Assignment -3

JUNE 13

Authored by: Aditya
Student Number-n10555650

Logo
Name

Contents

Introduction:

"If you think math is hard, try web design."-Trish Parr. I was asked to make a Stock app for mobile phones using React native programming framework for Assessment -3. The ask for UI was to be as close to the apple app. The app skeleton was provided to me with two important screens to be implemented- The search and Stock screen. To add some improvement Login and registration page were asked to be implemented. The basic backend for the functionalities was asked to be implemented for higher grades.

It took me a great deal of time and energy to implement all the features that I could within the given framework of time. Although this was my first-time web development learning unit I was successful in implementing almost all the features for the front-end mobile app.

Technical Description:

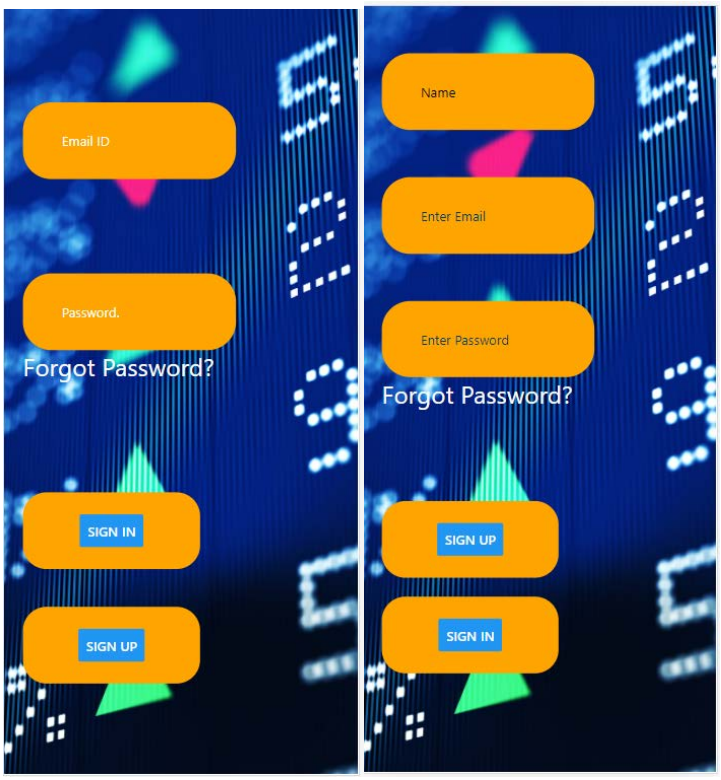
The app hits the Financial Modelling prep API for getting the values of the stocks and the search. The API for the chart is also taken from the same website but the API is different. One of the key difficulties that I faced during this period was to display the values in Virtualized list which was the only list available that offers a cell rendering feature. One of the features that are present in the apple stocks app is to get the charts and the table when one of the stocks is clicked. To implement this feature cell render was needed as it helps to improvise each cell through a common prototype code. The chart needs specific values from the API such as Open, Close, High, and Low which the First API wouldn't give hence this list was chosen and this was one of the trickiest dataflows I experienced in this project.

Although the data flow works in a format it is supposed to work there is a major warning that exists in the code for the react chart handler that was not done.

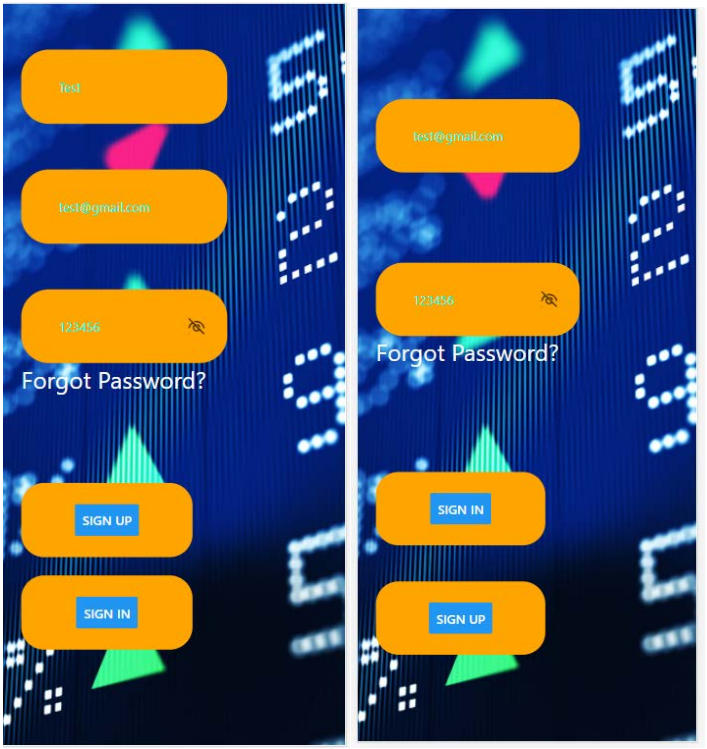
The warning wouldn’t stop the code from running or have any errors in the functionality. The other debug console works perfectly fine and wouldn't give out any warnings or errors.

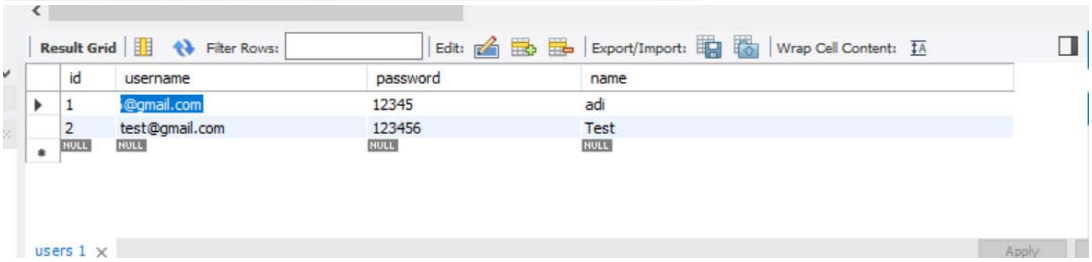
Testing and Limitations:

The Testing for the Search and the Stocks page was implemented and the testing screenshots are given below. The first page is the Login page and if the user doesn't have a login he can register it which will be saved in the backend of the project.

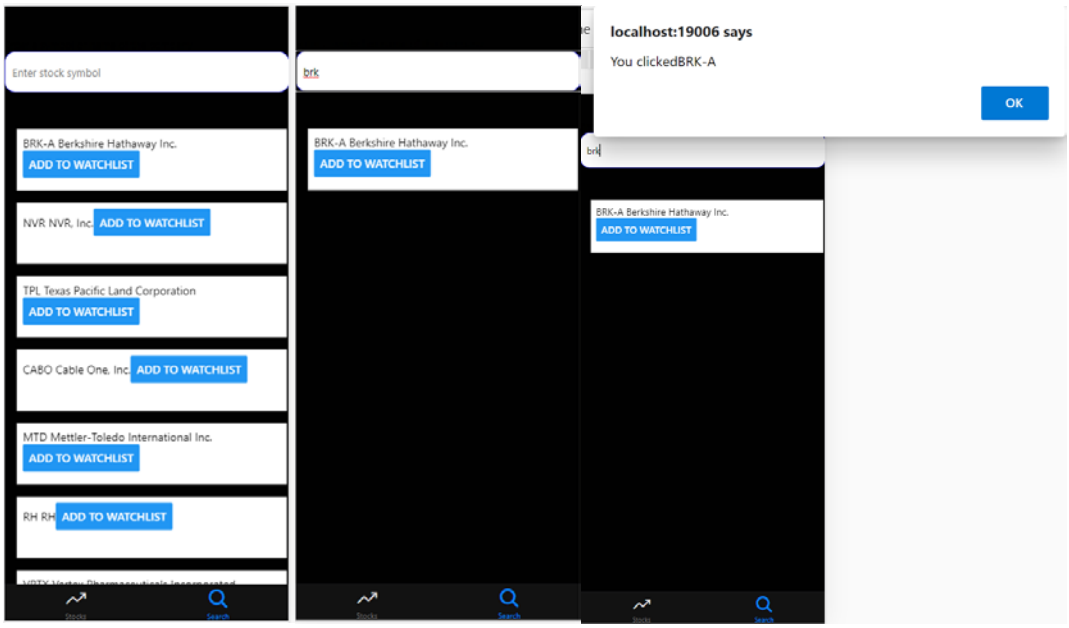


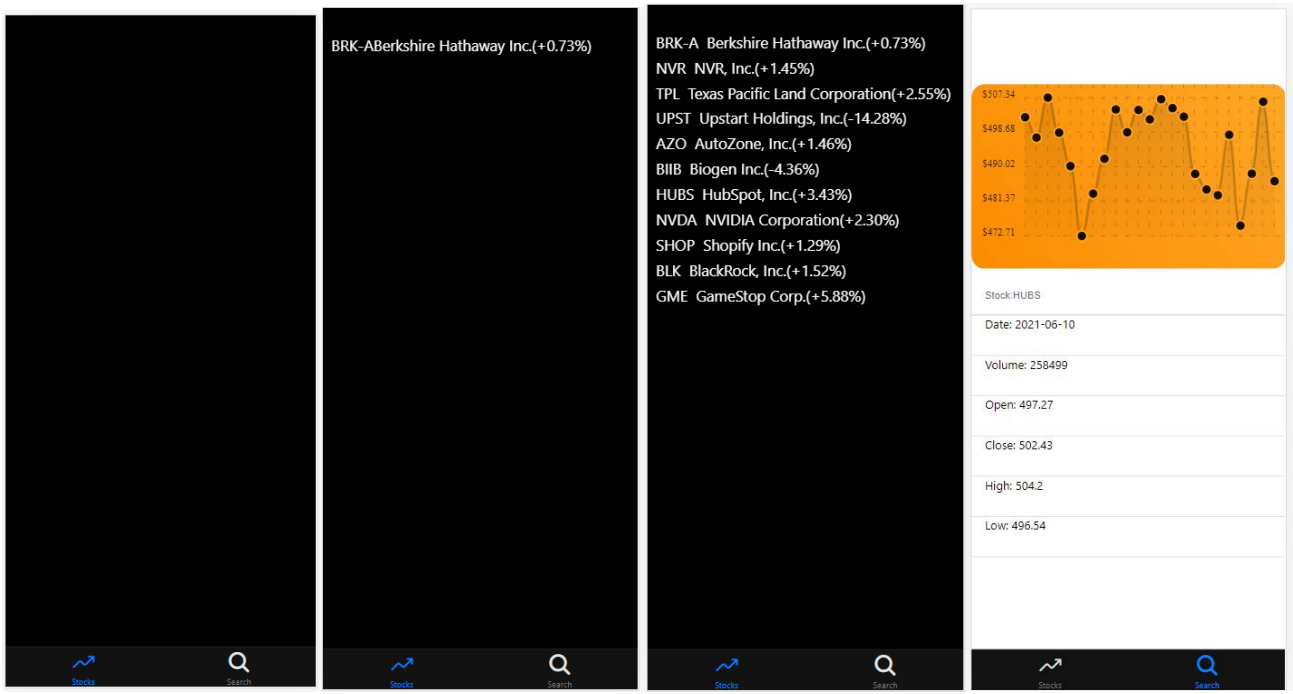
The backend server works on my system but I wasn't able to connect the Virtual machine to use it for checking. It would be great if you ping me while checking to start the system.





The Testing for Chart and the Table component for the particular stock was implemented and the screenshots are given below as follows:





Bugs-
The app has issues that the chart has different handlers but it hasn't been handled well.
The app has doesn't happen to have a proper backend so it doesn't happen to run on the phone for some reason.
The app doesn't have proper encryption for the token. The values for Username and Password are stored are directly and displayed directly.

Discussion:

Expo has some pros that I saw in this assignment

- Live development update
- Easy to use
- Can be used in both web development and mobile development software.
- Can be used on both platforms(iOS and android).
- Good documentation.

Cons

- Not all API is present
- Doesn't support background execution.
- The document to get published takes a lot of time
- The errors for the phone version aren't very clear.

Native support people with different disabilities for usage but the current app that is built doesn't support although.

The current app wouldn't have any legal issues if considered releasing.

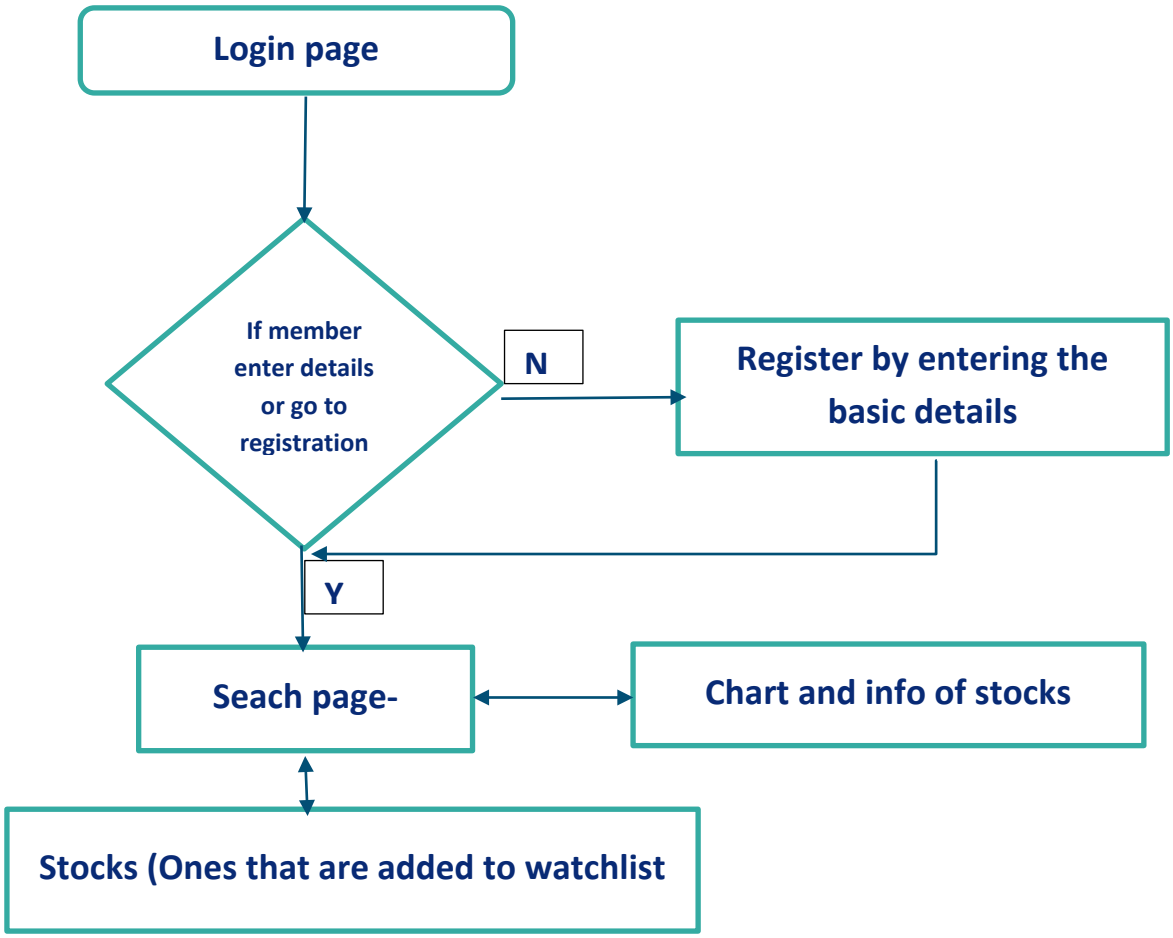
Reference:

1.Free Stock API and Financial Statements API - FMP API. (2021). Retrieved 13 June 2021, from https://financialmodelingprep.com/developer/docs/

2. indiespirit/react-native-chart-kit. (2021). Retrieved 13 June 2021, from https://github.com/indiespirit/react-native-chart-kit

Appendix:

Brief user guide-



The first page that comes when using the App is the Login page which helps the user to log in with the particular details. If the user does'nt have a login he can choose to register and then log in. There are two screens- Search and stock. The search screen can be used to search the stocks and if the user presses anywhere except the "Add to watchlist" in the particular list box he would be diverted to the chart page where the chart values are shown. To come back the user can click the Search button again. The

stocks page initially would have nothing but as the user adds the values are added into the Stocks page.

Video Demo with backend(Runs on my laptop so please ping for checking)
https://youtu.be/_ebfuc63rGU

Video demo of Expo on iPhone X
https://youtu.be/GkhHQxJDQWI

Assessment 3 Marking Criteria

	7 (85-100) %	6 (75-84) %	5 (65-74%)	4 (50-64)%	3 (40-49)%	2 (25-39)%/1 (<25%)
Front-end Mobile Application Functionality (20 marks)	The client side application implements all of the functionality as listed, and is thoroughly professional in its implementation and performance.	**The client side application implements all of the functionality listed as necessary, and is mostly professional in its implementation and performance. Some high level features may have minor issues or limitations.**	**The client side application implements all of the functionality listed as necessary to a good standard.**	**The client side application implements the basic functionalities to a good standard, but there may be some minor issues.**	**The client side application implements some of the functionality listed but the functionalities are poorly implemented.**	**The application does not work or does not meet the specification or is fundamentally flawed.**
Front-end Application Robustness (10 marks)	The application is robust, without any noticeable errors and handles service failures and errors gracefully.	**The application is robust and executes without noticeable errors and handles gracefully most service failures.**	**The application is robust and executes with only occasional errors and handles gracefully most service failures.**	**The application basically works, but the robustness is questionable and there are obvious errors and service failures are not handled well.**	**The application basically works, but there are numerous errors and failures.**	**The application is markedly incomplete and results in numerous errors.**
Front-end Application UI Design (10 marks)	The application user interface looks professional, and the principal use	**The application looks professional mostly, and the principal use**	**The application is clean and generally uncluttered, and the principal use**	**The application is somewhat clean and uncluttered, but overall the**	**The application is poorly laid out, but the basic functionality remains usable.**	**Layout is poor and usability very weak. The site is barely usable.**

Note: this is purely based on function and usability. There is no direct assessment of the quality of the graphic design.	cases are a natural fit for the user interface, allowing execution without undue effort or confusion. The user has no trouble navigating the app.	cases are readily executed, though there may be some clumsiness in the workflow. The user generally has little trouble navigating the app.	cases are reasonably easily executed, though there are some questionable choices in the design. Navigation may not always be straightforward.	design is clumsy and doesn't operate very well. Navigation may present some problems. .	Navigation is difficult.	
40 Marks	37					
Backend functionality, error responses and application reliability. (20 marks)	The server successfully implements all of the API endpoints according to specifications [or above and beyond expectations]. [See discussion of 6-7 levels in the spec.] The application is robust and executes without noticeable error and handles service failures and errors gracefully. Error conditions are returned and response codes are perfectly according to specification.	The server supports all of the API endpoints but there are some limitations in the implementation. [See discussion of 6-7 levels in the spec.] The application is robust and executes without noticeable error and handles service failures and errors gracefully. Error conditions are mostly returned in accordance with the API spec, with perhaps occasional incorrect response codes.	The server supports some API endpoints. The application is mostly robust and usually executes without error. Service failures and errors are handled adequately. Error conditions are basically in accordance with the API spec, but there are may be a number of incorrect response codes.	The server supports some API functionality as specified. The application basically works, but the robustness is questionable and there are obvious errors and service failures are not handled well. Error conditions are often not in accordance with the API spec	A basic express app with some functioning routes presented, but the API is markedly incomplete or poorly implemented. The application basically works, but there are numerous errors and failures. The API error responses are not to specification.	The server does not meet the specification or is fundamentally flawed or not implemented at all. The application is markedly incomplete and results in numerous errors.

Back-end Application Architecture and Middleware – DB Connectivity, Logging and Security. **(10 marks)**	**The application architecture is professional and uncluttered. The routes are well-organised and handled by appropriate routers. There is good use of middleware for DB connectivity, logging and security. User credentials are handled professionally.**	**The application architecture is reasonably professional and uncluttered. The routes are well-organised and routers are usually organised appropriately. There is good use of middleware for DB connectivity, logging and security, but there may be some variations in the approach. User credentials are handled correctly.**	**The application architecture is good but some aspects are awkward, with some routes not handled especially well, and routers not split appropriately. There is some use of middleware, but this is not as clearly thought out as it needs to be. There may be some inappropriate handling of user credentials.**	**The application architecture is basically sound but the route organisation is very clumsy in parts, and there is too much functionality handled at the application level or in large routers. There is limited use of middleware. Hashing of passwords is poorly implemented.**	**The application architecture is not well chosen, the routes are not well organised and there is little or no use of middleware. Security is poor. Hashing of passwords is not attempted.**	**The application architecture is flawed and there is no use of middleware. Security is deeply flawed or not attempted.**
(30 Marks)	**24**					

Development & Process Code Quality (10 marks) **Demonstrate a high level of familiarity with the JavaScript programming language and a near professional level of skill in its application to problems of significant size and complexity**	**The application shows clear evidence of a professional approach to development, with a coherent modular structure and code quality at a near professional level. For example, program is decomposed into simpler components. Code is clear and simple.**	**The application shows good evidence of a professional approach to development, with a coherent modular structure and code quality at a near professional level, apart from some minor variations.**	**The application shows some evidence of a professional approach to development, with a reasonably coherent modular structure and competent, but not always professional level code quality.**	**The application shows limited evidence of a professional approach to development, with ad hoc or unclear organisation of the application and variable levels of code quality.**	**Application development has been ad hoc and little more than hacking, with no obvious organisation. Code exhibits numerous defects when compared to the standard expected.**	**Application development is deeply flawed, with little structure and poor code quality. For example, horrendously complex code, poorly indented with poor choice of identifiers and no comments.**
Report and User Guide (10 marks)	**The report is thoroughly professional and addresses each of the listed requirements in detail and with only occasional**	**The report is thoroughly professional and addresses each of the listed requirements in detail and with some more**	**The report is professional but lacks some detail in a small number of the listed requirements. There may be**	**The report is adequate, but the coverage is deficient in a number of the listed requirements. Grammar and**	**The report is somewhat adequate, but the coverage is deficient in many of the listed requirements.**	**The report is flawed and doesn't meet the requirements. There may be whole sections missing or poorly covered. There is**

	errors of grammar or structure.	frequent errors in grammar or structure.	occasional errors of grammar or structure.	structure may be somewhat variable, but are overall ok.	Grammar and structure are of variable quality.	no coherent professional report structure as required
Video demo (10 marks)	The video presentation is professional and is able to present all the use cases.	The video presentation is professional and is able to present all the use cases with some small errors.	The video presentation is relatively strong but lacks some rigor and does not present all the use cases.	The video presentation is adequate but does not present all the use cases.	The video presentation is somewhat adequate but does not present the majority of the use cases.	The video presentation is basic and does not present all of the use cases.
(30 Marks)	26					
TOTAL (100 Marks)	87					