CSC 336 – Database Systems
Project #2:
Full Stack Application
Team Name: Quantum Quacks

Team Members:

Quoc Do (23776279)

Tanvir Youhana (23783099)

Md Islam (23649266)

Refat Monjur (23531671)

Emails:

qdo000@citymail.cuny.edu

tyouhana000@citymail.cuny.edu

mislam033@citymail.cuny.edu

rmonjur000@citymail.cuny.edu

Professor. Richard Khan

Heroku Deployment Link: https://quantumquacks-frontend.herokuapp.com/

Github Link: https://github.com/Tanvir-Youhana/Database-Project-2-Fullstack

Database Final Project Demo - Quantum Quacks

Documentation

    a.   What is the purpose of this software?

The purpose of this software is to allow users to input their prediction of a particular stock onto the website. The website will then tell them if their prediction was right or not once the time has come. This website features useful tables to a typical investor such as IPO Calendar, Earning Calendar, and Market Holidays.

In the future, we will make a leaderboard that shows the ranking of all the users. The more accurate and ambitious prediction the user makes, the higher he/she will be on the leaderboard. This leaderboard will fully utilize the stock prediction table as it will show which user deserves to be noticed. However as of right now, the prediction table can only be seen by the one who created it and by the admin as all the stock entries are stored in a database.

b. Why is the software interesting?

What makes this software interesting is that it uses real-time stock APIs. In our search feature, we offer an interactive chart of the searched stock. There one can see the price action of a stock and its data much more detailed than popular website such as Robinhood where it provides line chart rather than candles chart. We also have a button called "Trending Tickers" which gets data from Yahoo Finance API in real-time. This means every few minutes, the table will update.

Lastly, but the most important feature is the stock prediction itself. As soon as you input the required information in the entry box, you will get the current price of your chosen stock and the expiration Date based on the time interval you chose. As of right now we have three choices, EOD (End of Day), EOW (End of Week), EOM (End of Month). This means that if you select End of Week, your stock entry will "expire" 7 days from now and so on. On the right side is the "Actual Table", and if you press the "Refresh Actual Table" button, it will update to the actual price of that expiration date.

c. What existing product is like the project you are proposing? Create a complete diagram of the application and how it will work.

TipRanks is a product that is most similar to ours. It is used by a lot of full-time investors and traders to get stock ideas. In that website, there are a lot of reputable analysts, many from Wall Street, and they give their own analysis on their chosen stock. TipRanks implemented a Success Rate in the analyst's profile to show how accurate they were. Our next step in the project is to again, make a leaderboard. The big difference is that anyone who signs up in our account can be on the leaderboard however on TipRanks, you have to be a credible, basically a Wall Street Analyst, which doesn't give any spotlight to underrated investors/traders. The design for our user interfaces were inspired by Robinhood and its minimal design.

Since our full stack application requires us to create a login and registration page which asks a user to login to gain access to the application, we first began with coming up with the design for the login and registration page. Upon looking at various login/registration pages, we liked how Facebook did it and we used theirs as a reference as shown below. The project required us to be able to create, read, update, and delete (CRUD) within the application, meaning there must be a database which needs to be used. We used MySQL as our database to store each user's credentials containing their Email address, Name and Password.
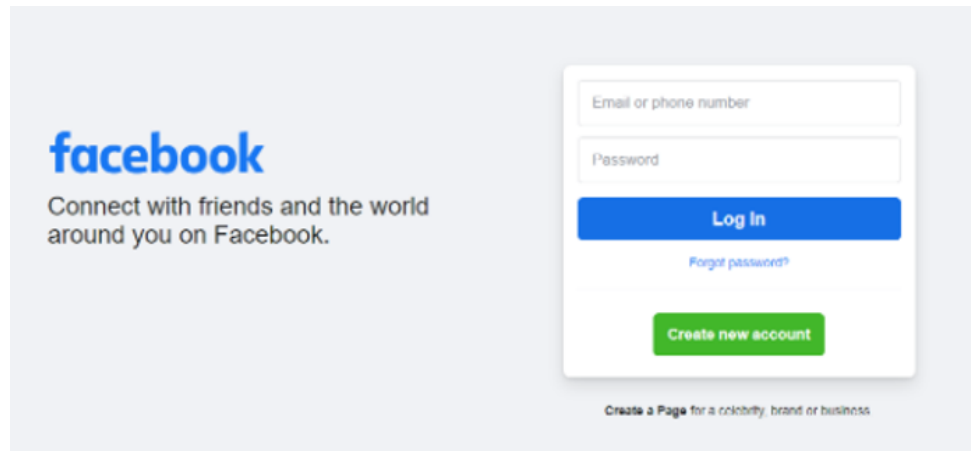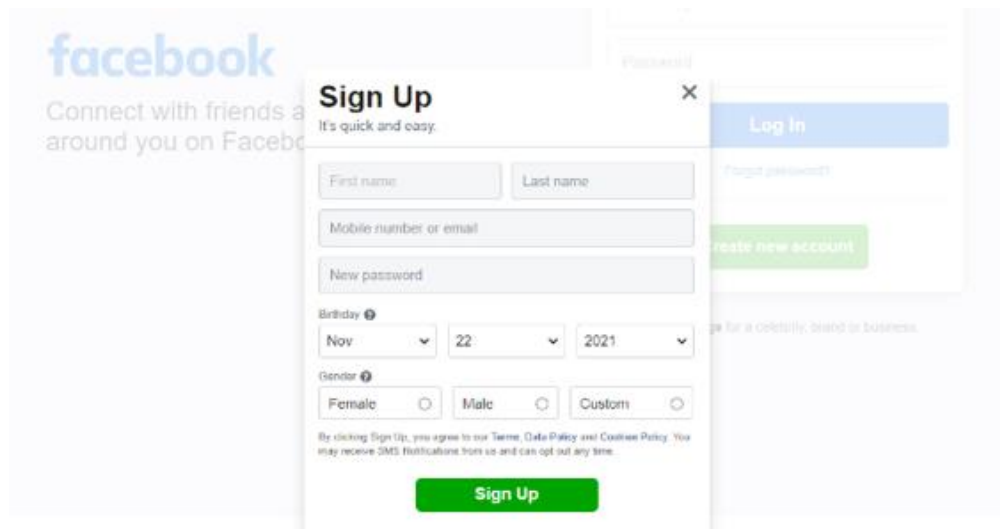
Figure 1: Login Screen Reference

Figure 2: Registration Page Reference

SCHEMAS FOR DATABASE:

We have total of 7 tables. User table stores the information of the user after registration. This

table is being used to track the user throughout the website. For example, when the user registers,

logs In or update password, the user table is being called. The Stock entry table keeps tracks of

the users prediction list. User can add new stocks to their prediction list and this table will be

updated or when user deletes a stock from the list, this table is updated. The table check_entry

depends on the stock_entry table. These two tables have one to one relationship. When the user updates their stock list, the check entry table is being reflected on it. Stock entry table shows the actual price of each of the stock_entry table and shows the users prediction accuracy. The tables trending_tickers, earning_calendar, ipo_calendar, and market_holidays depend on the user table only. The relationships of all these tables are shown on the ER diagram below:
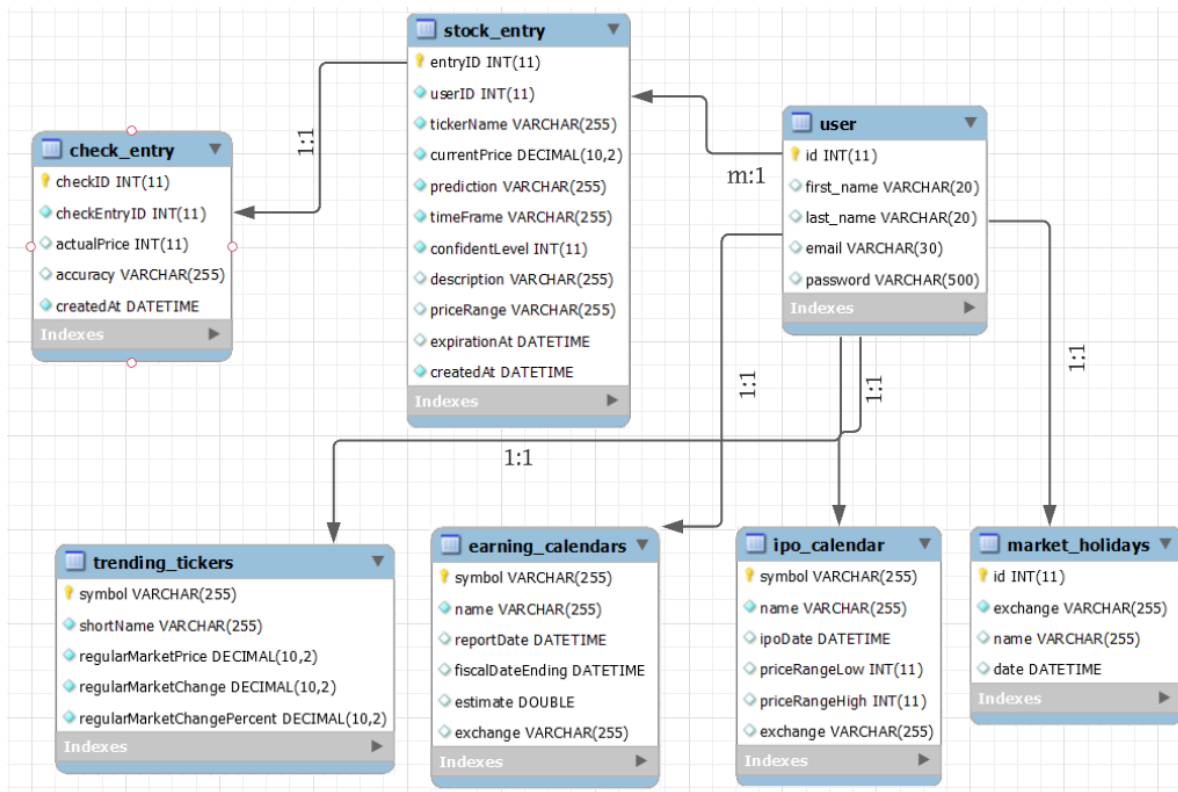
MANUEL/ TUTORIAL:

-> Registration/ Login:
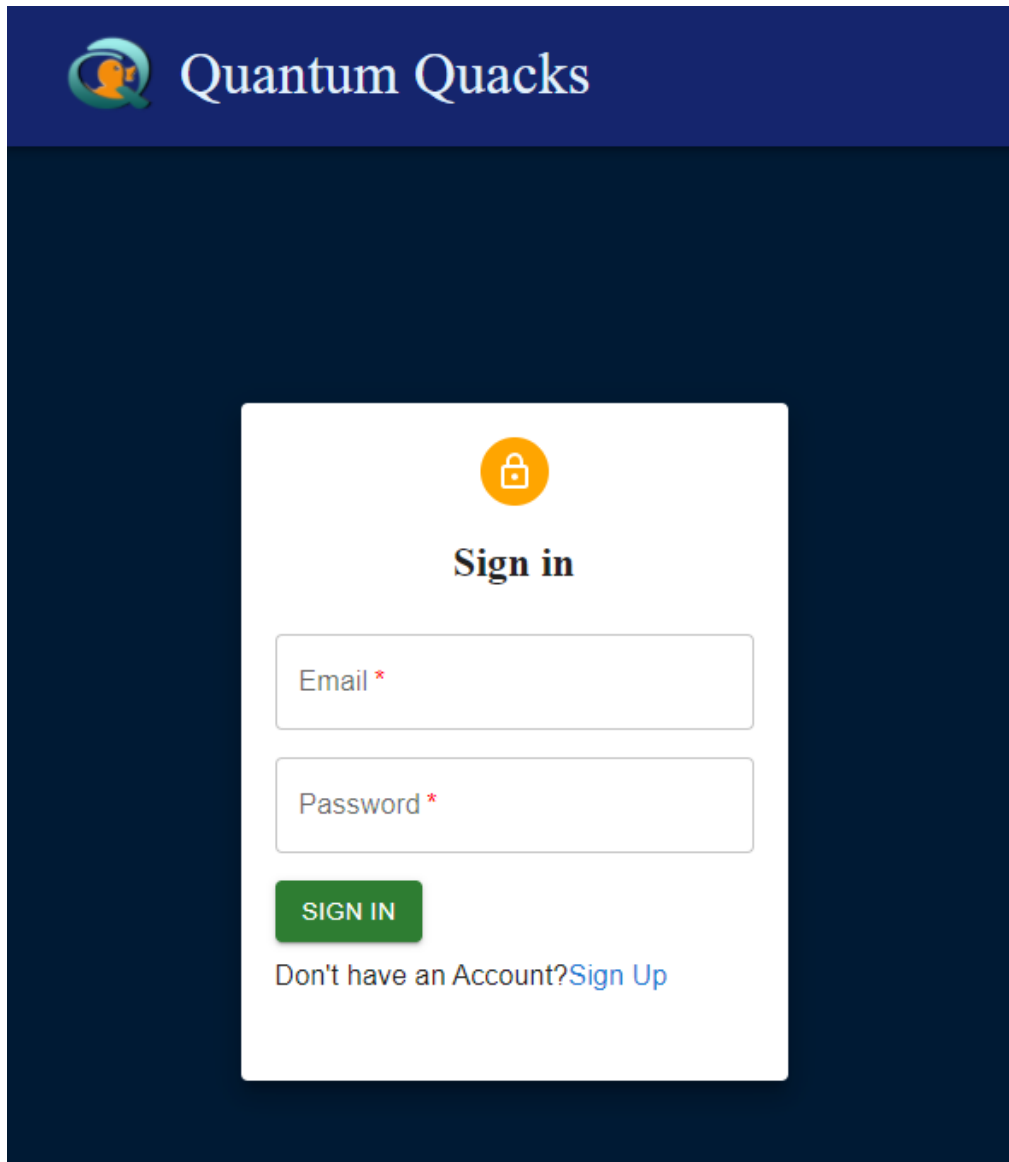
Figure 3: Quantum Quacks Launch Screen

This is the launch screen of our application. Upon loading up the application you're presented with a login screen requiring you sign in with your credentials. If the user does not already have an existing account, they can create a new account by clicking sign up which will forward the user onto the registration page.
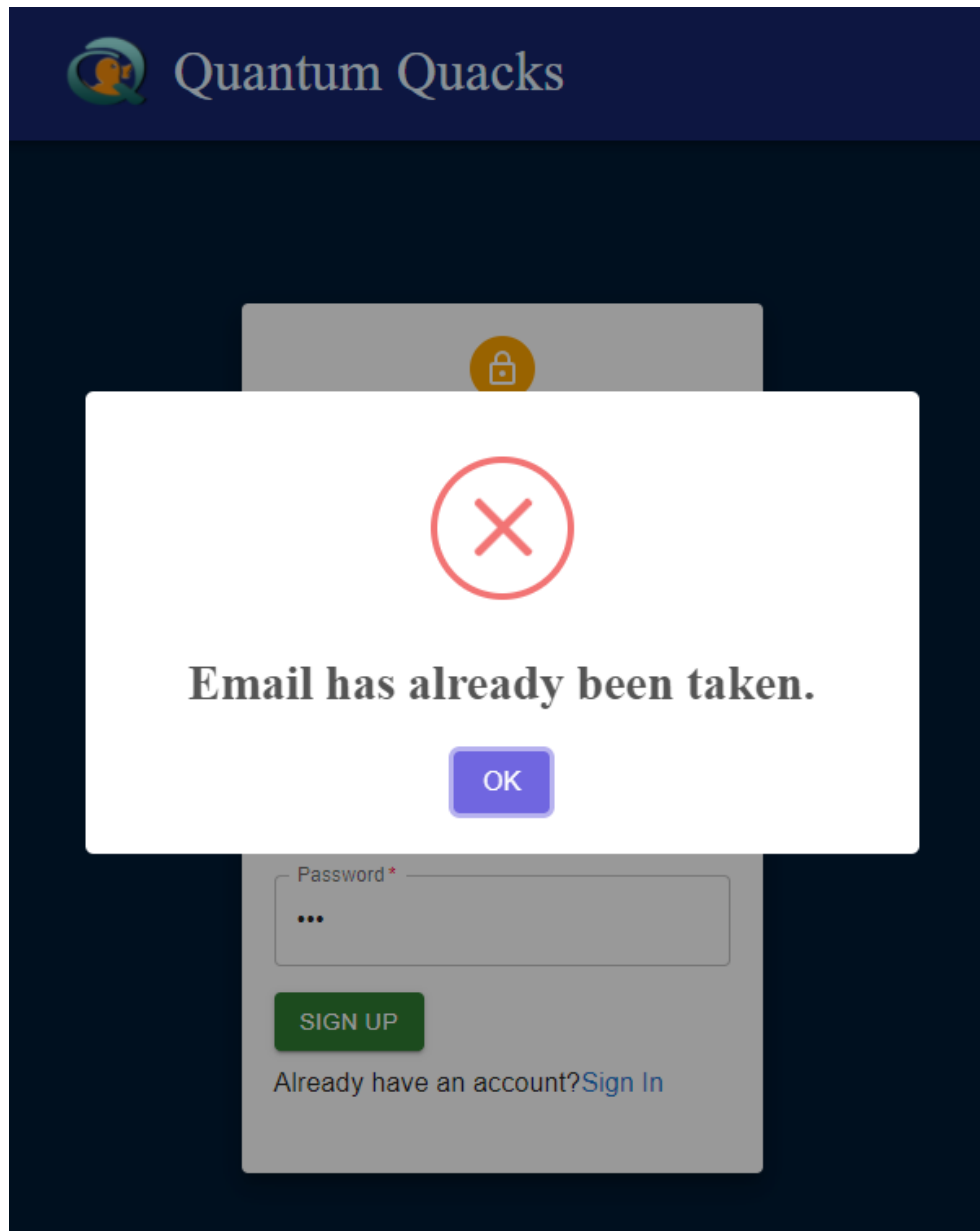
Figure 4: Registration Page

Once the user clicks sign up, the user will be forwarded to the registration which requires the user to input their credentials such as their name, email address and password to create an account.

Figure 5: Account already exists

When the user creates a new account, they need to use a valid email address which does not already exist in the MySQL Database. If the email address already exists in the database, the user will receive the message as shown above in a pop out window as an error which states that the email has already been taken.

id	first_name	last_name	email	password
1	Esra	Meriot	emeriot0@ucoz.ru	CK8txBfLjq
2	Sibella	Guwer	sguwer1@hao123.com	mZDnpf
3	Joni	MacCostye	jmaccostye2@rediff.com	aMaHNTakWvXq
4	Gabbie	Nuzzti	gnuzzti3@craigslist.org	TBEumXdIz
5	Wernher	Scholler	wscholler4@sourceforge.net	CgIKvP
6	Maurits	Bryson	mbryson5@constantcontact.com	zhsmzc
7	Rosanne	Gawkes	rgawkes6@github.io	7CysXTC
14	t	t	t	$2b$10$2cfmDrNcSFSNbI/x0UCTkejFQcL0kNRc...
24	hey	dfldsf	p	$2b$10$r4xk9pe97zDKk5Npz5/.M.rN/1vFcMpw...
74	Andy	andy	andy	$2b$10$Vg.yOGhjKyXHP1SllCMJLeTY9obHbFPR...
84	Quoc	Do	quocdo@gmail.com	password
94	Bob	Ross	bobross@gmail.com	bobross
114	m	m	m	$2b$10$fD5I5hcuB5HjIB7fzEVpYOmXQRvdE0SX...
124	q	q	q	$2b$10$tj3F.pSywRplg8XUHPJnruLibl2SuAQZG...
134	John	Nunez	john@gmail.com	$2b$10$1IsbIKbc88okEXakB0e62.aRJ.S.F3dcfA...
NULL	NULL	NULL	NULL	NULL

Figure 6: MySQL Database 1

This is the database which is connected in the backend that stores and retrieves the data containing users' credentials. As you can see the email address john@gmail.com already exists. The backend retrieves and reads this data and notifies the user with the message stating the email is already taken.
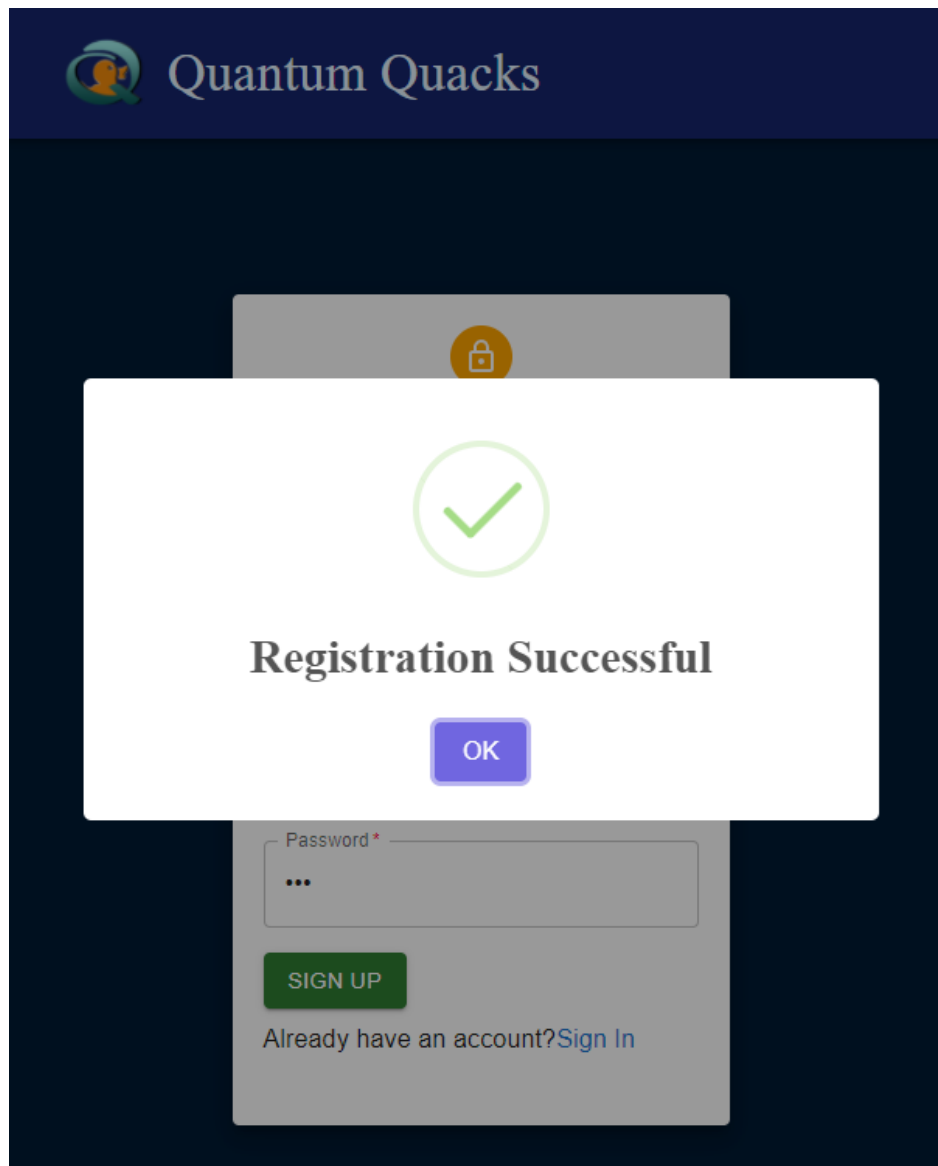
Figure 7: Create Account

When you create an account successfully, you're presented with this pop out window notifications stating registration is successful. This creates a new account in the database, assigning the account with and id. Attaching that id to their name, email, and password.

id	first_name	last_name	email	password
1	Esra	Meriot	emeriot0@ucoz.ru	CK8txBfLjq
2	Sibella	Guwer	sguwer1@hao123.com	mZDnpf
3	Joni	MacCostye	jmaccostye2@rediff.com	aMaHNTakWvXq
4	Gabbie	Nuzzti	gnuzzti3@craigslist.org	TBEumXdIz
5	Wernher	Scholler	wscholler4@sourceforge.net	CgIKvP
6	Maurits	Bryson	mbryson5@constantcontact.com	zhsmzc
7	Rosanne	Gawkes	rgawkes6@github.io	7CysXTC
14	t	t	t	$2b$10$2cfmDrNcSFSNbI/x0UCTkejFQcL0kNRc...
24	hey	dfldsf	p	$2b$10$r4xk9pe97zDKk5Npz5/.M.rN/1vFcMpw...
74	Andy	andy	andy	$2b$10$Vg.yOGhjKyXHP1SllCMJLeTY9obHbFPR...
84	Quoc	Do	quocdo@gmail.com	password
94	Bob	Ross	bobross@gmail.com	bobross
114	m	m	m	$2b$10$fD5I5hcuB5HjIB7fzEVpYOmXQRvdE0SX...
124	q	q	q	$2b$10$tj3F.pSywRplg8XUHPJnruLibl2SuAQZG...
134	John	Nunez	john@gmail.com	$2b$10$1IsbIKbc88okEXakB0e62.aRJ.S.F3dcfA...
144	John	Nunez	johnnunez@gmail.com	$2b$10$1dp5bIdqWTvcC2DbGuPURuC7U4/0yw...
NULL	NULL	NULL	NULL	NULL

Figure 8: MySQL database with new user

Above you can see the new user's credentials added successfully onto the database
containing their name, email address and password. We implemented a feature which encrypts
the password so any administrator with access to the database is unable to see the users'
passwords for privacy concern and to prevent hacking.
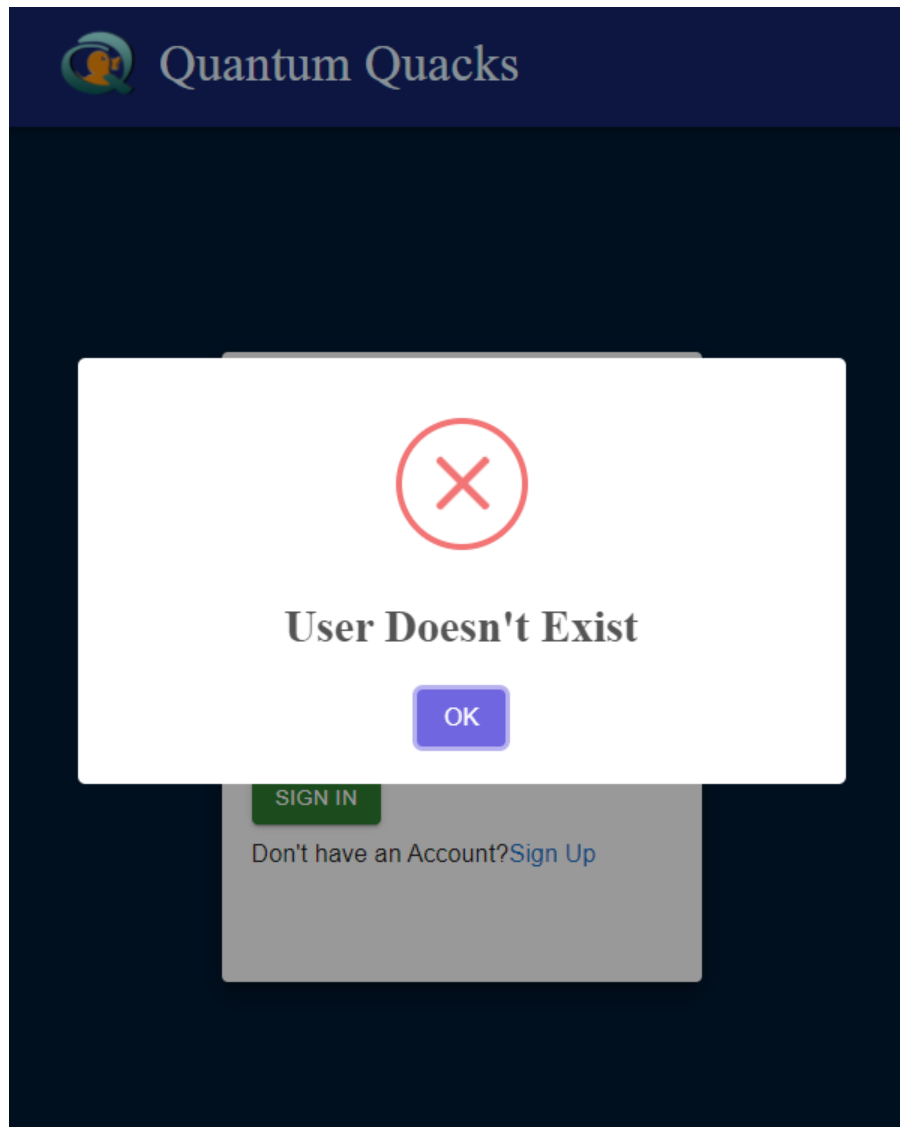
Figure 9: Login Screen Error Case 1

When a user tries to login using an invalid email address that does not exist in the

MySQL database, this is one of the possible outcomes. They'll see a pop-up window with the

message "User Doesn't Exist" to let them know there's no account associated with that email
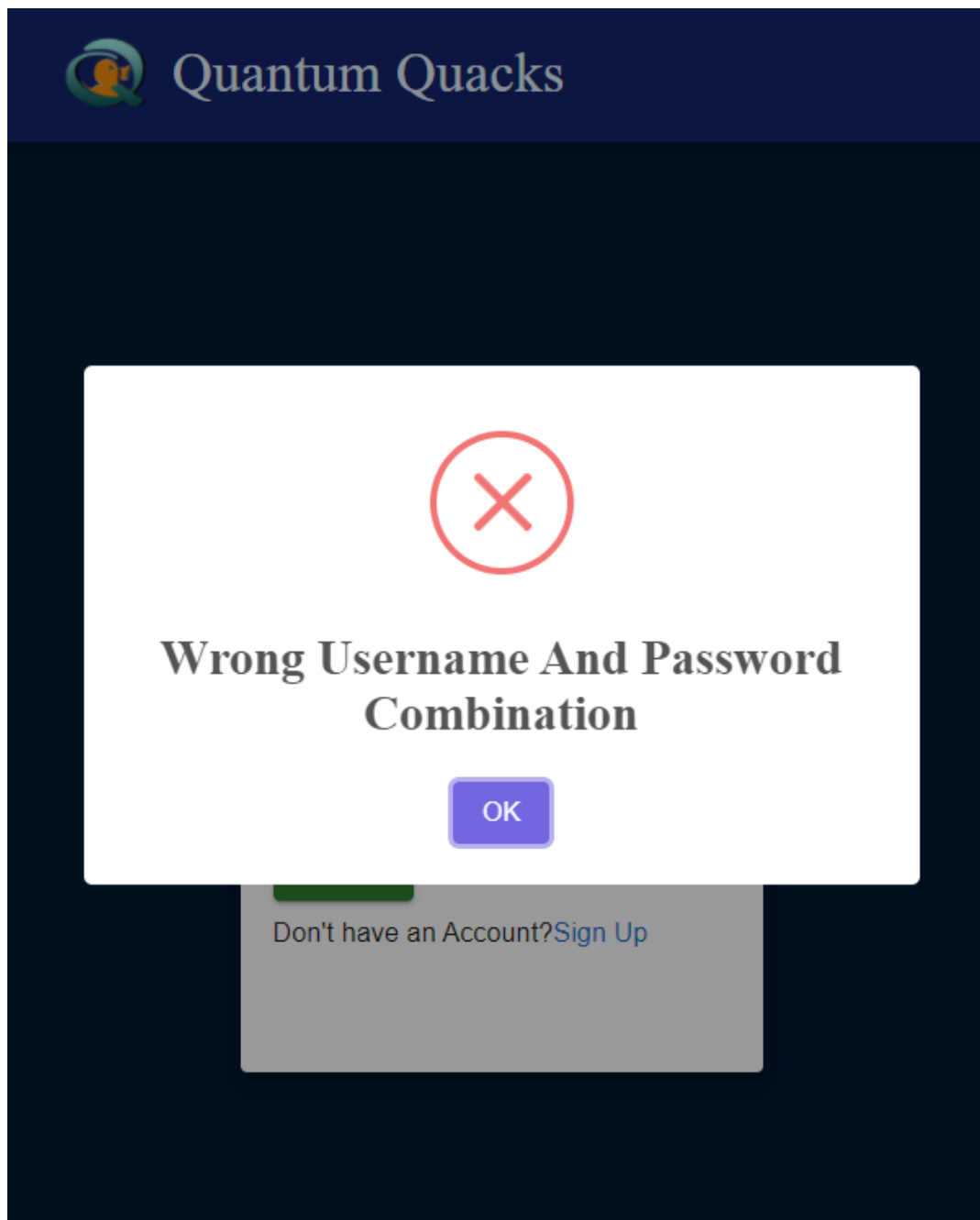
address.

Figure 10: Login Screen Error Case 2

This is another one of the cases on the login screen. When the user tries to login with an invalid password but an account with an existing email address within the database. They'll receive this pop out window with an error message stating that the username and password combination is wrong after retrieving the information from the database.

Even though we used MySQL for our database, we wanted to move it to the cloud rather than hosting it locally. As a result, we moved our database to the cloud and hosted it on Heroku.


```
backend >  ⚙ .env
  1     DB_HOST =
  2     DB_USER =
  3     DB_PASSWORD =
  4     DB_DATABASE =
  5     DB_PORT = 5000
```

Figure 11: Connection to Database on Heroku

```
6     alphaVantageKey =
7     polygonKey =
8     yahooKey =
9     quandlKey =
```

In our backend we created an .env file. The env file contains the individual user environment variables that override the variables set in the environment file. This file makes the connection to the database on Heroku. The other picture are the stocks API key. One will need this to access the calendars.

Before logging in:
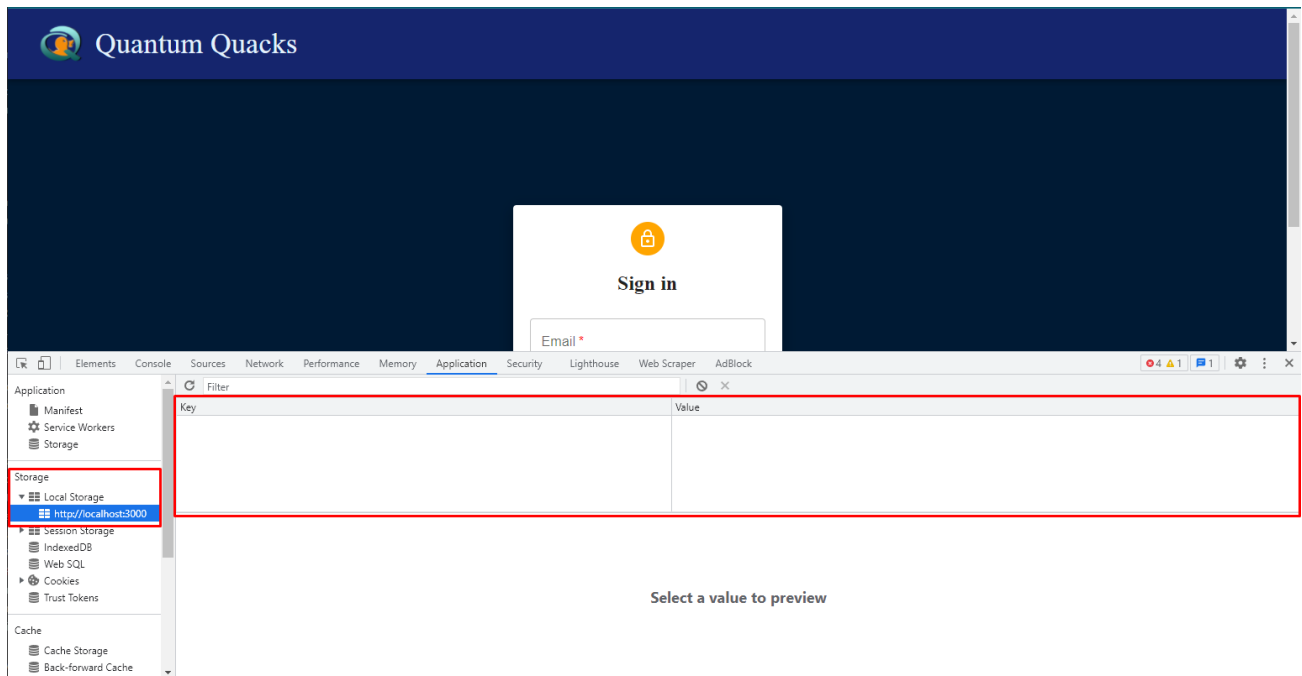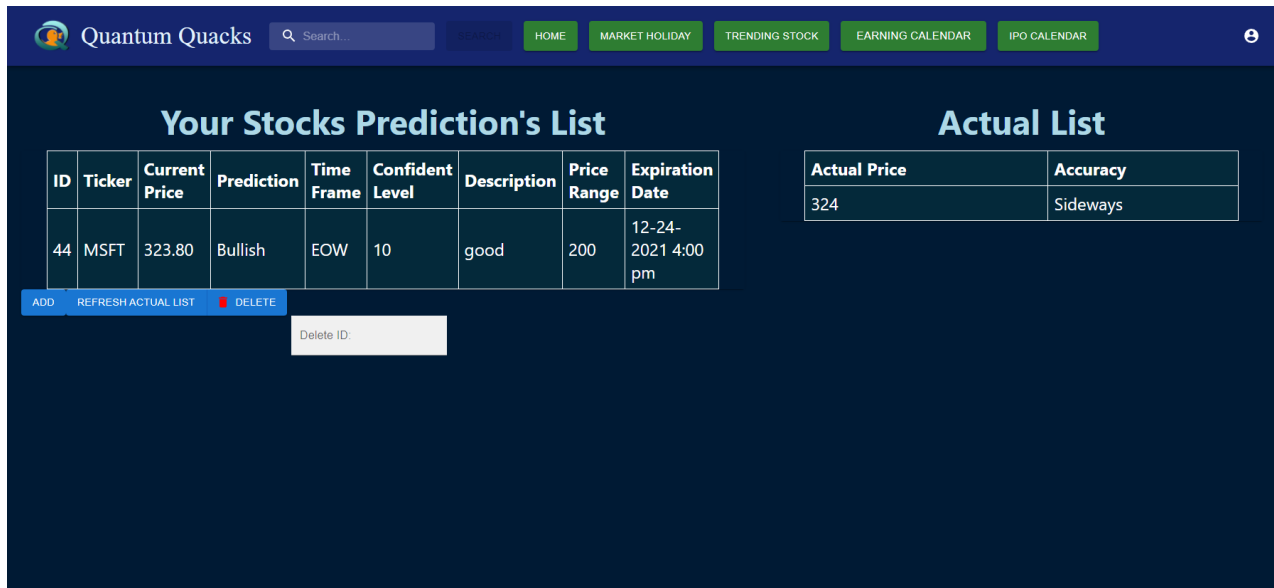
Before logging in, if we go to the localstorage we see that nothing is currently stored there.

After logging In:
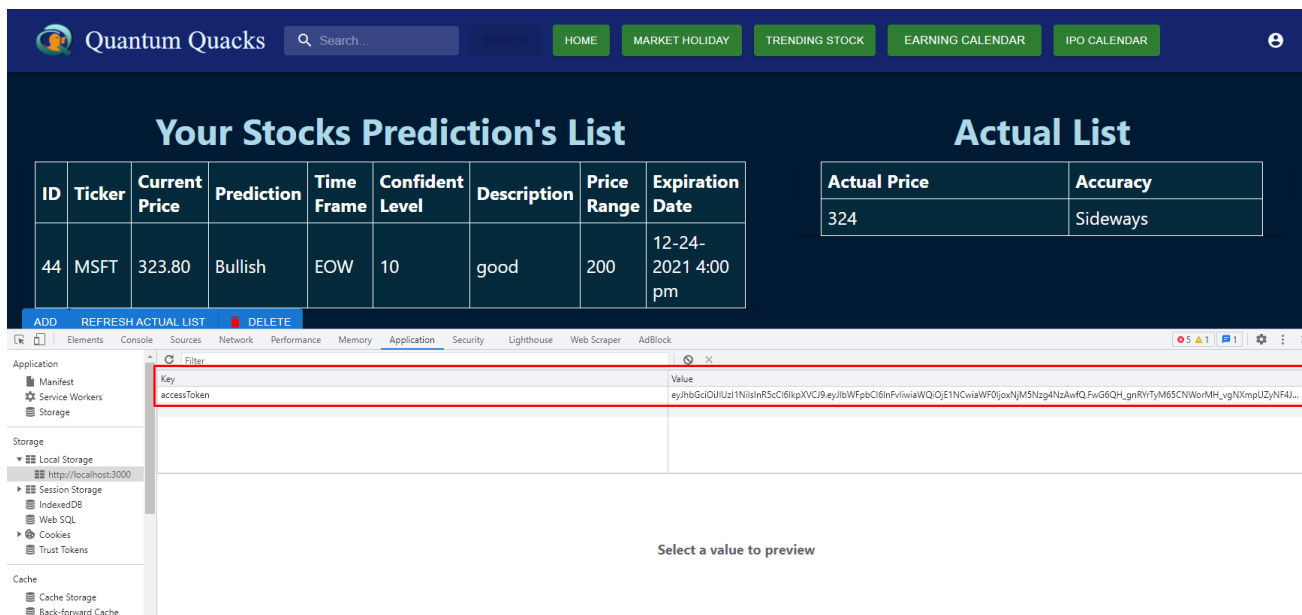
After we logged in, we see that the Local Storage contains a key and value. Once a specific user logs in ,

they get an accessToken, this is stored locally until the user logs out. We see that value is some random

number, this is because the value is being hashed, for security.

Pages:

Home

After the user successfully logs in, they are directed to the home page, the home page consists of the users Stocks Prediction List, and another table named Actual List. The Stock prediction list is unique to each user. The user can add or delete stocks from the Prediction List.

Adding a stock to the stock prediction List:

When the user clicks ADD button on the home page, it takes to a Dialog which asks for user inputs as shown in the figure above. Now when user add the stock entry, it adds the new stock to the users prediction List as shown in the figure below.

Invalid Ticker checker:

If a user puts invalid ticker name, it should not be added to the prediction list, and a alert message will be shown.

The above figure shown is when user attemps to input invalid tickers to their list.

Deleting stocks from the Stocks prediction List:

Let's say user would like to delete the stock with ID 44. The user can delete a stock by entering the ID of

the stock on the Delete ID field.

We see that the new stock is added to the users stock table and the actual List table is also updated with

AAPL stock price along with how accurate the user was with its prediction. Shown below a process of

deleting a row from the stock prediction List:

After user deletes the row with ID 44, it should delete the MSFT stock from the Stock prediction List and

the Actual List table should also be updated as shown in the figure below.

We see that both table is updated and the MSFT stock is deleted from both tables.

Search:

Users can search for a stock and see its details from a historical graph of that stock.
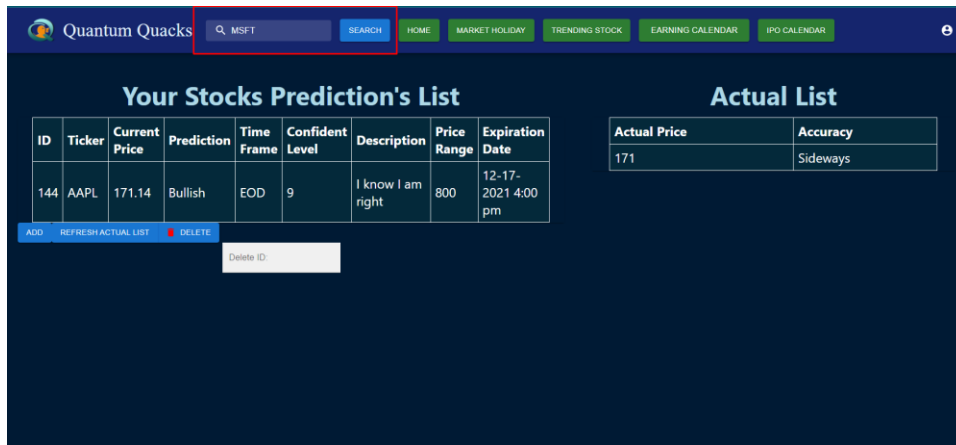
After user searches for a stock it shows the Historical chart of that stock. User can choose to zoom to

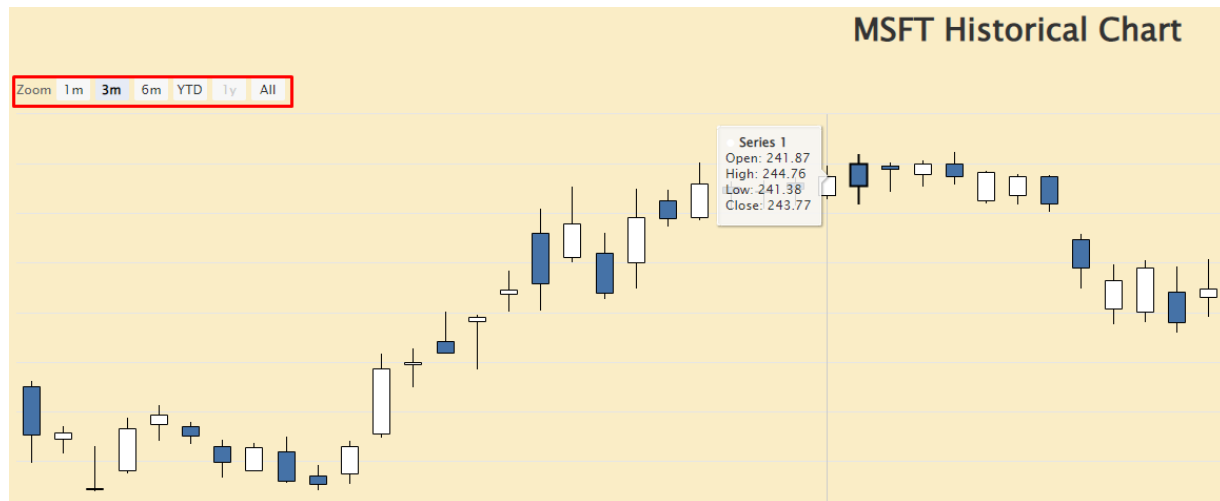filter the data as shown in the figure below.

MSFT Historical Chart

Zoom 1m **3m** 6m YTD 1y All

Series 1
Open: 241.87
High: 244.76
Low: 241.38
Close: 243.77

Figure 12: Search Bar

A search box in our navigation bar allows visitors to look up stocks by symbol or ticker name. You'll see an error popup if you use an invalid symbol/ticker name. When you search for a stock, you'll see a chart of the stock's performance over time. The chart shows the stock's performance over the course of a year, and you can see how it changed over time.

MARKET HOLIDAY

Name	Date	Exchange
Christmas	2021-12-24	NASDAQ
Christmas	2021-12-24	NYSE
Martin Luther King, Jr. Day	2022-01-17	NYSE
Martin Luther King, Jr. Day	2022-01-17	NASDAQ
Washington's birthday	2022-02-21	NYSE
Washington's birthday	2022-02-21	NASDAQ
Good Friday	2022-04-15	NYSE
Good Friday	2022-04-15	NASDAQ
Memorial Day	2022-05-30	NYSE
Memorial Day	2022-05-30	NASDAQ
Independence Day	2022-07-04	NYSE
Independence Day	2022-07-04	NASDAQ
Labor Day	2022-09-05	NYSE
Labor Day	2022-09-05	NASDAQ
Thanksgiving	2022-11-24	NYSE

Figure 13: Market Holiday

This is the market holiday page, which includes a calendar with all of the market's closed days for the following year.

Symbol	Name	Price	Price Change	Percent Change
AAPL	Apple Inc.	179.30	4.97	2.85
AI	C3.ai, Inc.	35.01	3.83	12.28
AMD	Advanced Micro Devices, Inc.	146.50	10.90	8.04
BTC-USD	Bitcoin USD	49136.60	2329.37	4.98
CCMP	CMC Materials, Inc.	195.50	49.53	33.93
CGX.TO	CINEPLEX INC.	13.14	1.37	11.64
CRWD	CrowdStrike Holdings, Inc.	210.71	12.00	6.04
ETH-USD	Ethereum USD	4061.54	299.50	7.96
IOT	Samsara Inc.	24.70	24.70	2936.59
LLY	Eli Lilly and Company	275.28	25.90	10.39
LMND	Lemonade, Inc.	43.89	2.61	6.32
MDT	Medtronic plc.	104.94	-6.75	-6.04
NIO	NIO Inc.	30.79	-1.51	-4.67
NUE	Nucor Corporation	108.22	-10.20	-8.61
NVDA	NVIDIA Corporation	304.59	21.22	7.49

Figure 14: Trending Stock

You'll be routed to this page with the current trending tickers after clicking the trending stock in the navigation bar. The stock symbol/ticker, the company name, the current price of the stock, the price change, and the percentage change are all loaded from the finance API every time you're routed to this page.

Figure 15: IPO Calendar

The IPO Calendar (Initial Public Offerings) is displayed above. This page will show you which firms are preparing to go public soon, as well as when their IPOs will take place.

Figure 16: Earning Calendar

The Earning Calendar page is shown above. This page provides the most up-to-date information about companies that are expected to release earnings results.
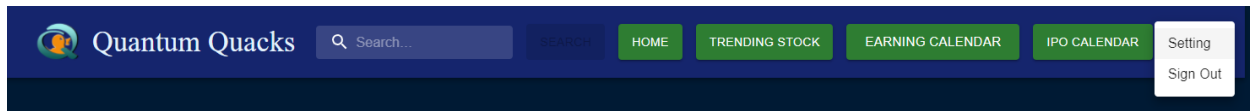
Figure 17: Navigation Bar

The user is presented with this navigation bar after logging in. Once logged in, users have the option of changing their password. They can go to the change password window by heading to settings and clicking on the right upper corner.
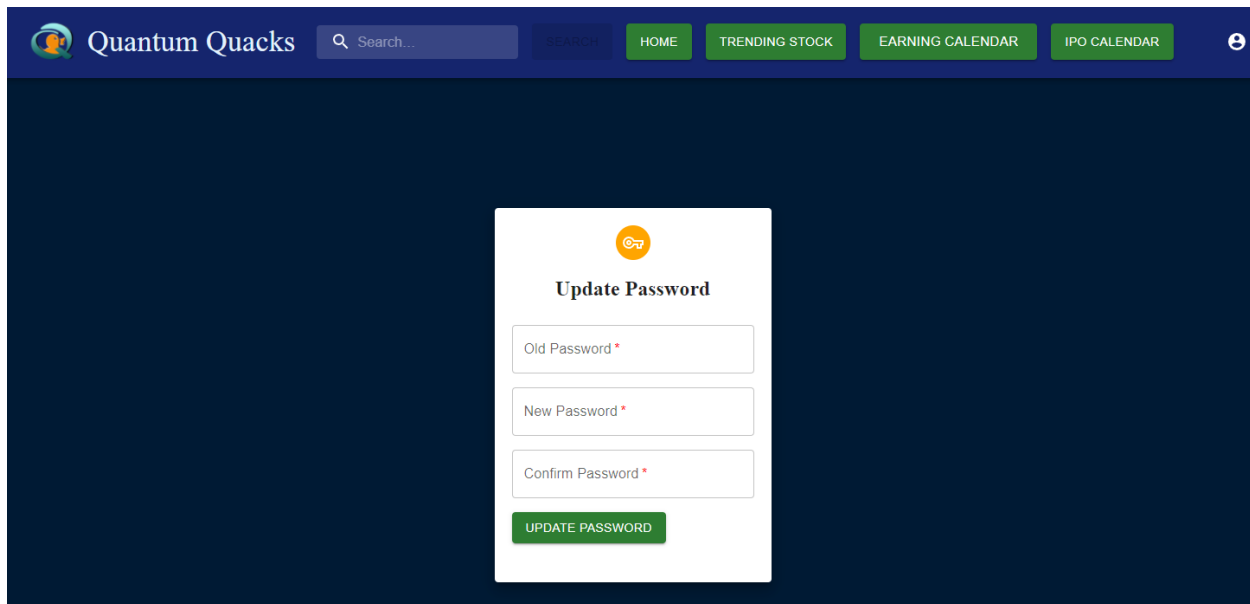

Figure 18: Change Password Window

When the user arrives at the Update Password screen, they must enter their current password to update and set a new one.
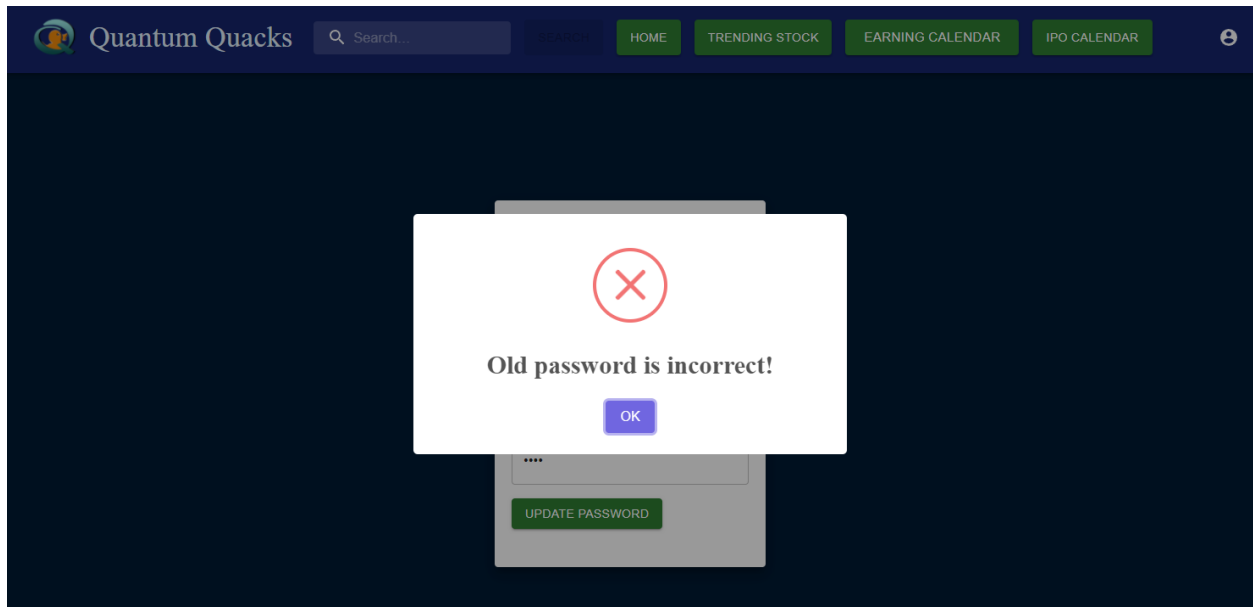
Figure 19: Change Password Error Case 1

This feature was implemented to protect the user, so any random user with access to the computer cannot just randomly change the host's password. Unless if the old password is current, they'll receive this error.
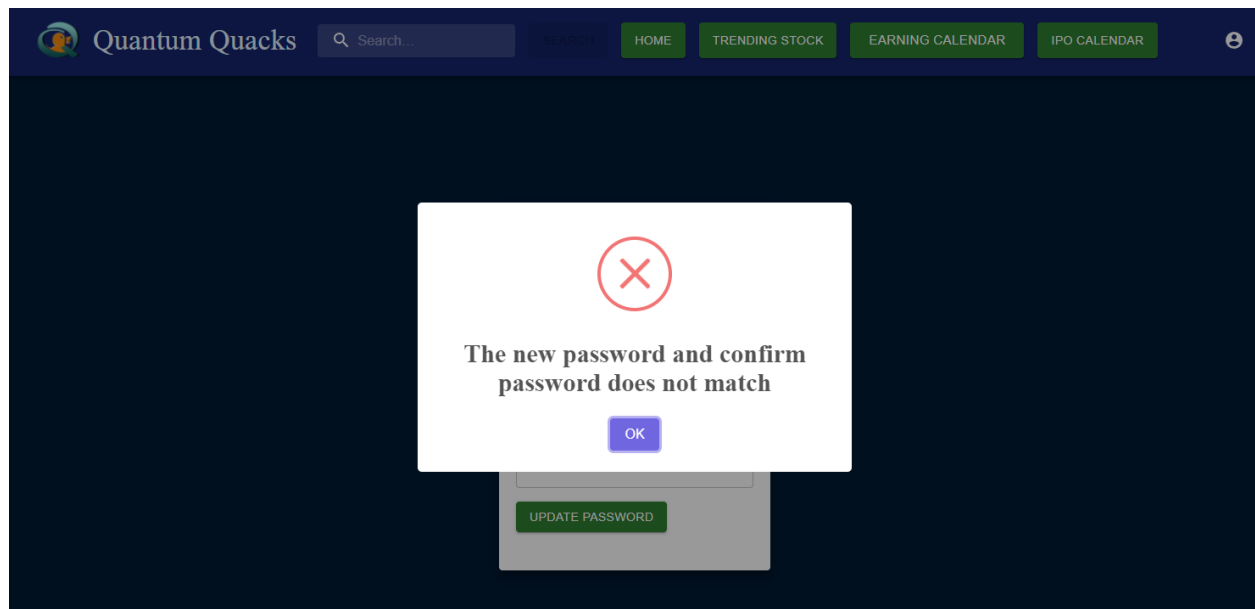
Figure 20: Change Password Error Case 2

When you input the new password, you must type in it again to confirm. This is set so you don't accidently type in a wrong password.

Figure 21: Password Changed Successfully

Once the password is changed successfully, the user will receive this pop out window with the message. This changes and updates the password in the database.

Figure 22: Local Storage signed in
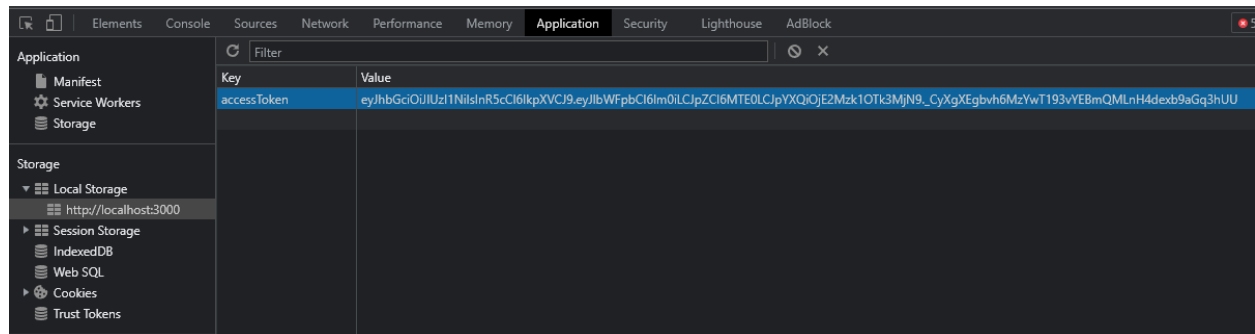
While the user is logged in onto the application, local storage contains an accessToken this is what keeps the user signed in as that accessToken is assigned to each user. The local storage can be viewed if you inspect element on your browser.
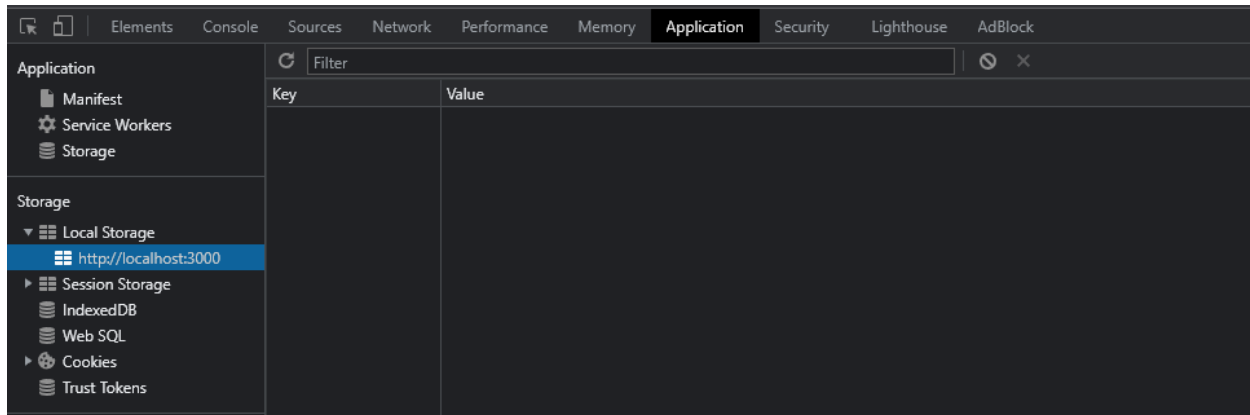
Figure 23: Local Storage signed out

After you sign out, local storage is empty, contains no key to keep user logged on and user data is not stored locally anymore.

We are aware that this is not the best approach to storing an accesstToken for security reasons and we know that we can use cookies to better implement accessTokens, however, we did not have enough time to implement cookies. We will be using cookies for our next fullstack application and learning how to implement them.