2023

Phan Thao Nhi Nguyen

N11232862

6/4/2023

IFN666 Web APP – Capstone project



IFN666

Web APP – Capstone project

Link to Demo video: https://youtu.be/BbUMPXhnBjg

Contents

[Introduction 4](#_Toc137159941)

[Purpose & description 4](#_Toc137159942)

[Completeness and Limitations 6](#_Toc137159943)

[Use of APIs 7](#_Toc137159944)

[Coingecko 7](#_Toc137159945)

[Self-developed crypto-api 7](#_Toc137159946)

[Modules used. 7](#_Toc137159947)

[Client-side module 7](#_Toc137159948)

[Server-side modules 8](#_Toc137159949)

[Application Design 9](#_Toc137159950)

[Overall 9](#_Toc137159951)

[Redux 12](#_Toc137159952)

[React @Asyncstorage to manage user token. 15](#_Toc137159953)

[Manage user watching list. 17](#_Toc137159954)

[Architecture 23](#_Toc137159955)

[Haven’t Implemented (Future plan) 24](#_Toc137159956)

[Test plan 24](#_Toc137159957)

[Testing QUT server deployment 28](#_Toc137159958)

[Discussions 30](#_Toc137159959)

[User guide 31](#_Toc137159960)

[Appendices 33](#_Toc137159961)

[Difficulties 33](#_Toc137159962)

[API screenshot 66](#_Toc137159963)

Figures

[Figure 1: Home page 4](#_Toc137076330)

[Figure 2: Home page bottom tab bar. 6](#_Toc137076331)

[Figure 3: Displaying for charts, details, figures 6](#_Toc137076332)

[Figure 4: System design 10](#_Toc137076333)

[Figure 5: crypto API routes 11](#_Toc137076334)

[Figure 6: Redux get current user holdings 12](#_Toc137076335)

[Figure 7: Redux Reducer handling. 13](#_Toc137076336)

[Figure 8: market reducer handling. 14](#_Toc137076337)

[Figure 9: Mapping between Redux state and props. 15](#_Toc137076338)

[Figure 10: AsyncStorage for Token, wallet, email. 16](#_Toc137076339)

[Figure 11: Get email storage. 16](#_Toc137076340)

[Figure 12: Checking if token exist to do pages navigation. 16](#_Toc137076341)

[Figure 13: Clear token when user signout. 17](#_Toc137076342)

[Figure 14: get user watchlist (component calling). 18](#_Toc137076343)

[Figure 15: get user watchlist by request to API. 18](#_Toc137076344)

[Figure 16: crypto-api routes for handling get user watchinglist. 19](#_Toc137076345)

[Figure 17: Delete coin in watchlist component. 20](#_Toc137076346)

[Figure 18: Delete coin by requesting to API. 20](#_Toc137076347)

[Figure 19: Delete coin handling by crypto-api. 21](#_Toc137076348)

[Figure 20: Add coin to watching list (component). 22](#_Toc137076349)

[Figure 21: Add to watchlist requesting from api. 22](#_Toc137076350)

[Figure 22: Add coin handling in crypto-api. 23](#_Toc137076351)

[Figure 23: client-side source code package. 24](#_Toc137076352)

[Figure 24: crypto-api QUT server folder location. 29](#_Toc137076353)

[Figure 25: pm2 start for crypto-api. 29](#_Toc137076354)

[Figure 26: checking server IP. 30](#_Toc137076355)

[Figure 27: Verify deploying successful on QUT server by call API. 30](#_Toc137076356)

[Figure 28: Splash page. 35](#_Toc137076357)

[Figure 29: Home page with bottom tab bar. 36](#_Toc137076358)

[Figure 30: Chart detail with coin selection. 37](#_Toc137076359)

[Figure 31: Select Search page. 38](#_Toc137076360)

[Figure 32: Coin detail when select a coin. 39](#_Toc137076361)

[Figure 33: Search valid coinID. 40](#_Toc137076362)

[Figure 34: Add an exist coin to watchlist. 41](#_Toc137076363)

[Figure 35: Search invalid coinID. 42](#_Toc137076364)

[Figure 36: Add non-exist coinID to watchlist. 43](#_Toc137076365)

[Figure 37: Showing user watchlist page. 44](#_Toc137076366)

[Figure 38: Remove a coin from watchlist. 45](#_Toc137076367)

[Figure 39: Pop up confirms/cancel removing a coin from watchlist. 46](#_Toc137076368)

[Figure 40: Removed coin successful and does not show in watchlist. 47](#_Toc137076369)

[Figure 41: Select a coin to see its detail. 48](#_Toc137076370)

[Figure 42: Show coin detail. 49](#_Toc137076371)

[Figure 43: Market tab bar. 50](#_Toc137076372)

[Figure 44: Market Assets switch tabs. 51](#_Toc137076373)

[Figure 45: Market Exchanges switch tabs. 52](#_Toc137076374)

[Figure 46: Profile screen. 53](#_Toc137076375)

[Figure 47: Login screen. 54](#_Toc137076376)

[Figure 48: Enable to see Password. 55](#_Toc137076377)

[Figure 49: Login without password. 56](#_Toc137076378)

[Figure 50: Register new account screen. 57](#_Toc137076379)

[Figure 51: Warning with wrong email format. 58](#_Toc137076380)

[Figure 52: Alert when user input invalid Password. 59](#_Toc137076381)

[Figure 53: Alert when user input with invalid Password (at least 1 digit). 60](#_Toc137076382)

[Figure 54: Password and confirm password do not match. 61](#_Toc137076383)

[Figure 55: Pop up for register successful. 62](#_Toc137076384)

[Figure 56: Login by incorrect password. 63](#_Toc137076385)

[Figure 57: Login successful and app direct to homepage. 64](#_Toc137076386)

[Figure 58: Search valid coin. 65](#_Toc137076387)

[Figure 59: User watching list is empty. 66](#_Toc137076388)

[Figure 60: Error 400 for register account with empty email or password. 67](#_Toc137076389)

[Figure 61: Create user successfully in API. 67](#_Toc137076390)

[Figure 62: Login with non-exist user. 68](#_Toc137076391)

[Figure 63: Login with incorrect password. 69](#_Toc137076392)

[Figure 64: Login with correct password. 69](#_Toc137076393)

[Figure 65: Get user watch list. 70](#_Toc137076394)

[Figure 66: Get user with empty watch list. 71](#_Toc137076395)

[Figure 67: Add a coin ID to user watch list (200). 72](#_Toc137076396)

[Figure 68: Add existed coin ID to user watch list (return 409). 73](#_Toc137076397)

[Figure 69: Add coin ID to non-exist user. 74](#_Toc137076398)

[Figure 70: Delete a non-exist coin ID in user watch list. 75](#_Toc137076399)

[Figure 71: Delete exist coin ID in user watch list (return 200). 76](#_Toc137076400)

## Introduction

### Purpose & description

Welcome to our cryptocurrency app Harvest, where you can explore the real-time market, create personalized watchlists, and learn more about a variety of coins. Our app is intended to give you a user-friendly experience while also keeping you updated on the ever-changing cryptocurrency ecosystem. You can easily browse via the app and add coins to your watchlist using a user-friendly interface, letting you track their performance and make educated investment decisions. We want to introduce transfer and withdrawal functionality in the future to improve your wallet management experience. Our constant dedication to security is shown in upcoming features like multi-factor authentication and encryption, which will provide the highest level of protection for your funds against unauthorized access. With our Harvest application, you can stay ahead of the game and go on the road to financial freedom and success in the digital sphere.

Introducing our modern cryptocurrency app, designed to help you make quick and educated decisions. Users may get real-time market data and understandable graphics with a single click, allowing them to swiftly assess crypto trends. Our streamlined design offers easy navigation and allows consumers to keep ahead of the competition. Take command of your digital assets with our sophisticated app and make confident investment decisions in real-time.



Figure 1: Home page

With our search coin’s function, you can effortlessly explore a vast array of cryptocurrencies. Simply enter the id of a coin, and our app will provide you with comprehensive details, including price, market cap, and key statistics, helping you make informed investment choices. (Figure 33)

The top coins feature in our app displays the most popular cryptocurrencies based on market capitalization, helping you remain up to date with the industry leaders. You can examine the top-performing coins briefly, allowing you to observe trends and make strategic investment decisions. (Figure 1)

The valuable coin management capabilities of our software allow you to easily add and remove currencies from your personalized watching list. This allows you to keep a close eye on the performance of your chosen currencies and remain up to speed on price fluctuations and market trends. You can quickly access and see your watch list using our user-friendly interface, ensuring you never miss a beat in the volatile world of cryptocurrency.

Furthermore, we have integrated cutting-edge technology into our app to provide an innovative and optimized user experience. We assure effective state management using Redux, allowing for smooth data flow and improved speed. We create aesthetically appealing and interactive charts that allow customers to easily analyze coin movements with Victory Native Chart. Bottom Tabs' interaction with React navigating allows seamless navigating across app screens, increasing user ease. In addition, we have used animated charts to provide our customers with a dynamic and engaging visual experience, making the app both practical and aesthetically beautiful.

A blue screen with a heart

Description automatically generated with medium confidence

Figure 2: Home page bottom tab bar.

A screenshot of a graph

Description automatically generated with low confidence

A green line on a black background

Description automatically generated with medium confidence

A picture containing screenshot, darkness, graphics

Description automatically generated

Figure 3: Displaying for charts, details, figures.

### Completeness and Limitations

*What works?*

All main features that need for Harvest app including:

\_ Display top of cryptocurrency.

\_ Display wallet balance info.

\_ Display coin detail by chart.

\_ Display coin trend by chart.

\_ Display coin detail information.

\_ Search cryptocurrency by their id.

\_ Add favorite coin to watching list.

\_ Remove coin in watching list.

\_ Display user watching list of coins.

\_ Login for authorization users.

\_ Logout for authorization users.

\_ Register new account.

\_ Get list of Cryptocurrency exchanges (a business that allows customers to trade cryptocurrencies or digital currencies for other assets).

\_ Get list of Crypto assets (digital assets which use cryptographic techniques to generate a medium of exchange of financial transactions).

*What can be implemented in the future?*

We have limiting API request for free API, and we have limiting in timing to finish this project, so we are planning to implement some features in the future.

\_ User can buy coins, transfer, withdraw on this wallet app.

\_ User can manage their balance.

\_ User can see their holding coins trends and details.

\_ User can have multi-factor authentication and encryption.

\_ App can verify user phone numbers, email, personal information.

In this version of capstone project, we focus on delivering real-time cryptocurrency charts, detail, information, user watching list. So, we haven’t implemented crypto wallet features, so the user balance in home page is random number.

# Use of APIs

## Coingecko

<https://www.coingecko.com/en/api>

CoinGecko API is a comprehensive cryptocurrency data API that provides access to real-time and historical market data, coin information, and more.

## Self-developed crypto-api

This API is deployed on QUT server.

This API handles login, register, get/update/delete user’s coin watchlist.

# Modules used.

## Client-side module

*React Navigation*

React Navigation is a popular navigation library for React Native, providing a variety of navigation patterns and features.

<https://www.npmjs.com/package/react-navigation>

*Redux*

Redux is a widely used state management library for JavaScript applications, including React Native, offering a predictable and centralized approach to managing application state.

<https://www.npmjs.com/package/redux>

*Victory Native*

Victory Native is a powerful charting library specifically designed for React Native, allowing developers to create stunning and interactive charts.

<https://www.npmjs.com/package/victory-native>

*React Native Chart Kit*

React Native Chart Kit is a comprehensive charting library that provides a wide range of customizable and feature-rich charts, enabling developers to create visually appealing data visualizations in React Native apps.

<https://www.npmjs.com/package/react-native-chart-kit>

*Axios*

axios is a popular promise-based HTTP client for JavaScript that allows you to make HTTP requests from both browser and Node.js environments with ease.

<https://www.npmjs.com/package/axios>

## Server-side modules

*bcrypt:*

bcrypt is a widely used library for password hashing and encryption in JavaScript applications, providing robust security features.

<https://www.npmjs.com/package/bcrypt>

*cors:*

cors is a middleware package for Express that enables Cross-Origin Resource Sharing (CORS) and allows controlled access to resources from different origins. npm link

express: express is a fast and minimalist web application framework for Node.js that simplifies the process of building robust and scalable web applications.

<https://www.npmjs.com/package/cors>

*express:*

express is a Node.js web application framework that is quick and simple, making it easier to construct strong and scalable online applications.

<https://www.npmjs.com/package/express>

*jsonwebtoken:*

JSON Web Tokens (JWTs), which are often used for authentication and secure communication between parties, may be generated and verified using jsonwebtoken.

<https://www.npmjs.com/package/jsonwebtoken>

*knex:*

knex is a Node.js query builder that provides a clean and straightforward interface to relational databases, simplifying database queries and migrations.

<https://www.npmjs.com/package/knex>

*nodemon:*

nodemon is a utility that watches changes in your Node.js application and restarts the server immediately anytime a file is modified, making the development process more efficient.

<https://www.npmjs.com/package/nodemon>

# Application Design

### Overall

**Link between crypto client app, Node server API, MySQL database**

Our Harvest app is supported by a powerful Node.js API, which acts as the foundation for handling user data. To securely store user information, the API makes use of a MySQL database. It adheres to the RESTful design, which enables smooth communication between the client (React Native app) and the server.

The API provides several endpoints for handling user-related actions. POST endpoints, for example, manage user registration and login, creating authentication tokens for safe access. The GET endpoints allow for the retrieval of user information as well as access to their watchlist, which provides a detailed view of their preferred currencies. Users can add or delete coins from their watchlist by using the POST endpoint. Furthermore, the DELETE endpoint allows for the removal of certain coins from the watchlist.

Our API server is hosted on a QUT Unix server, which offers stability, dependability, and effective security measures. Our crypto client app interacts with the API to obtain and show user-specific information on the client side. With a user-friendly clear design, users can access their accounts, manage their watchlists, and remain up to speed on their coin holdings.

MySQL DB

**Connections**

Node server

Express

**Request to hosted API**

**Response from server**

Crypto app client

Figure 4: Client server design

A picture containing text, screenshot, font

Description automatically generated

In the MySQL database, within the "crypto" schema, we have a table named "user" that serves to handle user-related information such as email addresses, wallet balances, password hashes, and their associated watching list for cryptocurrencies.



Figure 5: crypto API routes

Crypto API user management:

\_ /users/register (POST): This API enables the creation of new user accounts, allowing users to register and store their account information securely within the React application, utilizing appropriate encryption techniques and data handling practices.

\_ /users/login (POST): This API facilitates user login functionality by generating and returning a token that is securely stored in a React application, utilizing libraries such as jsonwebtoken and bcrypt for authentication and data protection.

- /users/watchlist/:email/:coinId (PUT): This API facilitates users to add their favorite cryptocurrency and be able retrieve it later in watching list.

- /users/watchlist/:email/:coinId (DELETE): This API provides functionality for users to remove unwanted coin IDs from their watching list, allowing them to customize and manage the list of coins they are monitoring within the application.

- /users/watchlist/:email (GET): This API facilitates users to get their watching list.

## Redux

Follow these steps to extract concurrency from the CoinGecko API in Redux using mapDispatchToProps and mapStateToProps:

\_ Define action creators in the mapDispatchToProps method that sends activities for collecting concurrency data from the API. These action creators encapsulate the API call logic and execute the relevant actions based on the data received.

\_ Use the mapStateToProps method to map the required bits of the Redux state to the component's props. This contains the API-fetched concurrency data and any other relevant state attributes.

\_ Connect the component to Redux using the react-redux library's connect() method. To make the necessary connections, use the mapDispatchToProps and mapStateToProps methods as parameters.

\_ With the component linked to Redux, we can now use props to obtain the concurrency data from the CoinGecko API. This data may be used by the component to display and alter the information as needed.

We can connect the CoinGecko API into your application, collect the concurrency data, and manage it within your Redux store by using mapDispatchToProps and mapStateToProps in combination with Redux.

*A screen shot of a computer code

Description automatically generated with low confidence*

Figure 6: Redux get current user holdings

*A picture containing text, screenshot, font

Description automatically generated*

Figure 7: Redux Reducer handling.

*A screen shot of a computer program

Description automatically generated with low confidence*

Figure 8: market reducer handling.

*A screen shot of a computer program

Description automatically generated with low confidence*

Figure 9: Mapping between Redux state and props.

## React @Asyncstorage to manage user token.

We implement AsyncStorage for storing the token returned from the API and the user's email to enhance the user experience in the app. Here's how it works:

\_ When the API delivers a token after a successful login, we use AsyncStorage to securely store the token locally on the device. This allows the user to stay logged in even after closing the app, avoiding the need to log in repeatedly.

\_ We obtain the token from AsyncStorage when the user reopens the app. If a token exists, the system uses it to immediately authenticate the user, skipping the login procedure and giving them quick access to the app's features and data.

\_ We also save the user's email address in AsyncStorage. This is useful for retrieving the user's watchlist based on their email address, ensuring personalized and relevant info is easily accessible.

We have a smooth login experience for users by utilizing AsyncStorage, reducing the burden of continually checking in and keeping a persistent session. The cached token enables automatic login, while the user's email enables retrieval of their unique watchlist, allowing the app's content to be tailored to their tastes.

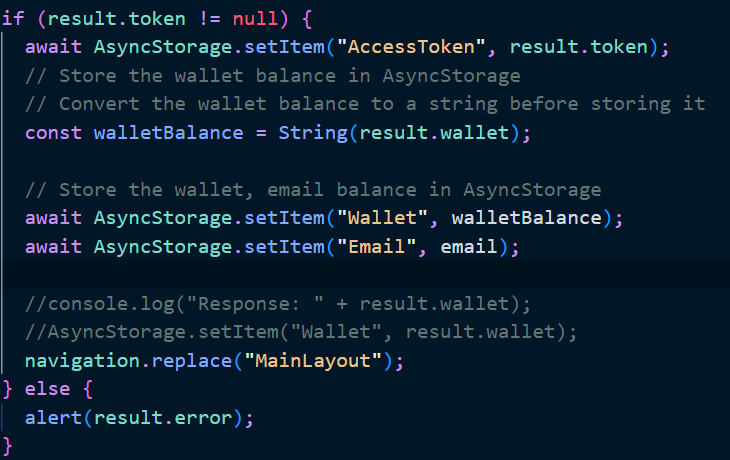


Figure 10: AsyncStorage for Token, wallet, email.

A picture containing text, font, screenshot, line

Description automatically generated

Figure 11: Get email storage.

A picture containing text, screenshot, font

Description automatically generated

Figure 12: Checking if token exist to do pages navigation.

A picture containing text, screenshot, font

Description automatically generated

Figure 13: Clear token when user signout.

## Manage user watching list.

Once a user logs in, their watchlist is just a few steps away. With the user's email securely stored in AsyncStorage, the system retrieves their watchlist by storage email. And then sending a request to the server's API, the app retrieves the watchlist data from the MySQL database. The watchlist, stored as a JSON list in MySQL, allows for seamless updates, retrievals, and deletions. Utilizing the power of the FlatList component, the app efficiently displays the coins in the watchlist, optimizing performance by rendering only the visible items.

To remove a coin from the watchlist, a dedicated function handles the deletion by sending a DELETE action to the API, triggering the removal of the specified coin from the database. The app then updates the displayed list, ensuring the user sees an accurate representation of their updated watchlist.

A screen shot of a computer program

Description automatically generated with low confidence

Figure 14: get user watchlist (component calling).

A picture containing text, screenshot, font

Description automatically generated

Figure 15: get user watchlist by request to API.

A screen shot of a computer program

Description automatically generated with low confidence

Figure 16: crypto-api routes for handling get user watchinglist.

A picture containing text, screenshot, font, software

Description automatically generated

Figure 17: Delete coin in watchlist component.

A screen shot of a computer code

Description automatically generated with low confidence

Figure 18: Delete coin by requesting to API.

A screen shot of a computer program

Description automatically generated with low confidence

Figure 19: Delete coin handling by crypto-api.

Users can add coins to their watchlist. When a user selects a coin to add, the system captures the coin ID. This coin ID is then utilized in executing a POST function that updates the user's watchlist.

Upon the user's request to add a coin, the app triggers the necessary actions to communicate with the server's API. The API handles the request to update the user's watchlist by adding the selected coin using the provided coin.

*A screen shot of a computer code

Description automatically generated with low confidence*

Figure 20: Add coin to watching list (component).

*A picture containing text, screenshot, font

Description automatically generated*

Figure 21: Add to watchlist requesting from api.

*A picture containing text, screenshot, software, operating system

Description automatically generated*

Figure 22: Add coin handling in crypto-api.

# Architecture

The overall architecture of this crypto client application:

* api: package for handling request, response from fetching crypto-api Node server.
* assets: application assets (icons, images…).
* components: these common reusable components are used across the application.
* constants: this folder is used to store files that define constants or configuration values used throughout the application.
* navigation: handling navigation for home tab bar.
* screens: organize and store the source code files for individual screens of app.
* stores: this folder is used to store the files related to managing the global state of your application (Redux).
* styles: the sharing style across application.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 23: client-side source code package.

Organizing the source code of this application into different folders provides various advantages and is critical for effective development. For beginnings, it encourages modularity and reusability by enabling you to design self-contained components or functions that may be reused across the application, saving time and effort (components). Second, a well-structured folder hierarchy improves code organization and readability, making it easier for developers to find files and comprehend their purpose and scope. As our system expands, this adds to greater code maintainability and scalability. You obtain a clear separation of concerns by splitting distinct issues into dedicated folders, such as screens, components, styles, or API management, making it easier to manage and adjust different elements of our application independently. Overall, separating folders in your application allows for a more methodical approach to development, which improves code quality, productivity, and the overall development experience.

## Haven’t Implemented (Future plan)

During the current implementation phase, our primary focus is on providing users with exceptional experience in terms of market chart visualization, analytics, and managing their personalized watchlist in the cryptocurrency market. Due to time constraints, features such as transfer, withdrawal, and buying cryptocurrencies have not been included in this version of the app.

It's important to note that we do not store any user buying, selling, or cryptocurrency ownership information within the app. Our app is designed to solely serve as a tool for users to explore the market, gain insights, and track their preferred cryptocurrencies.

We are committed to continuously improving and expanding our app's functionalities. In our future, we aim to implement features that will enable users to complete transactions, manage their cryptocurrency holdings, and facilitate buying and selling activities. These upcoming features will transform our app into a comprehensive cryptocurrency wallet, offering a complete solution for users to store, trade, and manage their digital assets securely. We strive to provide complete cryptocurrency wallet experience, empowering you to take full control of your digital finances in the next implementation phase.

### Test plan

We do unit tests when we do the development process. We also have manual testing to make sure the web pages work as expected. We have the table below to show our testing, expected outcome, result and screen shots. Please note that we have some limitations on CoinGecko API basic plan with (10-30 req/min).

Our tests include:

* Positive outcome cases
* Negative outcome cases (error scenarios)
* Edge cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Expected Outcome** | **Result** | **Screenshot(s)** |
| **Checking screen between tabs** | | | |
| Select the home screen. | Home screen displays user balance in database, dummy holding coins. Tab bar highlights Home icon. | PASS | Figure 29 |
| Select coin on home screen. | Top chart information displays coin detail. | PASS | Figure 30, Figure 29 |
| Select the search screen on tab bar. | Tab bar highlights Search icon. Page header changes to Search. | PASS | Figure 31 |
| Search coin by correct ID. | Searching detail section displays searching coin. | PASS | Figure 33 |
| Search coin by incorrect ID. | Searching detail section displays can’t find searching coin. | PASS | Figure 35 |
| Select by click at coin to see its detail. | Screen Coin detail displays with their detail information. Display if coins value increase (green text colour, up arrow icon), decrease (red text colour, down arrow icon). | PASS | Figure 32 |
| Select watch list screen. | Page title is “Favorited Coins” with list of users watching coins. | PASS | Figure 37 |
| Select user with empty watchlist | Page title is “Sorry, your watching list is empty!” | PASS | Figure 56 |
| Load user watching list. | App loads correct user watching list in database. | PASS | Figure 37 |
| Add a non-exist coin to watching list to database. | Add coin id to database and modal displays “Added successfully”. | PASS | Figure 36, Figure 37 |
| Add a exist coin to watching list to database. | Coin id can’t add to database and modal displays “Coin already in watching list”. | PASS | Figure 34 |
| Remove coin from watching list. | Modal pops up with text “Are you sure you want to remove this coin from your watch list”. | PASS | Figure 38, Figure 39, Figure 40 |
| Remove coin from watching list then select Confirm from modal. | Coin is removed from watching list in database and on screen. | PASS | Figure 40 |
| Remove coin from watching list then select Cancel from modal. | App does nothing, not removing coin. | PASS | Figure 37, Figure 39 |
| Select Market tab. | Screen header display “Market” header. Market icon highlight with Crypto assets tab bar highlights and list of coin name, icon, market chart, price with (increase, decrease colour). | PASS | Figure 43 |
| Select Assets tab bar. | Assets tab bar highlights and list of coin name, icon, market chart, price with (increase, decrease colour). | PASS | Figure 44 |
| Select Exchanges tab bar. | Exchange tab bar highlights and list of coin name, icon, market chart, price with (increase, decrease colour). | PASS | Figure 45 |
| Select Profile tab bar. | Profile screen with “Profile header” and user email. And Profile screen show some action for user (SignOut) | PASS | Figure 46 |
| **Login, register account, logout, remember user token** | | | |
| Display Login. | Login screen displays input for email, password and disable Login button. | PASS | Figure 47 |
| Input incorrect email format (Login, Register screen) | Login/Register button still disable with incorrect email format. (Correct format: with @, have .com, etc) | PASS | Figure 49 |
| Not input password or incorrect password or email in Login. | User can’t login into system. And have pop up to ask user input correct email and password. | PASS | Figure 49, Figure 53, Figure 52 |
| Click “eye” icon to show password. | Password input field shows input password and disable after click again. | PASS | Figure 48 |
| Display Register account screen. | Display email input, password input, confirm password input and some description to explain requirement for registered password. | PASS | Figure 48 |
| Not input the same password and confirm password in register account page. | App displays pop up to ask user type those input fields the same information. | PASS | Figure 54 |
| Register account successful. | App direct to Login screen to help user login to system. | PASS | Figure 55 |
| Remember user token after 1st login success. | App automatically direct to home screen without login action require. | PASS | Figure 29 |
| Select Sign-out in Profile screen | Direct to login page. User can’t see any tab bars, and their screens. | PASS | Figure 46, Figure 47 |
| Remove token after selecting Sign-out. | After Sign-out, app remove user token. User needs to login again to see their home, watching list, valid screens. | PASS | Figure 47 |
| **Testing for crypto API** | | | |
| Check empty body for Register user API | Response is 400 with message “Request body incomplete - email and password needed” | PASS | Figure 60 |
| Register correct account for Register user API | Response is “201 User created” | PASS | Figure 61 |
| Login with correct user | Response is “200” with user token | PASS | Figure 63 |
| Login user with incorrect password | Response is "error": true, "message": "Passwords do not match" | PASS | Figure 62 |
| Login with non-exist account | Response is  "error": true,  "message": "User does not exist" |  | Figure 62 |
| Get non-empty watchlist | Response 200 with array of watchlist successfully | PASS | Figure 65 |
| Get empty watchlist | Response 200 with empty watchlist array | PASS | Figure 66 |
| Delete an exist coin ID in watchlist | Response 200 OK with "status": "200",  "message": "Coin removed from watchlist" | PASS | Figure 71 |
| Delete non-exist coin ID in watchlist | Response with message “Coin ID is not exist” | PASS | Figure 70 |

## Testing QUT server deployment

1. Execute MySQL
2. Deploy user API

Step 1: Download source code zip file, extract files

A screenshot of a computer

Description automatically generated

Figure 24: crypto-api QUT server folder location.

Step 2: pm2 install, start and check list

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 25: pm2 start for crypto-api.

Step 3: check QUT server IP

A picture containing text, screenshot, font

Description automatically generated

Figure 26: checking server IP.

Step 4: verify connect with user API by calling Login function (Expected return user token => PASS)

A screenshot of a computer

Description automatically generated

Figure 27: Verify deploying successful on QUT server by call API.

## Discussions

If we wanted to deploy this application on Apple store or Google Play by using Expo React Native here are some pros and cons, we consider:

Pros of using Expo:

\_ Expo allows this application to run cross-platform both IOS and Android platforms without requirement separate codebases, it will save time and effort.

\_ Expo provides binaries and certificates.

\_ Expo allows us to push updating code without requiring users to download the new versions from app stores, this can help us quickly debug or maintenance.

Cons of using Expo:

\_ Depending on Expo infrastructure and services will limit availability of our applications if Expo needs to have downtime to update or maintenance.

\_ We should consider app size and performance add on by Expo runtime layers. By considering how many users are in target and performance of the app by real time displaying currency information.

In this app we have only test in mobile devices with stable of outcomes. We suggest that we should have more testing on multiple of devices such as Tablet, mobile devices with different version of operating system (this testing on iPhone 11). In addition, we have the same codebase for android and IOS devices, in the long-term development we consider specific features based on specific operating systems (Android, IOS). From that, we can have separate folders to deal with specific operating systems features.

For accessibility for persons with diverse disabilities, it is critical to evaluate the present application's accessibility and determine how it might be enhanced to accommodate disabled users. This can include features like screen reader support, keyboard navigation, color contrast for visually challenged users, and compliance with accessibility standards like WCAG 2.1. Conducting accessibility audits, including disabled people in testing, and incorporating their comments can help detect and fix accessibility problems.

Before launching the application, it is critical to examine any potential social, ethical, or legal consequences. Data privacy and security, ensuring compliance with relevant regulations such as GDPR or CCPA, user consent for data collection and storage, transparent communication about how user data is used, and adherence to ethical guidelines for handling sensitive financial information are just a few things to think about.

It is also vital to analyze the app's societal impact. Cryptocurrency markets may be extremely unpredictable and speculative, therefore it's critical to give consumers clear and accurate information, allowing them to make educated decisions rather than encouraging dangerous behavior. Users can be helped to avoid injury by educating them about the hazards involved and offering disclaimers.

Working with legal specialists and subject matter experts can help us navigate the legal and ethical issues that are unique to the cryptocurrencies business and the locations where the app will be deployed.

We can guarantee that the application is inclusive, ethical, and compliant by addressing accessibility, social effect, and legal concerns. This will provide a great user experience while limiting possible dangers and legal difficulties.

## User guide

1. Using menu
2. General Menu tab bar:

By selecting tabs on the bottom tab screens that can navigate you to page that suit with your selection of page (Figure 1, Figure 2, Figure 46, Figure 43, Figure 31)

1. Login Page:

From the first time open this app, the system will require you to login with your registered email and password (both are required fields). Users should follow the registered email format (Figure 47)

Application will enable Login button if user input correct email, password format. And click the Login button to access the app. (Figure 47, Figure 48)

After the 1st successful login, users don’t need to Login again, the system will save user token to automatically sign in (Figure 29)

1. Register new account:

Click Register new account (Figure 47) => Page direct to register session that allows you enter legal email address and password (password must contains at least one uppercase, lowercase, special character, digit and 8-16 Characters Long), confirm password to sign up your account.

1. Show home page:

By default, login will be on home screen to show user balance and buying cryptocurrencies (Figure 29)

User can choose holding coin to see their price on above chart (Figure 30).

1. Search coin:

Click at the 2nd tab at bottom tab to search for coin (Figure 29)

In the input field, users fill up the field with coin ID and click Search (Figure 33)

\_ If the coin ID exists, it will show the coin information => click to showing coin to see coin details.

\_ If the coin ID non-exist, it will show the “Sorry, we can’t find any results with your search”. (Figure 35)

1. Show user watching list:

Click at the “heart icon” to move to user’s watching list (Figure 29, Figure 37).

1. Remove coin from watching list:

By holding long press at “-“icon to remove coin from watching list, the pop up will show to ask “Are you sure to remove this coin from your watch list”. Click confirm to remove and click Cancel to cancel this action (Figure 38, Figure 39).

1. Add coin to watching list:

In Coin detail page (Figure 32), click add Watch list to add that cryptocurrency to your watch list. If the coin is already in your watch list, the system will show “Coin already in watch list” (Figure 34).

If the coin is non-existent in watch list, the system will add it to your list and show confirmation (Figure 36).

1. See cryptocurrency market:

By clicking at market icon (Figure) to direct page to see the cryptocurrency market. Switch between “Assets” and “Exchanges” to see details:

User can click to the crypto to see their coin detail (Figure 43, Figure 44, Figure 45)

1. See profile:

By choosing profile icon at the bottom tab bar to see your profile (Figure 46)

1. Logout to the system

Click at Sign out selection on Profile to log out from the system (Figure 46)

## Appendices

#### Appendix A – self-checking against CRA

|  |  |  |
| --- | --- | --- |
| Marks | Grade level (in 1- 7 scale) my work belongs to (delete the ones not suitable) | Marks I think I should get |
| Front end |  |  |
| Front-end Mobile Application Functionality   1. arks) | 7 | 19 out of 20 |
| Front-end Application Robustness  (10 marks) | 7 | 10 out of 10 |
| Front-end Application UI Design  (10 marks) | 7 | 10 out of 10 |
| Back end |  |  |
| Backend functionality, error responses and application reliability.  (20 marks) | 7 | 20 out of 20 |
| Back-end Application Architecture and Middleware – DB Connectivity, Logging and Security.  (10 marks) | 7 | 10 out of 10 |
|  |  |  |
| Development Process & Code Quality  (10 marks) | 7 | 9 out of 10 |
|  |  |  |
| Report and User Guide  (10 marks) | 7 | 10 out of 10 |
|  |  |  |
| Video demo  (10 marks) | 7 | 10 out of 10 |
|  |  |  |
| Overall Marks (100) | 7 | **98 out of 100** |

## Difficulties

\_ We have limitations on CoinGecko API basic plan with (10-30 req/min). To work within the limitations of the CoinGecko API basic plan, be cautious when testing features associated with this API. Design our code to minimize the number of API calls and be aware that receiving an empty response with error code 429 indicates that we have exceeded the rate limit.

#### [Appendix B – Screenshots of application](#_Toc22385)

A blue background with white text

Description automatically generated with medium confidence

Figure 28: Splash page.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Figure 29: Home page with bottom tab bar.

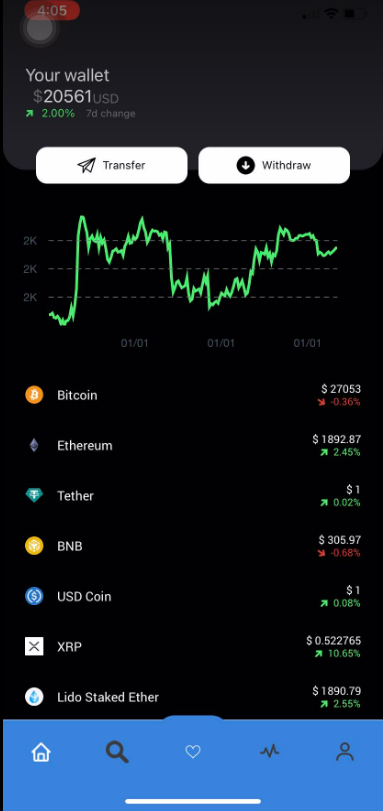


Figure 30: Chart detail with coin selection.

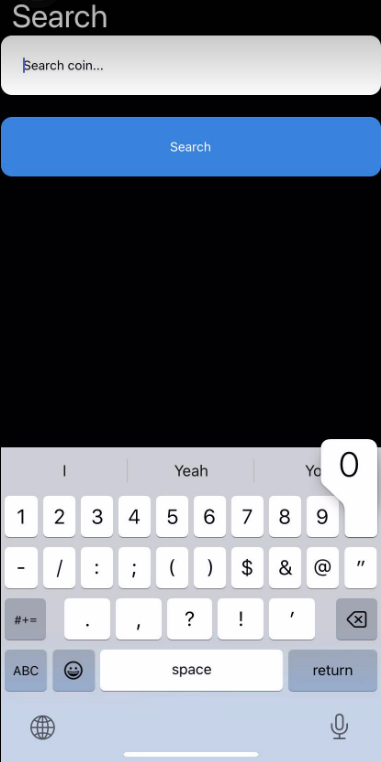


Figure 31: Select Search page.

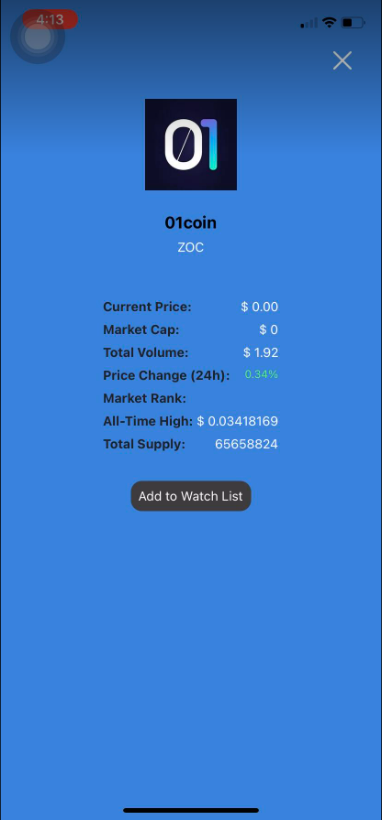


Figure 32: Coin detail when select a coin.

A screenshot of a cell phone

Description automatically generated with medium confidence

Figure 33: Search valid coinID.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 34: Add an exist coin to watchlist.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 35: Search invalid coinID.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 36: Add non-exist coinID to watchlist.

A screenshot of a phone

Description automatically generated with low confidence

Figure 37: Showing user watchlist page.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 38: Remove a coin from watchlist.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 39: Pop up confirms/cancel removing a coin from watchlist.

A picture containing text, screenshot, design

Description automatically generated

Figure 40: Removed coin successful and does not show in watchlist.

A picture containing text, screenshot, software, operating system

Description automatically generated

Figure 41: Select a coin to see its detail.

A screenshot of a blue screen

Description automatically generated with low confidence

Figure 42: Show coin detail.



Figure 43: Market tab bar.

A screenshot of a phone

Description automatically generated with low confidence

Figure 44: Market Assets switch tabs.

A picture containing screenshot, text, software, multimedia software

Description automatically generated

Figure 45: Market Exchanges switch tabs.

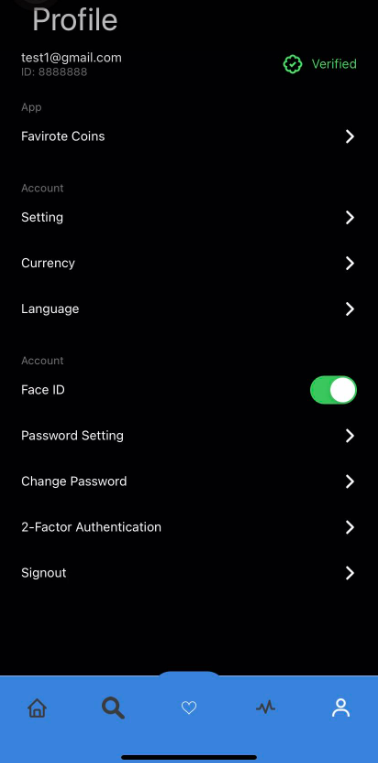


Figure 46: Profile screen.

A screenshot of a login screen

Description automatically generated with medium confidence

Figure 47: Login screen.

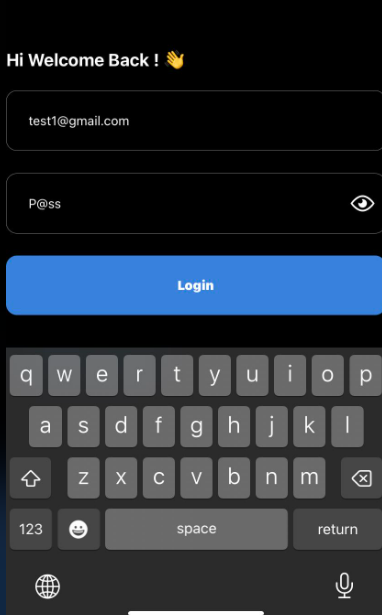


Figure 48: Enable to see Password.

A screenshot of a login screen

Description automatically generated with medium confidence

Figure 49: Login without password.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 50: Register new account screen.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 51: Warning with wrong email format.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 52: Alert when user input invalid Password.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 53: Alert when user input with invalid Password (at least 1 digit).

A screenshot of a phone

Description automatically generated with medium confidence

Figure 54: Password and confirm password do not match.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 55: Pop up for register successful.

A screenshot of a login screen

Description automatically generated with low confidence

Figure 56: Login by incorrect password.

A screenshot of a phone

Description automatically generated with low confidence

Figure 57: Login successful and app direct to homepage.

A screenshot of a phone

Description automatically generated with medium confidence

Figure 58: Search valid coin.

A picture containing text, screenshot, design

Description automatically generated

Figure 59: User watching list is empty.

## API screenshot

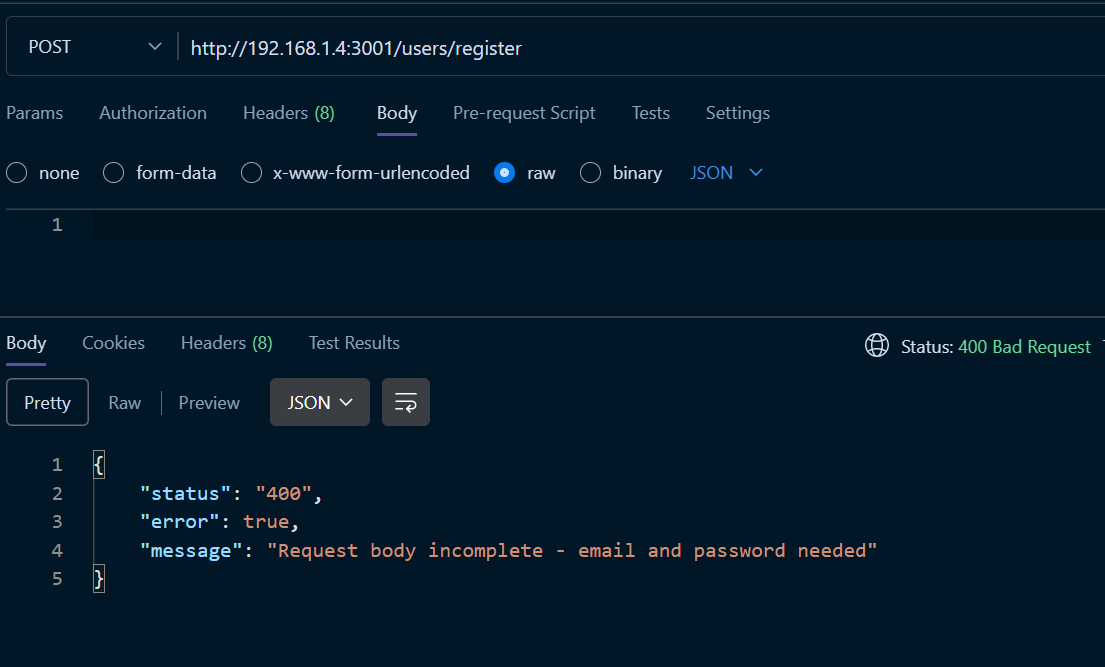


Figure 60: Error 400 for register account with empty email or password.

A screenshot of a computer

Description automatically generated



Figure 61: Create user successfully in API.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 62: Login with non-exist user.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 63: Login with incorrect password.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 64: Login with correct password.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 65: Get user watch list.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 66: Get user with empty watch list.

A screenshot of a computer

Description automatically generated

Figure 67: Add a coin ID to user watch list (200).

A screenshot of a computer

Description automatically generated with medium confidence

Figure 68: Add existed coin ID to user watch list (return 409).

A screenshot of a computer

Description automatically generated with medium confidence

Figure 69: Add coin ID to non-exist user.

A screenshot of a computer

Description automatically generated with medium confidence

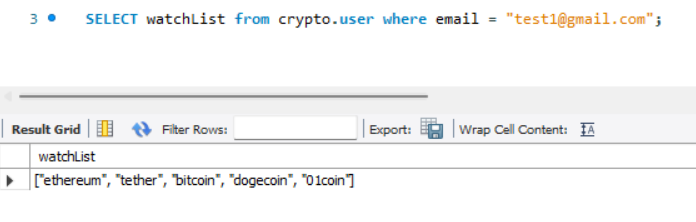


Figure 70: Delete a non-exist coin ID in user watch list.

A screenshot of a computer program

Description automatically generated with low confidence

Figure 71: Delete exist coin ID in user watch list (return 200).