Developing an Investment Tracker Web App

By: Athul Raj Nambiar

S4126351

University of Groningen, Honours College

Individual Theme Study, Subject: Web Engineering

TABLE OF CONTENTS

Contents

PURPOSE OF THE INDIVIDUAL THEME STUDY

Purpose of the Individual Theme Study

LEARNING OUTCOMES

In depth study of a topic related to one of the regular degree programme courses or an Honours College FSE course.

GOAL OF THIS PROJECT

The specific goal of the individual theme study is: "After following a course in either the regular degree programme or the Honours College FSE programme, a student may complete an additional individual assignment on a particular topic covered in one of these courses." [13]. For my project specifically I will be completing an additional individual assignment for the course Web Engineering. This assignment would be considered additional to the course because it will specifically look at some of the technologies used to create a functional website, and focus on how to create a basic Web App with useful functionalities that we see on modern websites. This deviates from the scope of the Web Engineering course as for the course our assignment was to explore the application of a REST API and when I did the course, little importance was given for the design of the Web App itself, only that we showed understanding of REST API. Furthermore in the course we worked as a group of 5 so I did not gain much knowledge on all the aspects of creating a Web App. With the project I will:

- Be able to understand what technologies will be used to create a basic Web App
- Learn how to use API's to obtain information and display them on my Web App
- Understand the technologies used to make a Web App more visually appealing
- Understand the technologies that make a Web App more user friendly
- Learn how to implement authentication measures for a Web App
- Connect a database to a Web App to store information and access it when necessary

PURPOSE OF THE INDIVIDUAL THEME STUDY

MAIN IDEA

The idea behind this project is to develop a basic investment tracker using an API that I will use to retrieve data from and display this information. I want this to have the basic functionality that a user is able to search for their desired crypto investment, store how many coins they bought and at what price of a single coin was when they made a purchase. For this project it will only be crypto assets because I found a free API and because all other investment API's I found usually had complicated documentation or had a limit on how many HTTP requests could be made without paying for the service. Based on this information Web App will make HTTP requests and get information for that coin and display it on a data table. Coins can be added to this table so that the user can make a small portfolio of all the coins they want. This portfolio can be saved in some way, so that when the user comes back to the website, they can see their portfolio again. Some sort of update of the values occur (since the price of investments changes). This change should somehow be indicated on the table for example with one of the values changes when reloaded. With this as the main ide of my project, it lead me to the research question: *How do can I develop a Web App using an API, in order to present information about selected crypto assets?*

Research of Technologies

WEB DEVELOPMENT UDEMY COURSE

To prepare the foundational understanding of developing a Web App, I took an online course for Web Development on Udemy. With this course I learned about the fundamental technologies used to create a good Web App (specifically HTML, CSS, Bootstrap, Node.js with Express.js, MongoDB and some basic React) [1]. These technologies are very beneficial to developing a good Web App because they focus on individual aspects of the Web App but work coherently in order to make the App more robust, functional, user friendly and well performing. I will cover some of these individual technologies in more detail. Along with the technologies, the course taught me how to layout and structure my code for readability.

HTML

HTML just stands for hypertext markup language and it is just the standard markup language for creating a website. I use it in combination with the other technologies but it is basically used to outline and create the divided structure of the webpage. Usually the layout of a Web App is to create HTML file that contains HTML code. However with the user of React that is not necessary. In my case I used react components that contains HTML code and creates the structure of the overall website and the HTML elements within my JavaScript files and I render these React components [1].

CSS

CSS stands for Cascade Style Sheets and the purpose of it is to make changes to the visual aspects of the Web App (colour, size, styles, etc.) by styling HTML documents. This can also be done with HTML alone, but with CSS it makes it easier for a user to make stylistic modifications too many elements of the Web App by assigning the elements to certain classes and making stylistic changes without too much redundant code in the HTML. Therefore CSS is a way to describe how HTML elements should be displayed when I load the Web App [1].

RESEARCH OF TECHNOLOGIES

BOOTSTRAP

Bootstrap is a framework used with HTML, CSS and JavaScript to create well designed and responsive front-end designs and can be really useful for many interface components [1].

NODE.JS

Node.js is a backend framework, which is used for a variety of tasks such as hosting API's, serving HTTP requests and accessing a database [2].

REACT.JS

It is an open-source front-end library. It is very useful when developing UI's for websites and web apps in a very structured manner [2]. For this project I use React to structure the website and it works very well with the authentication measures that has been included. With React, I am able to create a much more dynamic and responsive website that performs really well. It is a very popular technology to learn and is an in-demand skill for front end developers at the moment, so it only benefits my future opportunities to take the time and learn to use this library.

FIREBASE

Firebase was created by google and is a backend development software that has a lot of tools that can be used to create very useful Web Apps. For example it has tools and services such as analytics, authentication, cloud messaging, a real time database, performance and others [3]. I use firebase for its authentication and the database. With firebase I am able to allow authentication functionality to the Web App by allowing the user to register an account and set a password. This allows the user to login to an account and store the information on the real time data base on firebase. It also allows the possibility to reset the user's password if necessary. As mentioned the firebase database will be useful to keep specific information about each person's investments (basically it will be used to store each person's portfolio and will load that portfolio when they log back into the Web App).

RESEARCH OF TECHNOLOGIES

MATERIAL UI

MUI is a library that is used in React applications to improve the user interface as it provides many tools and different design components. I specifically use a table structure from MUI to present all the data of a user's saved portfolio in a table.

COIN GECKO API

I looked at multiple API's such as the Yahoo Finance API and other API's that provided real time updates every few minutes or so. However these required payments and therefore since the sub goal of this project was to work with an API and retrieve asset information from it, I found the Coin Gecko API that gives access to data and statistics on cryptocurrency, such as live prices, trading volume, historical data, crypto categories, images, etc. [4]. So with this API I was able to achieve results that met this assignment's sub goal.

AXIOS API

Axios is an HTTP client library that can be used with React for a variety of HTTP and REST related tasks. With Axios it makes sending asynchronous HTTP requests to REST endpoints a lot simpler and can aid in the use of CRUD operations [8]. In this project it will be used mainly to assist with HTTP requests related to the CoinGecko API in order to retrieve data from the requests.

Summary of the Development Process

SETTING UP AUTHENTICATION

I start off by setting up the web app's Firebase configuration, so that I have access to the firebase functions. By giving my program access to this configuration with a specific API key, I also have access to the Firestore database and the database that stores user authentication information designated for my Web App. All the user information (login details for each user, and their portfolio data) that we retrieve from the UI, will be correctly stored onto the Web App’s firebase account that we have access to with the configuration. Using the libraries in Firebase, I set up the registration, login and sign out functionalities. Using the firebase Auth service interface [5], I managed to add the functionality of creating a new user, logging in, logging out and resetting password. The functions available made it possible by storing the information and passing them through the functions. The backend code that is specifically linked with firebase to allow authentication operations can be seen on *src/contexts/AuthContext.js* [6]. Figure 1 shows an outline of the relationship between the Web App and Firebase based on the code developed in *src/contexts/AuthContext.js*.

SUMMARY OF THE DEVELOPMENT PROCESS

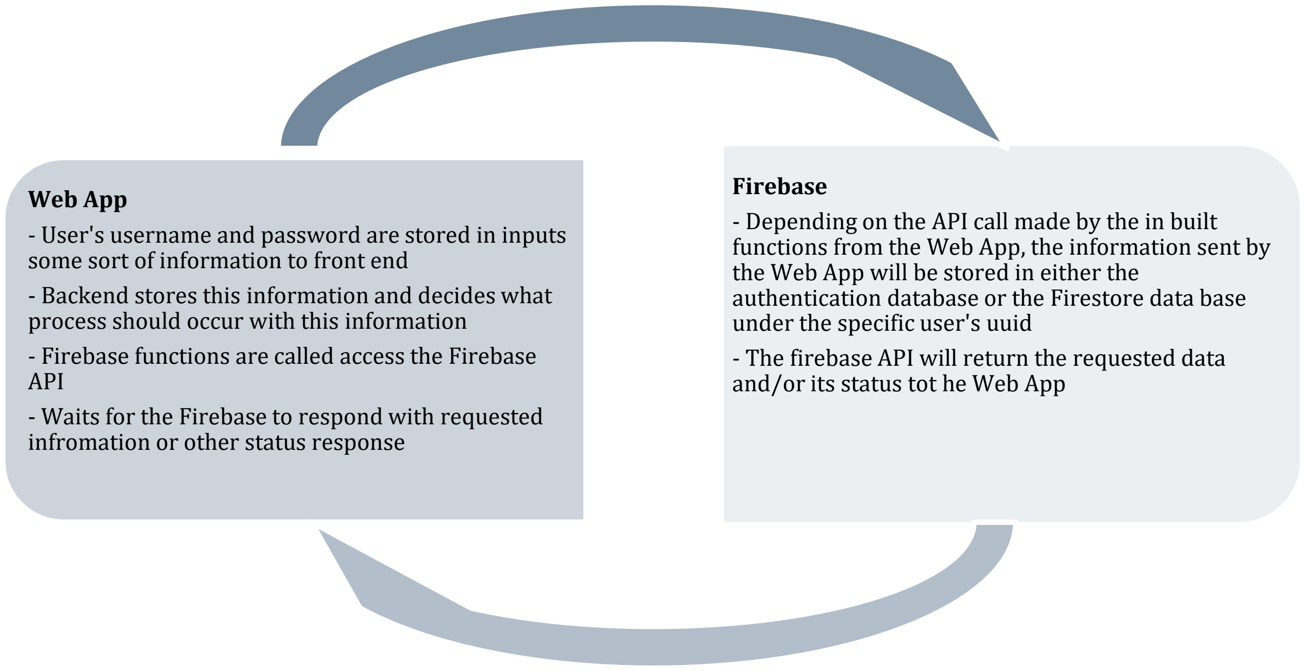

Web App
- User's username and password are stored in inputs some sort of information to front end
- Backend stores this information and decides what process should occur with this information
- Firebase functions are called access the Firebase API
- Waits for the Firebase to respond with requested infromation or other status response
Firebase
- Depending on the API call made by the in built functions from the Web App, the information sent by the Web App will be stored in either the authentication database or the Firestore data base under the specific user's uuid
- The firebase API will return the requested data and/or its status tot he Web App

Figure 1: General outline of code in *src/contexts/AuthContext.js* and how it interacts with Firebase

LOGIN, SIGN UP AND RESET PASSWORD PAGES

The URI's for these pages and the entire app are linked to *src/components/App/App.js*. To create these three pages I created React components that contain Bootstrap elements and it render the components on the Web App based on which URI we call on/ send a request to. The components have access to the firebase functions and so the functions are called on based on the inputs by the user. Once the user inputs their information, the data is stored on the database on firebase and stored with a unique uuid for each user [7]. Figure 2 shows displays how the Log In page is presented to the user. The code for these pages are located on these directories: *src/components/LogIn/, src/components/ResetPassword, src/components/SignUp*

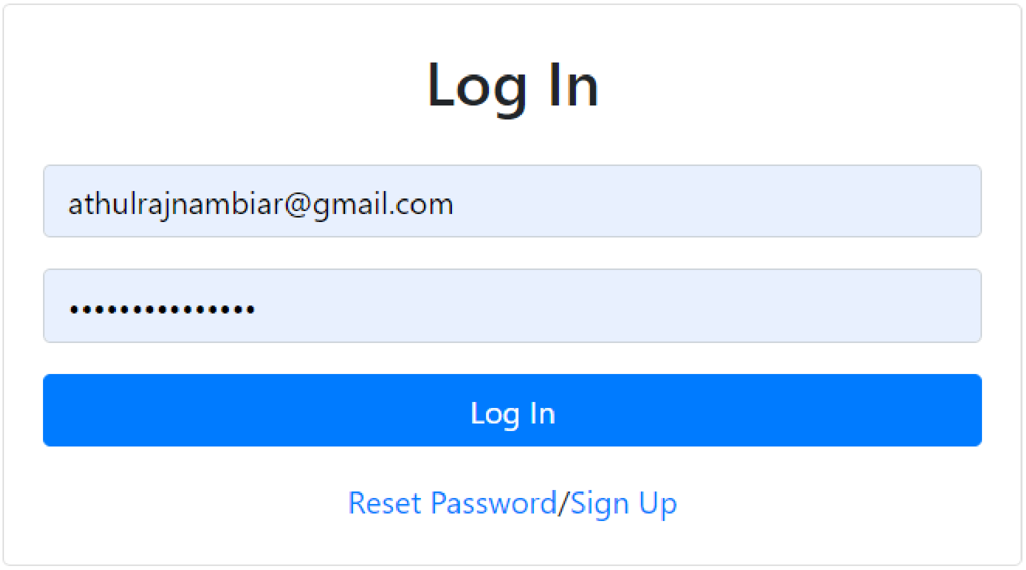


Figure 2: Screenshot of the Login Page, with hyperlinks to Reset Password and Sign Up

SETTING UP NAVIGATION BAR

I set up 2 types of navigation bars using react components. One navigation bar for when a user is logged in and one when a user is not logged in. When the user is logged in, the navigation bar has a button to "sign out", while when there is no user logged in there is a link to "Log In". The navigation bar contains the appropriate hyperlink references and the buttons call on the firebase auth functions. The code for the navigation bar can be found in this directory: *src/components/Navigation*

IMPLEMENTING A SEARCH BAR AND POP UP WINDOW

The search bar and pop up window were also created using React components. Axios is used to make GET requests to retrieve the data that is needed from CoinGecko [9]. By importing axios we have access to functions that can be used to make the specific GET request we want on to

SUMMARY OF THE DEVELOPMENT PROCESS

CoinGecko’s API and axios will return the response of the GET request (the market data we want) in a format that we can store on JavaScript data structures. For the search bar I used React to create a sort of filtered search so that when a user types letters, it filters all the available coin names that come up from the API and presents the filtered list [11]. Figure 3 shows an example when a user inputs the letters “eth” into the search bar, and it filters all the coin names retrieved from the CoinGecko API and outputs a list of coins that have the letters “eth” in the name. In order to make the pop up window, a form is used and it is opened up when the user clicks on an item from the list that appears beneath the search bar. Figure 4 shows the pop up window for when ethereum is selected from the search list. The code is located in this file: *src/components/Home/index.js*

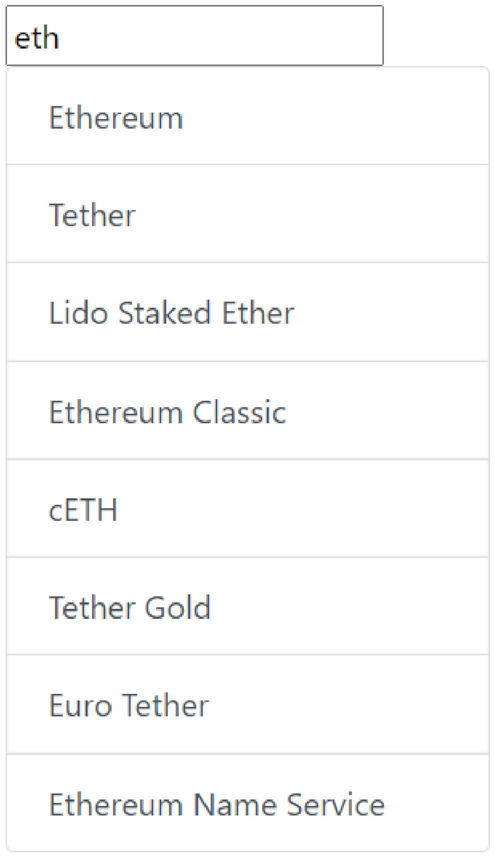
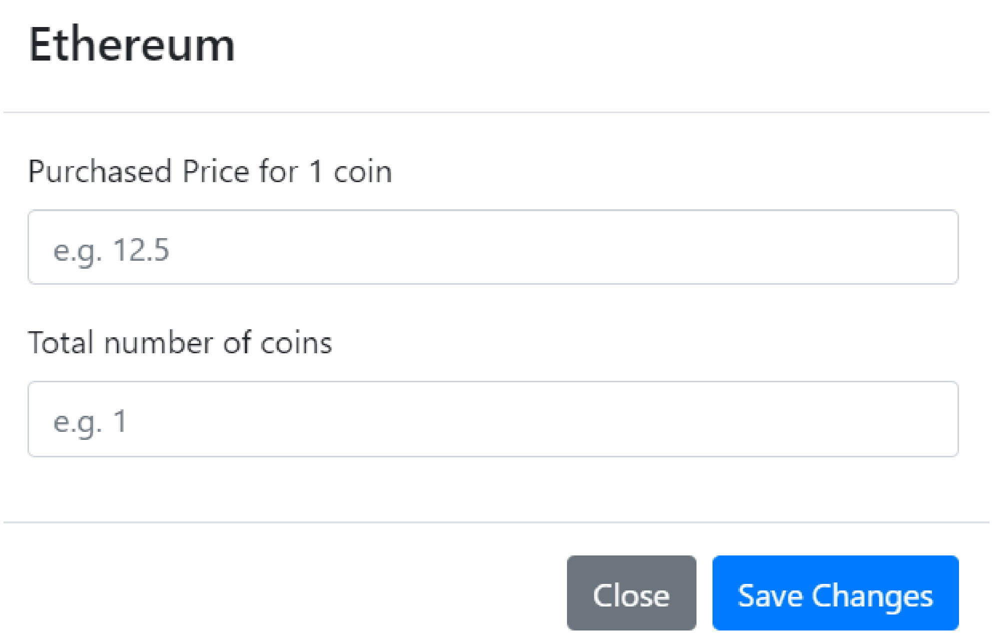


Figure 3: Search bar listing all potential options


Ethereum

Purchased Price for 1 coin

e.g. 12.5

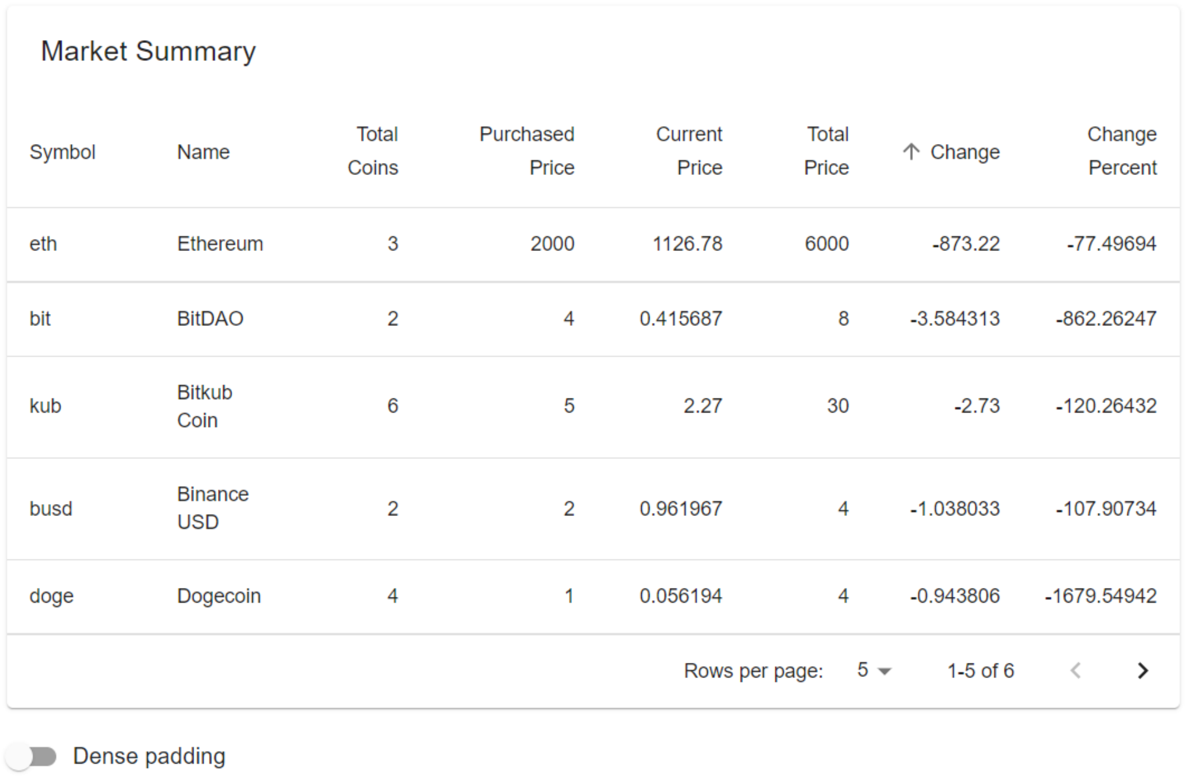
Total number of coins

e.g. 1

Figure 4: Pop up window for ethereum

CREATING A TABLE

The table that presents all the crypto information/the portfolio is made using MUI [12]. I use the same template they provide. This template is very useful as it allows the table to be sorted by each column and contains pagination if we have a large portfolio that we want to see in pages. Figure 5 shows the table of an example portfolio. The code is located in: *src/components/Home/MarketTable.js*



Market Summary

Symbol	Name	Total Coins	Purchased Price	Current Price	Total Price	↑ Change	Change Percent
eth	Ethereum	3	2000	1126.78	6000	-873.22	-77.49694
bit	BitDAO	2	4	0.415687	8	-3.584313	-862.26247
kub	Bitkub Coin	6	5	2.27	30	-2.73	-120.26432
busd	Binance USD	2	2	0.961967	4	-1.038033	-107.90734
doge	Dogecoin	4	1	0.056194	4	-0.943806	-1679.54942

Rows per page: 5 1-5 of 6

Dense padding

Figure 5: Portfolio Table displaying an example portfolio

STORING PORTFOLIO INFORMATION ON FIREBASE

In order to store data (in the case of this Web App, the portfolio data) for a specific user, the uuid is used along with the available Cloud Firestore functions that were imported from firebase to store data into the firebase account that was already configured for this Web App. The code is located in: *src/contexts/FirebaseContext.js*

RELOADING INFORMATION FROM FIREBASE

The data is reloaded to the table when the user first log's in. This is done by retrieving the user's data from the firebase database using the user's unique uuid and storing the values in a dictionary until they are passed onto the table that contains all the information from the portfolio. Furthermore when logging in, a GET request is made to CoinGecko using Axios to retrieve the latest/ up to date information on specific investments [10]. The code is located in these files: *src/contexts/FirebaseContext.js*, *src/components/Home/index.js*

Conclusion

At the beginning of this project I expected to implement a much more impressive UI. However as I went along and got into the research and the coding itself, making small implementations were a lot more complicated and time consuming than expected. Of course, this is very normal in programming, where a programmer expects the work for a set period of time and the project takes longer than expected due to unforeseen challenges. However these challenges were very useful learning tool. I started this project assuming that I will take a closer look into front end, but after I started implementing my ideas, I spent more time and gained a lot more valuable experience working with backend technologies as well. Through this project I managed to get a good grasp of how to work with API's in combination with React and Firebase. The project was very beneficial in terms of improving my fundamental understanding of how a Web App is developed and what available technologies can help build high performing and robust products. From this project I was able to complete all my goals and it has also indicated the direction in which I can focus on improvements. The technologies that were researched and used were ideal for my project as they worked very well coherently. Using React, Firebase and Axios helped the Web App work well together to create authentication measures, store data and make HTTP requests based on each unique user. Overall this was a successful individual theme study that has greatly improved my knowledge of Web Engineering and Web Development.

Improvements for the Future

In terms of improvements there are many possible changes that can be made and especially many additional features I can add to the Web App. All these changes could not be implemented purely based on time restrictions. There are plenty of changes to be made but here are the most notable changes that could be made:

- The divider between the search bar and the investments table can move if the values of the table get too big and the table shifts further down to the bottom of the web page. This should be fixed by finding a way to hold the positions of the individual components.
- CSS should be used to add more colour to the tracker and make a more appealing web page.
- I want to play around with more Bootstrap and MUI features and explore more features that could be added to the front-end side
- Images could have been added for the coin's symbol and placed in the table. This would have been done using the images that were retrievable using the Crypto API.
- Divide the Web App better so that the UI experience is smoother (could be done in a variety of ways such as shift the search bar to the side, fix the size of the table that shows the investments, place the search bar and table in a separate divider and use CSS to make it more visually appealing, etc.). This refers to both cosmetic changes and changes to the JavaScript code so that the containers know how to react appropriately in different situations. Examples of how container reactions would be when the table changes positions when the values are too big and the search bar shifts it. Cosmetic aspects would be for example the positioning/ sizing of elements in the page.
- Add a delete and modify button for the investments already on the table, that perform the action that they are supposed to do
- Modify the content of the table with more details, such as the total price changed (includes all the total price change of all the coins combined)

IMPROVEMENTS FOR THE FUTURE

- Expand using multiple API's so that stock investments can be added to, or real time information can be added so that the investments can be updated in real time when the values change
- Creating the possibility of adding new purchases of the same coin, since someone could buy the same coin again in the future

SOURCES:

Sources:

1. Yu, Dr. Angela. “The Complete 2022 Web Development Bootcamp.” *Udemy*, Udemy, 30 Apr. 2022, https://www.udemy.com/course/the-complete-web-development-bootcamp/. Accessed 11 June. 2022.
2. Tkachenko, Igor. “Node JS vs. React Comparison: Which to Choose for Your JS Project in 2022.” *Node Js Vs. React Comparison: Which to Choose for Your JS Project in 2022*, The App Solutions, 24 Sept. 2019, https://theappsolutions.com/blog/development/node-js-vs-react-js/#:~:text=js%20are%20JavaScript%20technologies%2C%20yet,used%20for%20developing%20user%20interfaces. Accessed 11 June. 2022.
3. Rosencrance, Linda. “What Is Google Firebase? - Definition from Whatis.com.” *SearchMobileComputing*, TechTarget, 25 Apr. 2019, https://www.techtarget.com/searchmobilecomputing/definition/Google-Firebase#:~:text=Google%20Firebase%20is%20a%20Google,creating%20marketing%20and%20product%20experiment. Accessed 11 June. 2022.
4. “Most Powerful Cryptocurrency Data API.” *CoinGecko*, CoinGecko, https://www.coingecko.com/en/api/documentation. Accessed 11 June. 2022.
5. “Auth | Javascript SDK | Firebase.” *Google*, Google, https://firebase.google.com/docs/reference/js/v8/firebase.auth.Auth. Accessed 11 June. 2022.
6. Ivanov, Maksim. “Firebase React Authentication Tutorial for Beginners - Private Route with Hooks.” *YouTube*, YouTube, 5 May 2019, https://www.youtube.com/watch?v=unr4s3jd9qA. Accessed 12 June. 2022.
7. Web Dev Simplified. “REACT Authentication Crash Course with Firebase and Routing.” *YouTube*, YouTube, 10 Oct. 2020, https://www.youtube.com/watch?v=PKwu15ldZ7k. Accessed 11 June. 2022.
8. Olawanle, Joel. “Axios React – How to Make Get, Post, and Delete API Requests.” *FreeCodeCamp.org*, FreeCodeCamp.org, 17 May 2022, https://www.freecodecamp.org/news/axios-react-how-to-make-get-post-and-delete-api-requests/#:~:text=Joel%20Olawanle,js%20server. Accessed 11 June. 2022.
9. Grey, Dave. “REACT AXIOS API Requests | Axios with React JS Tutorial.” *YouTube*, YouTube, 3 Aug. 2021, https://www.youtube.com/watch?v=ZEKBDXGnD4s. Accessed 12 June. 2022.

SOURCES:

10. RoadsideCoder. “Cryptocurrency Tracker with React JS, Material UI and Chart JS Tutorial.” *YouTube*, YouTube, 10 Oct. 2021, https://www.youtube.com/watch?v=QA6oTpMZp84. Accessed 11 June. 2022.
11. Lama Dev. “React Search Filter Tutorial Beginner to Advanced.” *YouTube*, YouTube, 17 Feb. 2022, https://www.youtube.com/watch?v=MY6ZZIn93V8. Accessed 11 June. 2022.
12. “React Table Component - Material.” *Material UI Documentation*, https://v4.mui.com/components/tables/#table. Accessed 11 June. 2022.
13. “Ocasys: Toon Vak Individual Theme Study.” University of Groningen, https://www.rug.nl/ocasys/frw/vak/show?code=HCSE21003. Accessed 16 June. 2022.