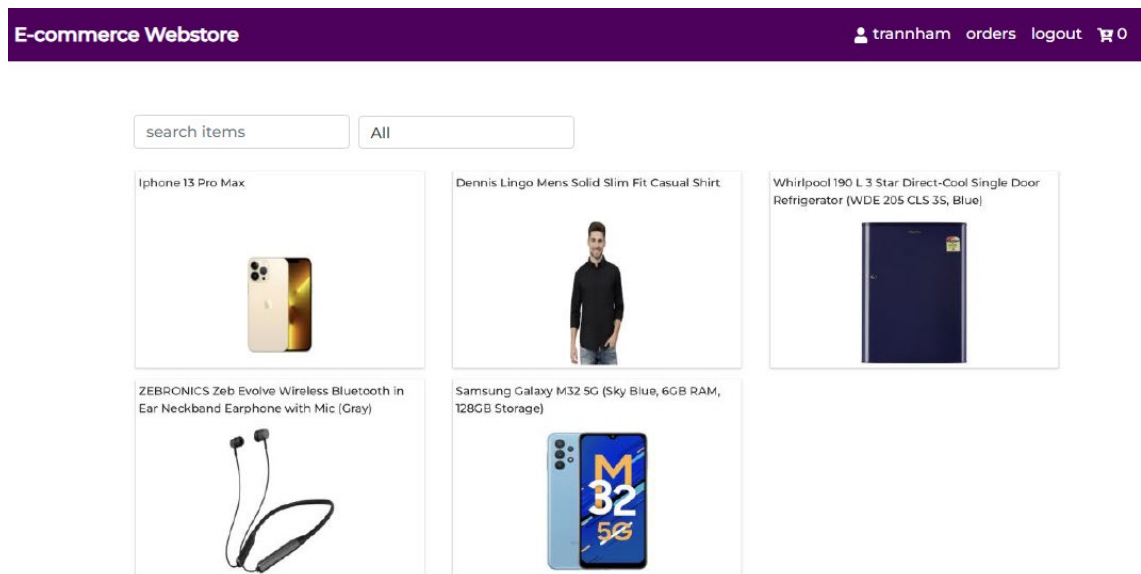


Figure 22. Screenshot of Homepage





E-commerce Webstore				trannham	orders	logout	0
Image	Name	Price	Action				
	Dennis Lingo Mens Solid Slim Fit Casual Shirt	850					
	ZEBRONICS Zeb Evolve Wireless Bluetooth in Ear Neckband Earphone with Mic (Gray)	400					
			Total Amount = 1250 RS/-		PLACE ORDER		

Figure 23. Screenshot of Cart page





E-commerce Webstore				trannham	orders	logout	0
Image	Name	Price					
	Dennis Lingo Mens Solid Slim Fit Casual Shirt	800					
	Samsung 198 L 4 Star Inverter Direct Cool Single Door Refrigerator	15000					
	Lenovo 300 Wireless Compact Mouse (GX30K79401)	680					
Image	Name	Price					
	ZEBRONICS Zeb Evolve Wireless Bluetooth in Ear Neckband Earphone with Mic (Gray)	600					

Figure 24. Screenshot of Order Page
















E-commerce Webstore				trannham	orders	logout	0
Products		Orders Users					
Products List							ADD PRODUCT
Image	Name	Category	Price	Action			
	Dennis Lingo Mens Solid Slim Fit Casual Shirt	fashion	740				
	THE ARCHER Men's Slim Fit T-Shirt (Pack of 2)	fashion	580				
	ZEBRONICS Zeb Evolve Wireless Bluetooth in Ear Neckband Earphone with Mic (Gray)	electronics	400				
	Samsung 198 L 4 Star Inverter Direct Cool Single Door Refrigerator	electronics	13000				
	OnePlus Nord CE 5G (Charcoal Ink, 8GB RAM, 128GB Storage)	mobiles	30000				

Figure 25. Screenshot of Admin Panel

4.7 Deploying the project with Firebase Hosting

Installing the Firebase Hosting will enable developers to deploy the application, and then sign into Firebase from your terminal to get started. If this is the first time developing, you will be required to provide email and password before proceeding. Additionally, you must check the root directory whether if your React application has already been configured for using Firebase.

Throughout the configuration process, you need to answer a series of questions. Choose “Hosting: Configure and deploy Firebase Hosting sites” for the first question. Moving to project setup questions, you should select “Use an existing project” and select your desired project by the correct name. For the final, hosting setup, “build” should be entered because Firebase will look for assets to deploy in this directory. You should also say N (No) for both “Configure as a single-page app” and whether to overwrite the existing build/index.html file.

If everything is done correctly, .firebaserc, firebase.json file should be generated in the directory.

Now all that remaining is to deploy to Firebase Hosting. Simply use the command “firebase deploy”, then let Firebase take care of everything.

4.8 Testing

It's also essential to test the application to ensure that it works and is compatible with the great majority of browsers. Refer to being bug-free, the online material is also examined for grammatical mistakes.

Every item was examined, items were seen with pictures, the products were filtered according to their name and category. Preferred items were added to and removed from the cart as necessary. The system operates perfectly in every way. There were no issues with the examination.

The test was done on a variety of platforms, including Android, iOS, and Windows. All the tested mobile devices could connect to the webstore without any issues, indicating a positive outcome. It seems that all images, buttons, and other design elements are shown correctly on the screen.

4.9 Further Discussion

Many programming languages, such as .NET, Java, PHP, and JavaScript, facilitate the building of a website as a result of technical improvements. Numerous website builders, such as Wordpress and OpenCart, offer premade website themes to consumers with no coding experience. However, these templates frequently cannot be modified freely as desired by developers. Another significant disadvantage is the expensive service price.

Using JavaScript is beneficial in many ways: fast speed, simplicity, browser compatibility and reduce the server load. JavaScript is independent from external resources and servers. Thanks to the fully integration with HTML and CSS, JavaScript could be implemented in any website in few seconds.

React is known as a lightweight framework for developing web applications due to its efficiency. A memory-based data structure cache is established by React. React detects and updates the browser whenever the source code is updated. This unique function will enhance the efficiency of the webpage.

Because of the lack of time, real payment and invoicing could not be implemented in this project. But it is recommended to add these features in future development. Subscribing to the newsletter for promotion and discount is also a promising developing option. When it comes to E-commerce, website visibility and online marketing play a crucial role in the success of the business. One of the good ways to boost the marketing is to optimize the search engine. Additionally, we can consider making the webstore available in different languages than English to target more customer sectors.

5 Conclusion

The goal of the thesis and implementation project is to conduct a friendly, responsive and promising E-commerce webstore prototype with the help of React and Firebase services. This thesis project could be seen as successful because the website is fully functional.

In this webstore, users could perform various actions such as viewing the collections of items with thumbnails, adding and editing the product items, putting items to shopping cart and placing order.

For frontend technology, following by HTML/CSS, React has help to build a basic but useful user interface. Redux was used in state managements. Firebase services plays a vital role in building a serverless backend

Deploying and testing was conducted in different platforms. The outcome was positive because the webstore satisfies all the listed requirements.

In short, the web application prototype can fulfil all basic needs of an e-commerce webstore with a low cost. Therefore, it can help small companies a sustainable way to expand their businesses.

References

1. 2022. Key Internet Statistics to Know in 2022 (Including Mobile). Online. 5 March 2022. <[Key Internet Statistics to Know in 2022](#)>. Accessed 25 April 2022.
2. Walburg, Marlana. 2021. Is React Native good? Advantages and disadvantages. Online. 8 January 2021. <[Is React Native good? Advantages and disadvantages](#)>. Accessed 23 April 2022.
3. Sheth, Kishan. 2021. What is Virtual DOM? How Virtual DOM works? What is Reconciliation? What is diffing algorithm? What makes React so fast? Online. 4 May 2021. <[What is Virtual DOM? How Virtual DOM works? What is Reconciliation? What is diffing algorithm? What makes React so fast?](#)>. Accessed 23 April 2022.
4. Singh, Vrijraj. 2018. Introduction to Firebase. Online. 12 December 2018. <[Introduction to Firebase](#)>. Accessed 25 April 2022.

5. Firebase hosting. Online. <Firebase hosting>. Accessed 25 April 2022.
6. ElHousieny, Rany. 2021. What Is Redux? Online. 30 January 2021. <What is Redux?>. Accessed 25 April 2022.
7. Fayock, Colby. 2022. How to Build a React E-Commerce Web App. Online 21 March 2022. <How to Build a React E-commerce Web App>. Accessed 15 April 2022.

Figures

Figure 1. DOM Working flow in React	6
Figure 2. Defining a React component	7
Figure 3. Firebase concept	9
Figure 4. Firebase-supported 3rd party logins	10
Figure 5. Data stored in real-time database.	11
Figure 6. Realtime syncing illustration	11
Figure 7. Firebase Cloud Storage example	13
Figure 8. Firebase Hosting usage flow	15
Figure 9. Example of Firebase Performance Monitoring	17
Figure 10. Project structure	19
Figure 11. Screenshot of index.html	20
Figure 12. Screenshot of index.css file	21
Figure 13. Screenshot of stylesheets folder	21
Figure 14. Screenshot of App.js	22
Figure 15. Screenshot of components and pages	23
Figure 16. Screenshot of store.js redux file	24
Figure 17. Screenshot of cartReducer.js redux file	25
Figure 18. Screenshot of rootReducer.js redux file	25
Figure 19. Screenshot of fireConfig.js	26
Figure 20. Screenshot of Login page	27
Figure 21. Screenshot of Register page	28
Figure 22. Screenshot of Homepage	28
Figure 23. Screenshot of Cart page	29
Figure 24. Screenshot of Order Page	29
Figure 25. Screenshot of Admin Panel	30