

RocketFin Stock Trader

Full Stack Candidate Programming Exercise



October 27, 2024

Jeremy Fenech

RocketFin

Table of Contents

[Using the App 2](#_Toc180958242)

[Getting Setup 2](#_Toc180958243)

[First Launch 2](#_Toc180958244)

[Purchasing/ Selling a Stock 3](#_Toc180958245)

[Homepage Features 5](#_Toc180958246)

[Portfolio Page 7](#_Toc180958247)

[Error Messages 8](#_Toc180958248)

[Responsive Design 9](#_Toc180958249)

[Swagger API Documentation 10](#_Toc180958250)

[Unit Tests 11](#_Toc180958251)

# Using the App

## Getting Setup

Refer to the README.md for a detailed guide on the setup and config of the application. This document aims to show the application in use and highlight all the features included.

In the case of any issues with running the frontend/ backend services do not hesitate to reach out to me on [jeremyfenech03@gmail.com](mailto:jeremyfenech03@gmail.com).

## First Launch

Upon first launching the application the user will not have any entries within the database. This is reflected onto the various application screens with clear indications that there is nothing to show the user and that they should start by making a stock purchase from the transactions page.

A screenshot of a computer

Description automatically generated

Figure 1 – Homepage message indicating that the user has no stocks in portfolio

## Purchasing/ Selling a Stock

Navigating to the transactions page, either through the on-screen link or the navigation bar will allow the user to search for any stock and make their initial purchase.

A screenshot of a computer

Description automatically generated

Figure 2 – The transactions page

Following the inputting of a correct ticker symbol and clicking on the search button several details about that ticker are shown. The user is given the options to purchase or sell the stock at the current market price. Given that the user currently has no stocks attempting to sell will result in am error message. Similarly, searching of an incorrect ticker will also display a relevant error message.

A screenshot of a computer

Description automatically generated

Figure 3 – Details and options provided for a correct ticker

A screenshot of a computer

Description automatically generated

Figure 4 – Error when selling unavailable shares

Buying any number of shares will show a success message and then redirect the user to the homepage.

A screenshot of a computer

Description automatically generated

Figure 5 – Successful purchase message

If the user has sufficient balance to sell shares they can do this using the sell button. A message is also shown upon a successful sale and the user is redirected to the homepage.

A screenshot of a computer

Description automatically generated

Figure 6 – Successful sale message

## Homepage Features

After using the buy and sell features the homepage will update to reflect these transactions. The top of homepage shows a summary of the portfolio now with the number of stocks currently owned, their cost, their market value, return rate and return amount.

A screenshot of a computer

Description automatically generated

Figure 7 – Homepage after several transactions

The bottom half of the homepage shows the top 5 most recent transactions with information for the ticker name, the number of shares and the date and time of the transaction.

Clicking on any transaction will expand the details to show a few more information such as the price-per-share, cost basis and exact timestamp including milliseconds.

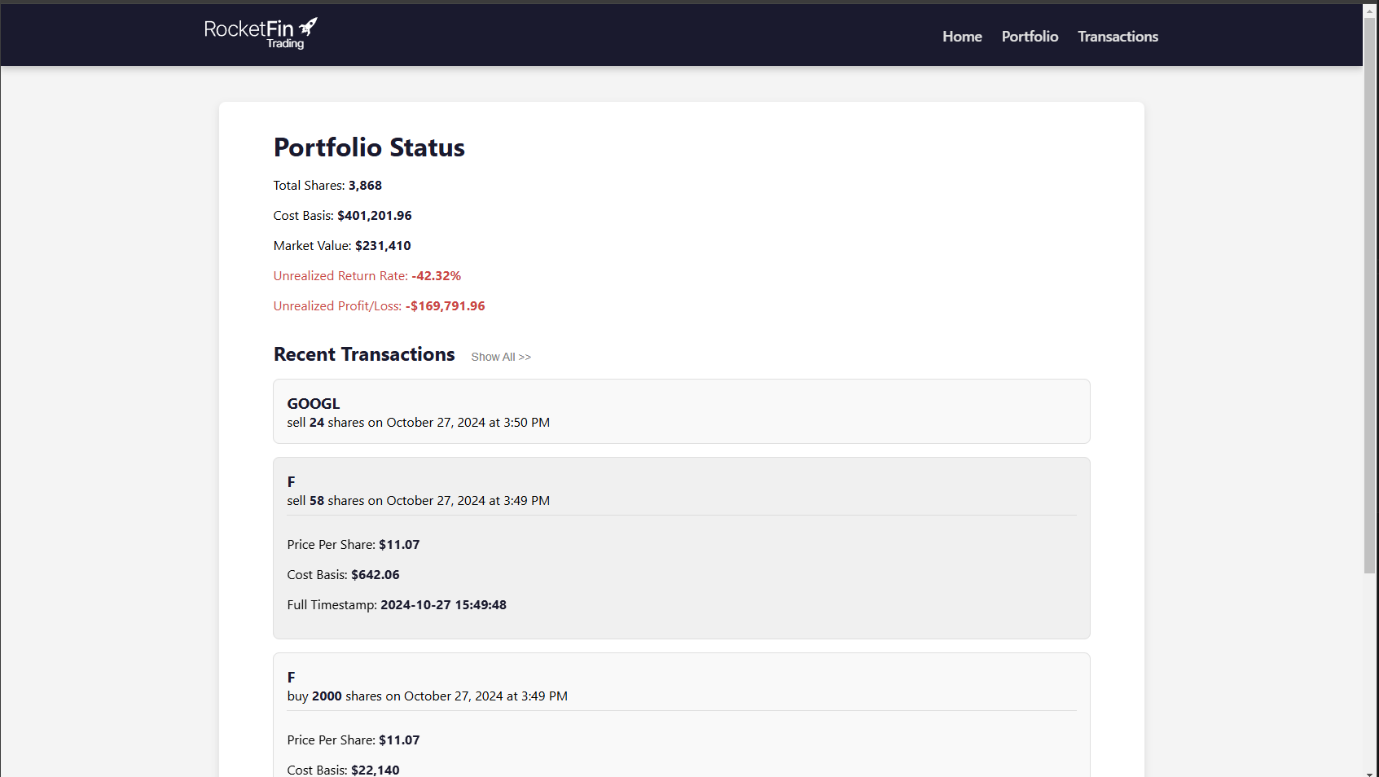


Figure 8 – Recent transactions with expanded details

A show all button is available next to the recent transactions title which opens a new page showing the entire history of all the transactions done. Details can also be expanded on this screen.

A screenshot of a computer

Description automatically generated

Figure 9 – All transactions screen

## Portfolio Page

The portfolio page is accessible through the navigation bar. This page shows a list of all currently owned stocks and the amounts of each ticker. Clicking on one of the stocks will expand to show further details such as the cost basis of all the stock, the current market value, the current return rate and the current return amount.

A screenshot of a website

Description automatically generated

Figure 10 – Portfolio page with some details expanded

# Error Messages

As shown within the previous section (Figure 4) error messages are displayed at the top of the page using a custom react component. Below are some other examples of error messages one may encounter in various scenarios.



Figure 11 – Error message searching for invalid instrument



Figure 12 – Error message when purchasing or selling invalid number of shares (zero or less)



Figure 13 – Error message on homepage when backend is unreachable or returns incorrectly



Figure 14 – Error message on portfolio when backend is unreachable or returns incorrectly



Figure 15 – General error message when backend is unreachable

All these error messages can be easily dismissed using the X button at the right side.

# Responsive Design

While designing the frontend of the application and writing the CSS it was ensured that the application was designed with responsiveness, simplicity and a modern look in mind. The responsiveness was tested for most device sizes through resizing the desktop application as well as using different mobile devices to ensure that the design translates correctly.

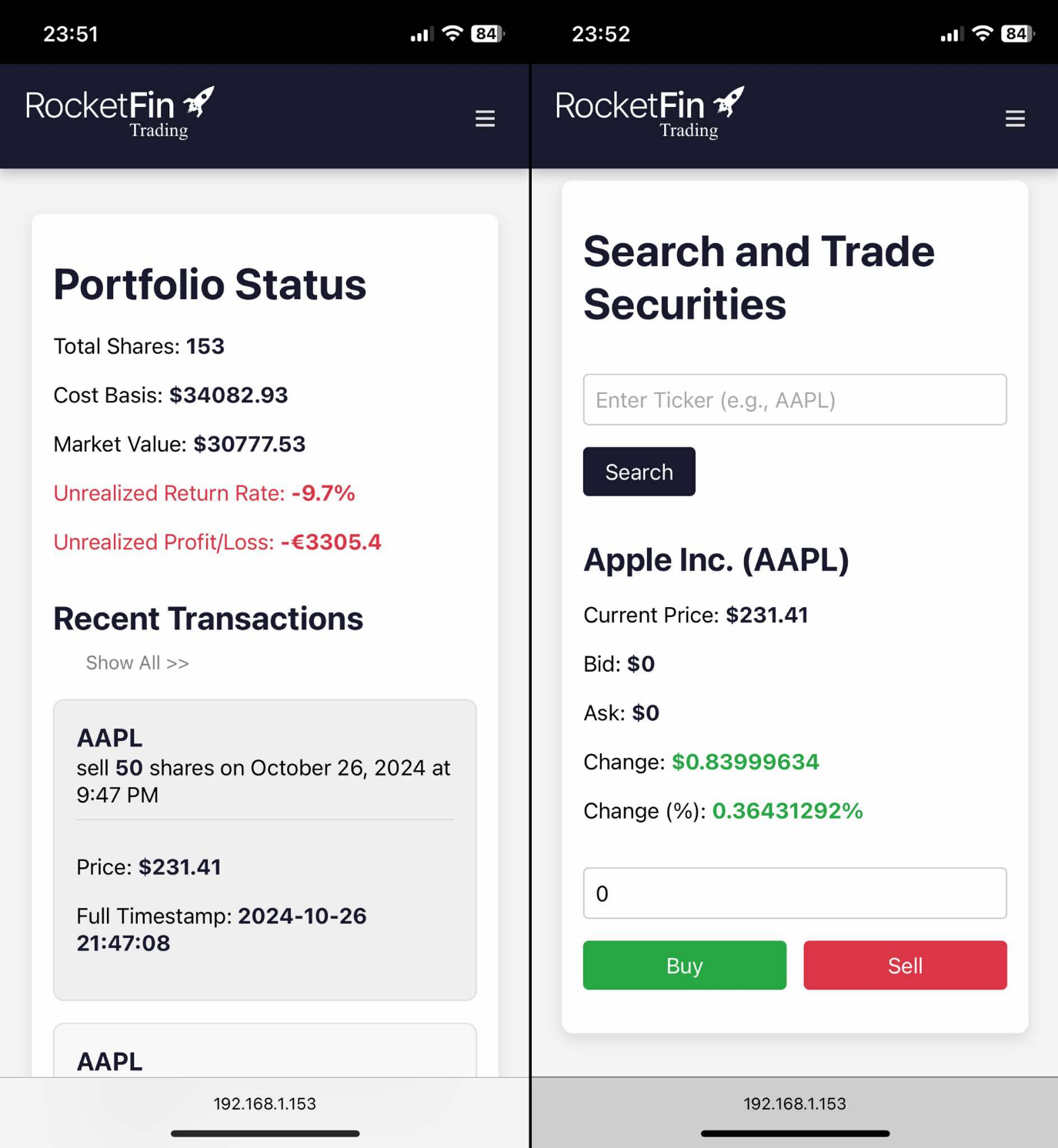


Figure 16 – Example of two application screens running on mobile devices

# Swagger API Documentation

The Flasgger Python package was utilized to document the various API routes utilized in this application. After starting the backend service, the Swagger documentation can be accessed at <http://localhost:5000/apidocs/>.

A screenshot of a computer

Description automatically generated

Figure 17 – Default Swagger page for this application

Each route contains a thorough description of what it does as well as the acceptable parameters, and an example of valid and invalid responses. Each route can be tested out to ensure that a valid response is shown.

A screenshot of a computer

Description automatically generated

Figure 18 – Example usage of the instrument search route

# Unit Tests

Fourteen different tests were written using Pytest to test the routes thoroughly. A 92% coverage was achieved with these tests showing a great overall coverage.

These tests can be run by executing the command, *‘pytest --cov=app tests/’*, inside the backend root folder and further details can be extracted from the result by generating the html results with the command *‘coverage html’.*

**Warning:** these tests will delete data from the database to test the error handling. It is thus imperative to note that any data within the database will be removed when running these tests. This data can be backed up by making a copy of the ‘rocketfin.db’ file within the instance folder in the backend root folder, and replacing this after the tests complete.

A black screen with green lines

Description automatically generated

Figure 19 – Output of successfully running the unit tests