A REPORT

ON

WEB DEVELOPMENT OF A TRADING WEB-APPLICATION

BY

Name of the student ID No.

Abhishek Bapna 2018A7PS0184H

Divyansh Pant 2018B3A30856P

Richa Chetan Naik 2018AAPS0267G

Abhishek Prasad 2018A7PS0331P

Jashandeep Singh 2018B5A30621P

Atharva Trivedi 2018A7PS0228G

Divij Nandan Sharma 2018B3A70751P

Aditya Bhandari 2018A7PS0805G

Kaival Parikh 2018A7PS0176P

Tanmay Agrawal 2018B5A30728P

Vooka Ram Kishan 2018A7PS0201G

Atishay Jain 2018A7PS0220G

Archit Jain 2018B3A70792P

Aman 2018A7PS0089G

At

IIFL Mumbai

A Practice School-1 Station of



**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**(June 2020)**

**A REPORT ON**

**WEB-DEVELOPMENT OF A TRADING WEB-APPLICATION**

BY:

|  |  |  |
| --- | --- | --- |
| Name of the student | ID no. | Discipline |
| ABHISHEK BAPNA | 2018A7PS0184H | Computer Science |
| RICHA CHETAN NAIK | 2018AAPS0267G | Electronics and Communication |
| ABHISHEK PRASAD | 2018A7PS0331P | Computer Science |
| JASHANDEEP SINGH | 2018B5A30621P | Electrical and Electronics Engineering + Physics |
| ATHARVA TRIVEDI | 2018A7PS0228G | Computer Science |
| DIVYANSH PANT | 2018B3A30856P | Electrical and Electronics Engineering + Economics |
| DIVIJ NANDAN SHARMA | 2018B3A70751P | Computer Science + Economics |
| ADITYA BHANDARI | 2018A7PS0805G | Computer Science |
| KAIVAL PARIKH | 2018A7PS0176P | Computer Science |
| TANMAY AGRAWAL | 2018B5A30728P | Electrical and Electronics Engineering + Physics |
| VOOKA RAM KISHAN | 2018A7PS0201G | Computer Science |
| ATISHAY JAIN | 2018A7PS0220G | Computer Science |
| ARCHIT JAIN | 2018B3A70792P | Computer Science + Economics |
| AMAN GUPTA | 2018A7PS0089G | Computer Science |

Prepared in partial fulfillment of the Practice School-I Course Nos.

BITS C221/BITS C231/BITS C241

AT

IIFL, Mumbai

A Practice School-I Station of

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**(June 2020)**

**ACKNOWLEDGEMENTS**

We are immensely grateful to Mr. Santosh Jayram, for providing us with this opportunity in this prestigious company and guiding us in all our pursuits.

Next, we are very thankful to our PS-1 instructor Dr. Ramprasad S Joshi, for guiding us through this program and providing us with his invaluable knowledge. His guidance and inputs during this project were immensely helpful.

We are also grateful to our PS-1 industry mentor, Mr. Subash Maji, and the various 5Paisa members we have had insightful interactions with for providing us with his never-ending support and ensuring our project proceeds in an efficient manner.

Lastly, we would like to thank the Practical School 1 Division for providing us this opportunity to work on this project along with the various informative and interesting seminars conducted which motivated us to pursue our projects and thus not only improve our technical skills but also to improve the scope of our teamwork, knowledge, and coordination.

**ABSTRACT**

**Station:** IIFL, Mumbai  **Centre:** Mumbai

**Date of Start:** 18th May 2020  **Date of Submission:** 25th June 2020

**Duration:** 6 weeks

**Title of the Project:** Web Development of A Trading Web-Application

**Names, ID no. and disciplines of members involved:**

|  |  |  |
| --- | --- | --- |
| Name of the student | ID no. | Discipline |
| ABHISHEK BAPNA | 2018A7PS0184H | Computer Science |
| ABHISHEK PRASAD | 2018A7PS0331P | Computer Science |
| JASHANDEEP SINGH | 2018B5A30621P | Electrical and Electronics Engineering + Physics |
| DIVYANSH PANT | 2018B3A30856P | Electrical and Electronics Engineering + Economics |
| DIVIJ NANDAN SHARMA | 2018B3A70751P | Computer Science + Economics |
| ADITYA BHANDARI | 2018A7PS0805G | Computer Science |
| ATHARVA TRIVEDI | 2018A7PS0228G | Computer Science |
| RICHA CHETAN NAIK | 2018AAPS0267G | Electronics and Communication |
| KAIVAL PARIKH | 2018A7PS0176P | Computer Science |
| TANMAY AGRAWAL | 2018B5A30728P | Electrical and Electronics Engineering + Physics |
| VOOKA RAM KISHAN | 2018A7PS0201G | Computer Science |
| ATISHAY JAIN | 2018A7PS0220G | Computer Science |
| ARCHIT JAIN | 2018B3A70792P | Computer Science + Economics |
| AMAN GUPTA | 2018A7PS0089G | Computer Science |

**Name and designation of the expert:** Mr. Santosh Jayram, Mr. Subhash Maji

**Names of the PS Faculty:** Dr. Ramaprasad Joshi

**Key Words:** Web Development, API, Javascript, HTML, CSS,

Server-requests, React.js, Node.js, Git, Github

**Project Areas:** Web Development, Stock Market Trading, API Handling

**Abstract:** This report is a summary of the methods and approaches used to develop and implement a web-application that serves as a model stock trading application using the API’s provided by 5Paisa and evaluates the functionality of using APIs as an effective application development tool.

This report describes the front-end and back-end development techniques used towards achieving this goal.

**LIST OF ILLUSTRATIONS & LINKS**

**Fig**:

1. Login Page
2. Navigation drawer code
3. Navigation bar UI
4. AppBar UI
5. AppBar code
6. WatchList class (code)
7. WatchList UI
8. WatchListComp code
9. WatchList UI depiction
10. Simple React Component
11. Stateful Component in React
12. Material-ui/core/Drawer example
13. Material-ui/core/AppBar example
14. Material-ui/core/Toolbar example
15. Material-ui/icons example
16. Transaction History UI

**Src**:

1. Sample request body (JSON)

**Links:**

[**https://github.com/Abhishek2022/5paisa-clone-react**](https://github.com/Abhishek2022/5paisa-clone-react)

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| Acknowledgment | 3 |
| Abstract | 4 |
| List of Illustrations & Links | 6 |
| 1. Introduction | 8 |
| 1. Front-end Development 2. Login 3. Dashboard 4. Transaction History | 10  10  11  21 |
| 1. Back-end Development    1. Encryption Algorithm    2. Login    3. Report Modules | 22  22  23  24 |
| 1. Conclusion | 27 |
| 1. Recommendations | 28 |
| 1. References | 29 |
| 1. Glossary | 30 |
|  |  |
|  |  |
|  |  |

**INTRODUCTION**

5Paisa ltd. is a discount [broking](https://en.wikipedia.org/wiki/Discount_broker) company based in [India](https://en.wikipedia.org/wiki/India) and was originally incorporated in July 2010. 5paisa is a member of [NSE](https://en.wikipedia.org/wiki/National_Stock_Exchange_of_India), [BSE](https://en.wikipedia.org/wiki/Bombay_Stock_Exchange), [MCX](https://en.wikipedia.org/wiki/Metropolitan_Stock_Exchange), [MCX-SX](https://en.wikipedia.org/wiki/MCX-SX) that provides [retail broking](https://en.wikipedia.org/wiki/Retail_brokerage), currencies and [commodities trading](https://en.wikipedia.org/wiki/Commodities_trading), [mutual funds](https://en.wikipedia.org/wiki/Mutual_funds), [IPO](https://en.wikipedia.org/wiki/IPO), and [bond](https://en.wikipedia.org/wiki/Bond_(finance))s. 5paisa began operations in 2016 as a subsidiary of [IIFL Holdings Ltd](https://en.wikipedia.org/wiki/India_Infoline).

The company's headquarters is located and has headquarters in [Mumbai](https://en.wikipedia.org/wiki/Mumbai). 5paisa is listed on the [National Stock Exchange](https://en.wikipedia.org/wiki/National_Stock_Exchange_of_India) and the [Bombay Stock Exchange](https://en.wikipedia.org/wiki/Bombay_Stock_Exchange) of India.

5paisa allows trading facility in equity, derivative, commodities, and currency segments and provides services regarding [mutual fund](https://en.wikipedia.org/wiki/Mutual_fund), [personal loan](https://en.wikipedia.org/wiki/Personal_loan) and insurance.It recently rolled out Options X, brokerage plan for options trading, enabling investors and traders to capitalize on market opportunities.

Our project is an endeavor to develop and deploy a web-application of a model trading app using the openly available 5Paisa API’s and to evaluate the feasibility of using APIs as an adequate development platform and tool.

This is an attempt to recreate the functionalities provided by the 5Paisa app and in turn, help the technological division of 5Paisa to solve similar problems future users might face and to understand the current needs and demands and also how to make algorithmic trading as a whole more approachable so they can modify them and increase their efficacy. We do this by building a web-application using the openly available APIs and free technological skill trees. (Eg. React, HTML, CSS, Node.js, Postman, Git, and Github)

This task of web-application creation has been divided into multiple separate modules handled by subdivisions within our group focusing on both the front-end development and back-end development of the module so that each of us gets a chance to explore all aspects of the project.

We used HTML, CSS, Javascript & React for front-end development. Technologies such as Node.js for sending requests and fetching responses from the APIs and Postman for testing the APIs were used for back-end development.

In this report, we first discuss the website we have envisioned and the features we have developed. Then, we shall discuss the progress we have made in both front-end and back-end aspects. We will conclude with a basic overview of all the features we’ve implemented and wished to incorporate. We will also be dwelling on the various obstacles we encountered and how we crossed them.

The source code for the entire project can be found on : <https://github.com/Abhishek2022/5paisa-clone-react>

Note: We have provided links, illustrations to the source code in certain places to make the report as compact as possible.

**FRONT- END DEVELOPMENT**

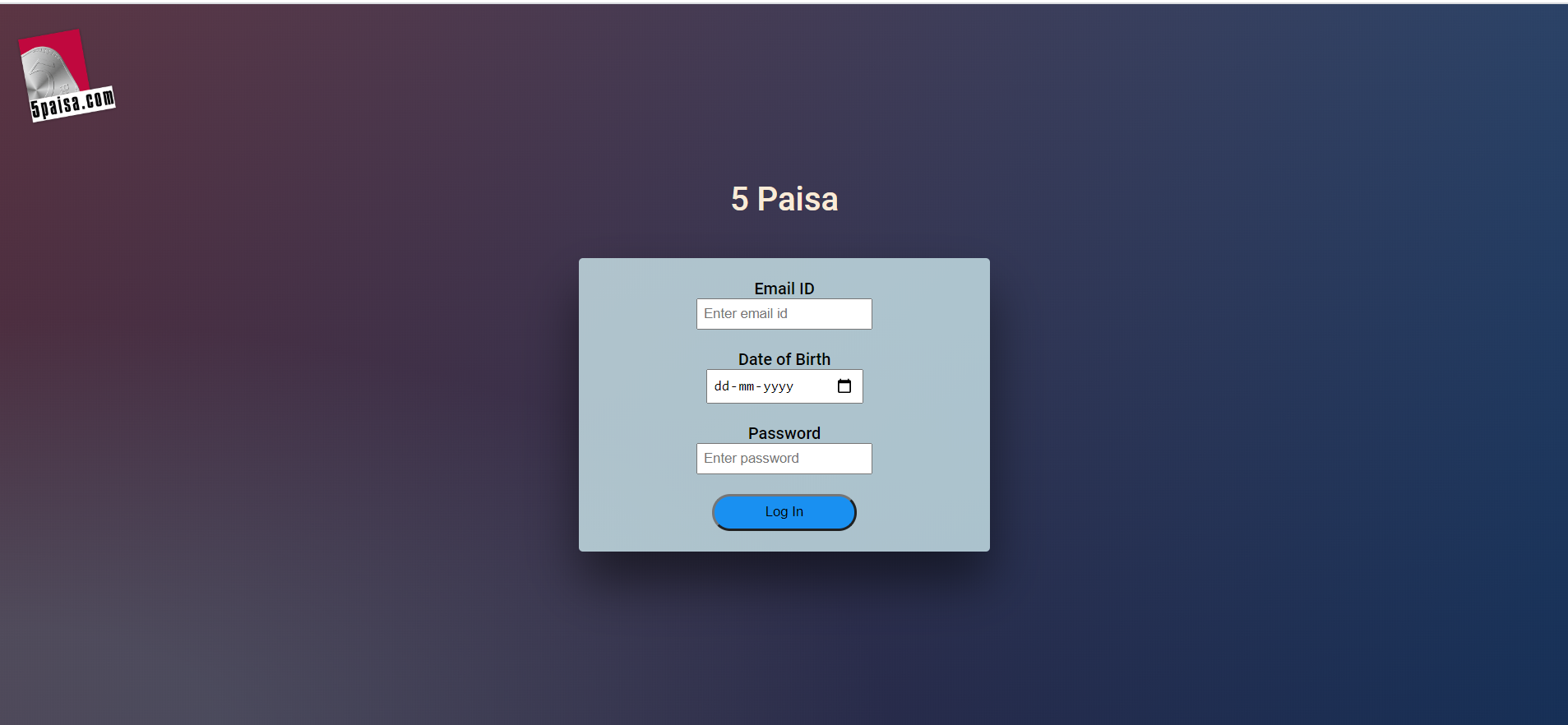
Front-end functionality and design are equally if not less important to any web-application project as it is the component the end-user interacts with and is what attracts users in the first place.

A good front-end should be easy to navigate, clutter-free and should render without any glitches. In this section, we will be discussing the various front-end components created and their functionalities in their respective modules.

The source code for all the components mentioned below can be found in the link mentioned in the Introduction section.

1. **Login Page:**

The login page takes the user’s email ID, date of birth, and password as input fields. It is clutter-free as per best practices for login pages and possesses the 5paisa logo at the top left corner of the page.

****fig.1

1. **Dashboard-page:**

**i) NavBar/Navigation Drawer:**

Navigation drawers provide access to destinations in your app. Side sheets are surfaces containing supplementary content that are present to the left or right of the screen. Our app has a clipped under the app bar type of drawer with a black background image in it.

Drawer is imported from material-ui having customized height and width. Home is an option on the navbar which enables us to visit the dashboard of the app. It has an appropriate icon beside it.



fig.2

Order is another option of the navbar which has a dropdown list which consists of Order History and Order Status which help the user to keep track of the stocks he/she has purchased. It also has an appropriate icon attached with it.

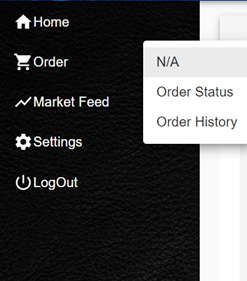


fig.3

Market Feed, Settings and LogOut are the various other options present on the navbar. Each of them has an icon along with them.

**ii) AppBar:**

The AppBar provides content and actions related to the current screen. It’s used for branding, screen titles, navigation, and actions. It can transform into a contextual action bar or be used as a navbar.



fig.4

The AppBar of our app contains 5 Paisa’s logo at the left corner with the title of 5 paisa in the middle along with a search option on the right corner.

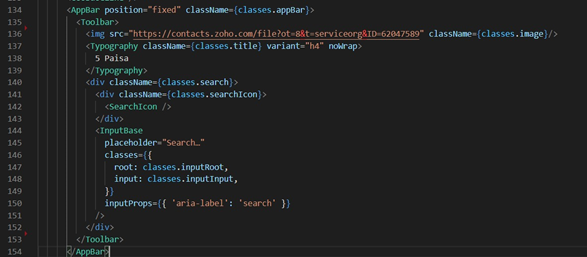


fig.5

AppBar, Typography, SearchIcon, etc. are all imports of material-ui , which is a popular framework of React

**iii) Watchlist:**

In a trading app watchlist is a list of stocks or securities being monitored for potential investment opportunities. An item in watchlist contains the following information in general,

* Name of the stock
* Percentage change of its price with respect to the previous day’s closing price.
* The current price of the stock.

We have used two JavaScript files i.e., WatchList.js and WatchListComp.js which export two react components <WatchList/> and <WatchListComp/> to the dashboard page.

**WatchList.js**



fig.6

Here we create a react class named WatchList which extends React.Component. It iterates over the stock array which contains information about each stock and passes it to the <WatchListComp/> and renders these components inside a <div> tag.

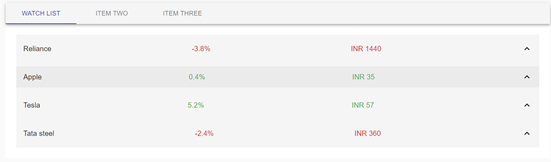


fig.7

**WatchListComp.js**

In this file, we create the <WatchListComp/> which takes in the data from its parent component i.e., <WatchList/> and renders it to the page. For rendering, we use a custom-built <List/> component from material-ui/core. “Material-ui” is a react UI framework that provides us with pre-built react components. The following is the image of JSX returned from WatchListComp.js .

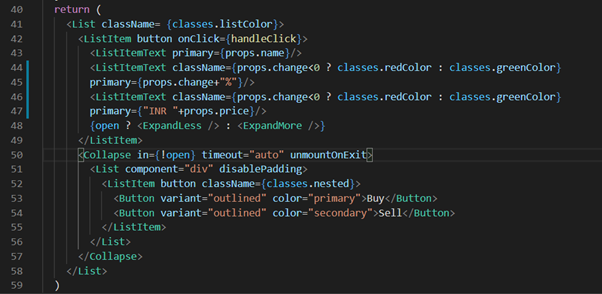


fig.8

It renders the data in green or red color depending on whether the change in price is positive or negative. Each list item has a drop-down, upon clicking gives the option to buy or sell that particular stock.



fig.9

**Populating Watchlist**

Watchlist can be populated by parsing the JSON data received from API calls and storing it in a JavaScript array, which later is imported into WatchList.js .

**Technologies used in Dashboard-page:**

**React- A JavaScript Library**

# **Introduction:**

React is a JavaScript library created by Facebook. It is a User Interface (UI) library and is used to build UI components. React makes it painless to create interactive UIs. Design simple views for each state in the application, and efficiently update and render just the right components when the data changes.

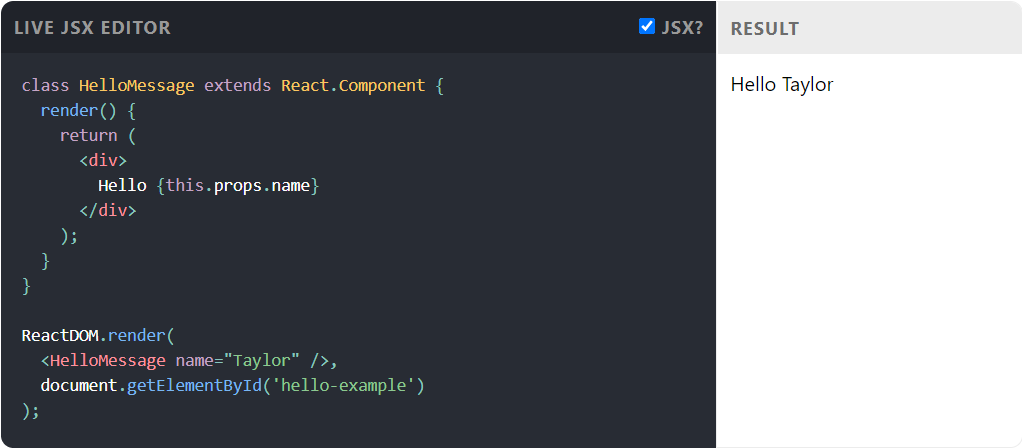
React can also render on the server using Node and power mobile apps using React Native.

React is component-based, therefore we can build encapsulated components that manage their own state, then compose them to make complex UIs.

Below are a few snippets to show how React is used.

# **A Simple Component:**

React components implement a render() method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by render() via this.props.

fig.10

# **A Stateful Component:**

In addition to taking input data, a component can maintain interval state data (accessed via this.state). When a component’s state data changes, the rendered markup will be updated by re-invoking render().

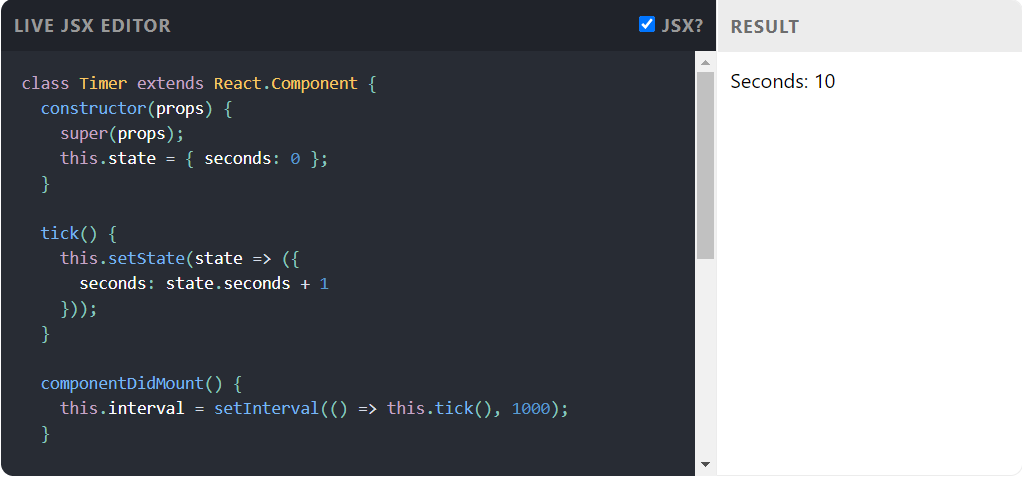


fig.11

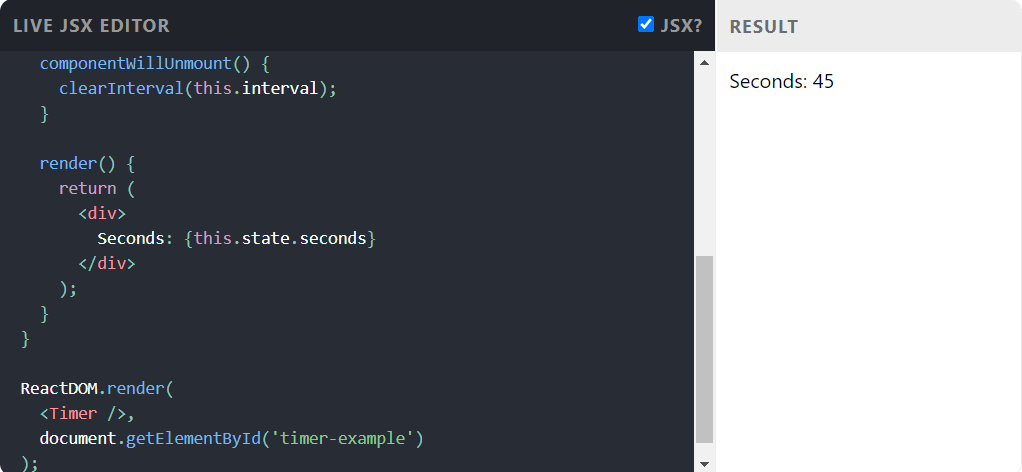


fig.11

**Material-UI**

# **Introduction:**

In a nutshell, Material-UI is an open-source project that features React components that implement Google's Material Design. ... With over 35,000 stars on GitHub, Material-UI is one of the top user interface libraries for React out there.

Material-UI components work in isolation. They are self-supporting, and will only inject the styles they need to display. They don't rely on any global style-sheets such asnormalize.css.

All the Material-UI component we used will be defined below:

**Material-ui/core/styles:**

Material-UI's styling solution is inspired by many other styling libraries such as [styled-components](https://www.styled-components.com/) andemotion.

* 💅 You can expect [the same advantages](https://www.styled-components.com/docs/basics#motivation) as styled-components.
* 🚀 It's [blazing fast](https://github.com/mui-org/material-ui/blob/master/packages/material-ui-benchmark/README.md#material-uistyles).
* 🧩 It's extensible via a [plugin](https://github.com/cssinjs/jss/blob/master/docs/plugins.md) API.
* ⚡️ It uses [JSS](https://github.com/cssinjs/jss) at its core – a [high performance](https://github.com/cssinjs/jss/blob/master/docs/performance.md) JavaScript to CSS compiler which works at runtime and server-side.
* 📦 Less than [15 KB gzipped](https://bundlephobia.com/result?p=@material-ui/styles); and no bundle size increase if used alongside Material-UI.

**Material-ui/core/Drawer:**

Navigation drawers provide access to destinations in your app. Side sheets are surfaces containing supplementary content that are anchored to the left or right edge of the screen.

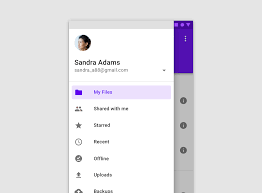


fig.12

**Material-ui/core/AppBar:**

The top app bar provides content and actions related to the current screen. It’s used for branding, screen titles, navigation, and actions.

It can transform into a contextual action bar.

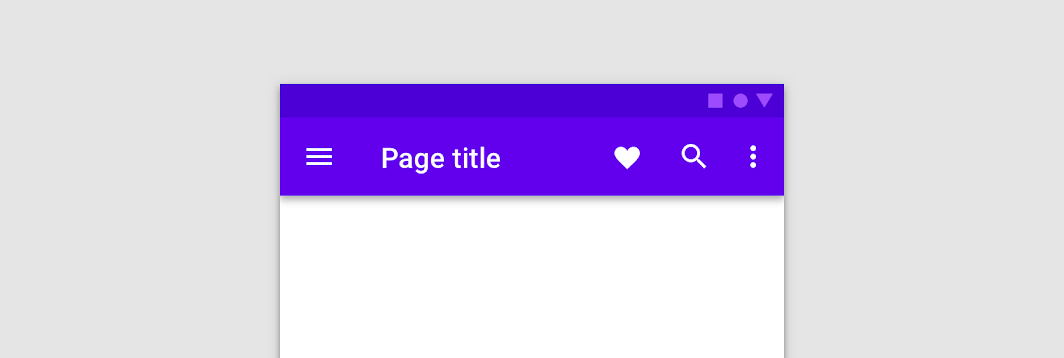


fig.13

**Material-ui/core/CssBaseline:**

Material-UI provides a CssBaseline component to kickstart an elegant, consistent, and simple baseline to build upon.

## **Approach**

### **Page**

The <html> and <body> elements are updated to provide better page-wide defaults. More specifically:

* The margin in all browsers is removed.
* The default Material Design background color is applied. It's using [theme.palette.background.default](https://material-ui.com/customization/default-theme/?expand-path=$.palette.background) for standard devices and a white background for print devices.

### **Layout**

* box-sizing is set globally on the <html> element to border-box. Every element—including \*::before and \*::after are declared to inherit this property, which ensures that the declared width of the element is never exceeded due to padding or border.

### **Typography**

* No base font-size is declared on the <html>, but 16px is assumed (the browser default). You can learn more about the implications of changing the <html> default font size in [the theme documentation](https://material-ui.com/customization/typography/#typography-html-font-size) page.
* Set the theme.typography.body style on the <body> element.
* Set the font-weight to theme.typography.fontWeightBold for the <b> and <strong> elements.
* Custom font-smoothing is enabled for better display of the Roboto font.

**Material-ui/core/Toolbar:**

Toolbar is to display its children with an inline display (elements are placed next to each other), something Appbar doesn't do.

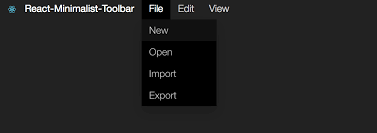
****

fig.14

**Material-ui/core/List:**

[Lists](https://material.io/design/components/lists.html) are a continuous group of text or images. They are composed of items containing primary and supplemental actions, which are represented by icons and text.

**Types of list**

* [Simple List](https://material-ui.com/components/lists/#simple-list)
* [Nested List](https://material-ui.com/components/lists/#nested-list)
* [Folder Lis](https://material-ui.com/components/lists/#folder-list)t
* [Pinned Subheader List](https://material-ui.com/components/lists/#pinned-subheader-list)
* [Inset List](https://material-ui.com/components/lists/#inset-list)
* [Virtualized List](https://material-ui.com/components/lists/#virtualized-list)

**Controls for the various list:**

* [List Controls](https://material-ui.com/components/lists/#list-controls)
  + [Checkbox](https://material-ui.com/components/lists/#checkbox)
  + [Switch](https://material-ui.com/components/lists/#switch)

**You can add text to list by using :**

import ListItemText from '@material-ui/core/ListItemText'

**You can add icons to list by using :**

import ListItemIcon from '@material-ui/core/ListItemIcon'

**Material-ui/icons:**

Material-UI provides icons support in three ways:

1. Standardized [Material Design icons](https://material-ui.com/components/icons/#material-icons) exported as React components (SVG icons).
2. With the [SvgIcon](https://material-ui.com/components/icons/#svgicon) component, a React wrapper for custom SVG icons.
3. With the [Icon](https://material-ui.com/components/icons/#icon-font-icons) component, a React wrapper for custom font icons.

Each icon also has a "theme": Filled (default), Outlined, Rounded, Two tone and Sharp. If you want to import the icon component with a theme other than the default, append the theme name to the icon name. For example @material-ui/icons/Delete icon with:

* Filled theme (default) is exported as @material-ui/icons/Delete,
* Outlined theme is exported as @material-ui/icons/DeleteOutlined,
* Rounded theme is exported as @material-ui/icons/DeleteRounded,
* Twotone theme is exported as @material-ui/icons/DeleteTwoTone,
* Sharp theme is exported as @material-ui/icons/DeleteSharp.

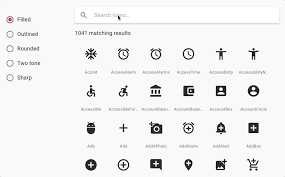


fig.15

1. **Transaction History:**

The Transaction History (OrderBook) part of the dashboard basically is used to fetch the user’s order details based on the date.

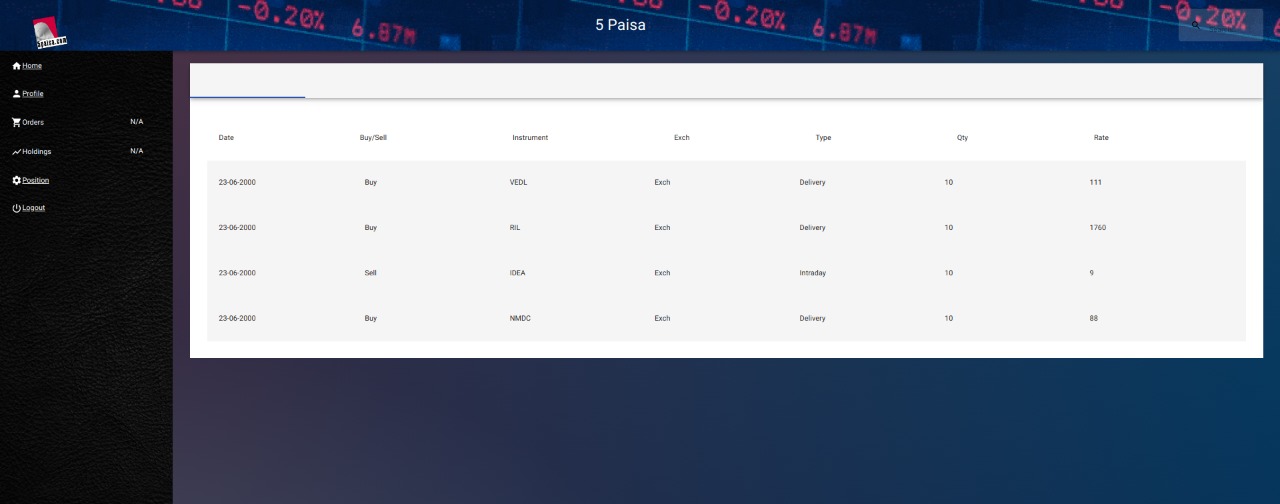


fig.16

**BACK-END DEVELOPMENT**

Back-end functionality and design are the core of a web-application project as it forms the crux of the project as a whole.

A good back-end should be closed from any external unwanted interference and clear in flow of movements. In this section, we will be discussing the various back-end components created and their functionalities in their respective modules.

Framework used for back-end development of this project is Node.js. Node.js is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside a web browser. It has an event-driven architecture capable of asynchronous I/O. It is built on Chrome's V8 JavaScript engine. Node Package Manager (npm) is a package manager for the JavaScript programming language.It is the default package manager for the JavaScript runtime environment Node.js. It allows users to consume and distribute JavaScript modules that are available on the registry. Various node modules were used while building the back-end.

The source code for everything mentioned below can be found in the link mentioned in the Introduction section.

1. **Encryption Algorithm**

5paisa uses 256-bit Advanced Encryption Standard (AES) algorithm, here 256-bit implies the length of the key used in encryption. It is the maximum key length that can be used and provides the maximum security margin. Variables needed by the encryption function are the string which needs to be encrypted and the encryption key which is provided to the developer at the time of registration.

The node library required for this function is [CryptoJs](https://www.npmjs.com/package/crypto-js), which is JavaScript implementations of standard and secure cryptographic algorithms. In this library, the cipher function AES takes string, key and {iv} as parameters and returns the encrypted string.

encrypted\_string = CryptoJS.AES.encrypt(*string*, key, { iv })

1. string: String text that needs to be encrypted.
2. key: Key is obtained from below mentioned code snippet.

key = new *CryptoJS*.*lib*.*WordArray*.*init*(keyBytes.words.splice(4), 32)

A WordArray object represents an array of 32-bit words. .lib.WordArray.init converts an array to a WordArray. Here keyBytes is a stronger encryption key made with the use of an older encryption key and some randomized function. For this purpose PBKDF2 function is used. PBKDF2 is a password-based key derivation function present in the CryptoJs library. Its input parameters are encryption\_key (provided to the developer at the time of registration), salt, and keySize with iterations. Salt is random data that is used as an additional input to a one-way function that hashes data, a password, or the encryption\_key, in our case. A salt provides a large set of keys for any given password, and an iteration count increases the cost of producing keys from a password, thereby also increasing the difficulty of attack. Human memorizable encryption\_key are known to be insecure. In order to make them more secure, we add a bunch of random bits (the salt) so that the actual key = function(Salt, encryption\_key). An effect of Salt is that the same encryption\_key doesn’t always produce the same key. iv (initialization vector) is used, similarly, to ensure the same thing. It appears that Salt is used with encryption\_key to generate a stronger key for encryption, then the resulting encryption is processed with iv. So to speak, the encrypted\_string = function(string, encryption\_key, salt, iv). The specific salt array that is to be used is provided by 5paisa which is given below in the code snippet.

keyBytes = CryptoJS.PBKDF2(

encryption\_key,

new *CryptoJS*.*lib*.*WordArray*.*init*(byteArrayToWordArray([83, 71, 26, 58, 54, 35, 22, 11, 83, 71, 26, 58, 54, 35, 22, 11]).words, 16),

{

keySize: 12,

iterations: 1000

}

)

1. iv : iv stands for the initialization vector. It is obtained the same way as the key. Except for the first 4 array members of the keyBytes array are spliced away in the key.

iv = new *CryptoJS*.*lib*.*WordArray*.*init*(keyBytes.words, 16)

**b) Login**

/login - Encrypts user’s credentials and fetches user details and cookies for authentication. Sample JSON format request body for login API is as follows

{

"head": {

"appName": "5P55155874",

"appVer": "1.0",

"key": "<user key>",

"osName": "WEB",

"requestCode": "5PLoginV2",

"userId": "CirMuUW85Fi",

"password": "3wREYKyYoOX"

},

"body": {

"Email\_id": "7ntChiybFM1KOXFNnSirrbtwtnnMJNepgKQQPl7VxXQ=",

"Password": "ayU9r8oa2keFJOXiLZ7txg==",

"LocalIP": "171.76.155.206",

"PublicIP": "171.76.155.206",

"HDSerailNumber": "ST1000LM035-1RK172\_ZDE9FTTG",

"MACAddress": "54:bf:64:18:a5:95",

"MachineID": "/6HSGFP2/CNCMC0085S001B/",

"VersionNo": "1.7",

"RequestNo": "1",

"My2PIN": "ls/KaZiYlDe1biLkNTLJqA==",

"ConnectionType": "1"

}

}

The fields in the head object are the credentials given to the developer during registration for 5paisa.com. Email\_id, Password and My2PIN are fetched from the user. Every other field is fetched from a default configuration file. My2PIN is the date of birth of the user in YYYYMMDD format. All of the above three mentioned fields first need to be encrypted and then sent to the API request. A valid login request gets a response with the Client Code of the user and cookies for session-based authentication. Both of these data are crucial for making requests for further modules. Hence they are stored.

**c) Report Modules**

These modules are used to report to the user’s profile in the app. These include the following modules -

1. /orderBook - Used to fetch the user’s order details for the particular day.
2. /holdings - Used to fetch user’s holdings as of the beginning of the day.
3. /positions - Used to fetch the user’s net position details.
4. /margin - Used to fetch the total fund a user has to trade, which includes both of what they've added themselves and borrowed from the broker.

Requests for all the modules are similar except for the requestCode field which is different for different modules. The sample request body in JSON format is as follows:

|  |
| --- |
| {  "head": {  "appName": "APPTRIAL",  "appVer": "1.0",  "key": "ue73jH6AKVXeESDFSGzBswG5ss9ugyrAyKYxg",  "osName": "Android",  "requestCode": "<request code>",  "userId": "s9RzKCjdL9SDAN",  "password": "nznS4f34zASDAXw"  },  "body": {  "ClientCode": "12345678"  }  } |

src.1

Cookies fetched from the login response headers are needed to be sent via request headers for session-based authentication.

Node library [axios](https://www.npmjs.com/package/axios) was used to make HTTP requests from the client-side. Axios is a promise-based HTTP client for the browser and node.js. It makes asynchronous HTTP requests possible.

[Express](https://expressjs.com/) is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some of the core features of Express framework that we used to build the back-end for the app −

1. Allows to set up middlewares to respond to HTTP Requests.

1. Defines a routing table that is used to perform different actions based on HTTP Method and URL. All the different modules are routed by their respective HTTP method. A sample routing of /login page is as follows.

app.post('', async(req, res) => {

let { email, password, date } = req.body

date = date.replace(/-/g, '')

const result = await authenticate(email, password, date)

if (result) {

res.render('dashboard')

} else {

res.render("login", { message: "Invalid Credentials" })

}

})

Here POST HTTP method is used to send the login credentials sent by the user. Which are then sent to authenticate function asynchronously. Result is a boolean which will decide if the dashboard will be rendered or “invalid credentials” be printed. This feature was also used to connect the front-end React.js files with the back-end node.js file.

1. Allows to dynamically render HTML Pages based on passing arguments to templates.

**CONCLUSION**

The completed features are functioning as expected. The overall functioning of the application possesses the following features:

1. A **login** page
2. The app’s **dashboard**

The application dashboard has an **AppBar** at the top, which provides content and actions related to the current screen. It is used for branding, screen titles, navigation and actions.

Further, the Navigation bar (**NavBar**) on the left of side of the dashboard has many important options such as:

1. Home
2. Profile
3. Orders- order history, order status
4. Holdings- funds, holdings
5. Position
6. Logout

Lastly, two more crucial features of the dashboard of the application created are:

**Watchlist**, which is a list of stocks and securities which are being monitored as potential investment opportunities.

**Transaction history,** which is used to fetch the user’s order details for the particular day.

**RECOMMENDATIONS**

1. **API Documentation**

The API documentation is not thorough enough for a developer to make a proper, well functioning app without any assistance from 5Paisa. There were some discrepancies observed in the documentation which were reported right away to the 5Paisa team. We recommend the 5Paisa team to review their API documentation.

2. **Support & Updates:**

To make API’s more lucrative and open to even the not so technically motivated we recommend a dedicated API support mechanism for all future users which could be incorporated in 5Paisa’s various subscription plans along with essential videos,tools to aid in its development.

3. **Resources**

Building a proper functioning app requires the kind of skill-set that most of the sophomore students do not possess. This was the case with our team as well as no skill set prerequisites were mentioned in the PS-1 preference list page of the station. As a result of that most of our time was spent searching for credible resources and learning skills from them in an improper order. We recommend the experienced mentors to provide the upcoming batches with some roadmap as to how a student should proceed efficiently with their learning curve and with credible resources.

**REFERENCES**

* [HTML Documentation](https://devdocs.io/html/)
* [CSS Documentation](https://developer.mozilla.org/en-US/docs/Web/CSS/Reference)
* [JavaScript Documentation](https://developer.mozilla.org/en-US/docs/Web/JavaScript)
* [Postman Documentation](https://learning.postman.com/docs/postman/launching-postman/introduction/)
* [React Documentation](https://reactjs.org/docs/getting-started.html)
* [Node.js Documentation](https://nodejs.org/en/docs/)
* [Github](https://github.com/)
* [5Paisa API’s](https://www.5paisa.com/developerapi)
* [Express Documentation](https://expressjs.com/en/api.html)

**GLOSSARY**

* **CSS:** Cascading Style Sheet
* **GitHub**: A Git repository hosting service
* **HTML:** Hyper Text Markup Language
* **JavaScript:** A high-level programming language used to make webpages interactive
* **Postman:** Tool to get and receive server requests
* **React:** JavaScript library mainly used for front end using components,states and props
* **Node.js:**An open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside a web browser.
* **5Paisa API’s:** Openly available API’s for development
* **Express:** Express, is a web application framework for Node.js