Final Report
(Portfolio Management System)

Course Code: CS254 **Course Title:** DBMS Lab

Semester: B. Tech 4^{th} Sem **Section:** S2

Academic Year: 2020-21 **Course Instructor:**Dr. Annappa B and Mr. Sharath Yaji

Team Members:

1. Adithya Rajesh, 191CS203, 6380666274, adithyarajesh.191cs203@nitk.edu.in

2. Chaithanya Shyam, 191CS218, 9480042929, chaithanyashyam.191cs218@nitk.edu.in

3. K Dharmick Sai, 191CS221, 8904222008, kdharmicksai.191cs221@nitk.edu.in

1 Abstract

The project aims to build a web application that eases and automates the task of monitoring one's investments. It maintains a record of the list of stocks that have been bought and sold by the user and other details such as the number of units of stock and the price of selling/buying at that time. By maintaining a history of all transactions the user at any time can get a comprehensive overview of the various investments and the amount spent and profit/loss generated from each transaction.

We maintain an admin user who can adjust the prices of all the stocks in the exchange. These changes will be reflected in each user's portfolio and the current prices of the stocks they have bought will change according to this. There will be a global table maintained by the admin user that lists all the stocks and their current prices. The users can only view this table and search for any stock they are interested in. The app provides a simple user friendly functionality to buy/sell the mentioned number of units of stock of a particular corporation. To ease the life of an investor we maintain various statistics to indicate the different sectors in which the investor has put money in and it will also help to maintain a diverse portfolio.

2 Introduction

Key Features

In our implementation of the portfolio management system we have implemented the following features :

1.Login and User authentication - Basic Login facility that allows registered users to log in to their account

2.Profile Dashboard - Displays the current portfolio of the user

3.Buy - Allows User to buy the mentioned units of shares of some company

4.Sell - Allows User to sell the mentioned units of shares of some company

5.Quote - Find the current price of a stock and also view its previous history of price changes

6.History - User can view their previous transactions

7.Watch list - User can add stocks to their watch list that they are potentially interested in to track prices

8.Discussion - Forum where users can discuss the about the current trends in the stock market

9.Admin Panel - Provided different functionality for the admin user that allows to change the price of the stocks and add new ones to the market. The changes in price made by the admin will be reflected in the users portfolio

The above mentioned features provide the users with a comprehensive platform to keep track of their stock trading activities in an efficient manner

The backend used is NodeJS and Express. For frontend templates we have used embedded javascript(EJS). The database used is MySQL. All the code will be there on our github repository link will be provided below.

All our queries have been done from scratch in MySQL without using any ORM. The database definition file in the repository has been used by all team members to ensure consistency of schema.

3 ER Diagram

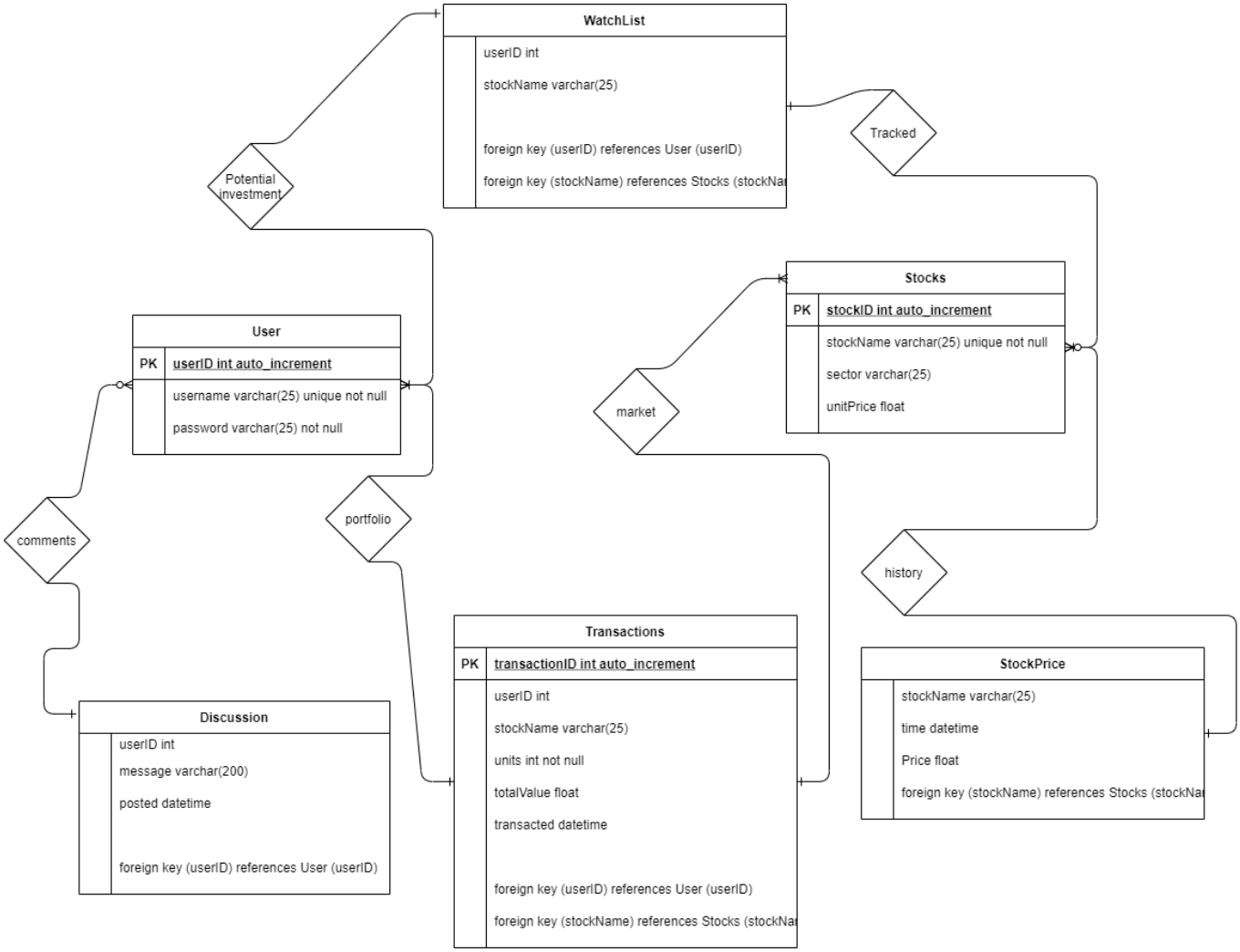


Figure 1: ER diagram of Portfolio Management Database

4 Source Code

Since the source code is very big we have included only some important snippets. The rest of the code will be there in the github repository
Link to github repository:source-code

4.1 Backend

Code for login, register also is similar

```
app.get('/login', (req, res) => {
    res.render('login', {message: "", username: ""})
});


app.post('/login', (req, res) => {
    var query = "select userID, password from User where
    username = \"" + req.body.username + "\";";
    dbms.query(query, (err, result, fields) => {
        if(err) throw err;

        if(result.length == 0) {
            res.render('login', {message: "Invalid Username",
            username: req.body.username})
        }
        else if(req.body.password != result[0].password) {
            res.render('login', {message: "Wrong Password",
            username: req.body.username});
        }
        else {
            activeUsers[result[0].userID] =
            req.socket.remoteAddress;
            var travel;
            if(result[0].userID == 1) {
```

```
                travel = '/control';
            }
            else {
                travel = '/profile/' + result[0].userID;
            }
            res.redirect(travel);
        }
    });
})
```

Code for dashboard, buy, sell, quote and history are similar

```
app.get('/profile/:id', (req, res) => {
    //console.log(req.params.id);
    var key = parseInt(req.params.id);
    if(activeUsers[key] == undefined) {
        return res.redirect('/login');
    }

    //  VERY VERY IMPORTANT QUERY
    var findID = "with UserStocks(userID,units,stockname) as (
    SELECT userID, units,stockname from Transactions natural
    join User having userID ="+key+") select sum(units)as
    num_stocks , sum(units)*unitprice as stocksworth, stockname,
    sector from UserStocks natural join Stocks  group by
    stockname ;";
    var username ;
    var find_user = "SELECT username from User WHERE userID = "
    + key + ";"
    dbms.query(find_user , (err,result,feilds)=>{
        if(err) throw err;
        username = result[0].username;
    });
```

```
    dbms.query(findID, (err, result, fields) => {
        if(err) throw err;
        // console.log(result[0].stockname);

        res.render('profile', {username: username,shares:result,
        key: key});
    });
});
```

4.2 Frontend

buy templat, the scroll search used is similar for others

```
<body>
    <center>
        <h1><%= username %> start buying stocks !!!</h1>
        <% if (statusMessage !=="") {%>
          <h1><%= statusMessage %></h1>
        <%}%>
        <form action="/buy/<%= userID %>" method="POST">
          <div class = "form-group">
            <label>Name: </label>
            <select name = "chosenStockID">
                <% for(var i = 0; i < stocks.length; i++) { %>
                <option value="<%= stocks[i].stockID %>"> <%=
                stocks[i].stockName %> </option>
                <% } %>
                <input autocomplete="off" autofocus
                class="form-control" name="sharesBought"
                placeholder="# of Shares" type="number"
                min="1"/>
                <button class="btn btn-default"
                type="submit">Buy</button>
          </div>
```

```
        </form>
    </center>
</body>
```

navbar similar for all templates

```
<head>
<nav class="navbar navbar-inverse">
<div class="container-fluid">
<div class="navbar-header">
<a class="navbar-brand">Portfolio Manager</a>
</div>
<ul class="nav navbar-nav">
<li id="profile"><a href="/profile/<%= userID %>">Dashboard</a></li>
<li id="buy" ><a href="/buy/<%= userID %>">Buy</a></li>
<li id="sell" ><a href="/sell/<%= userID %>">Sell</a></li>
<li id="quote" ><a href="/quote/<%= userID %>">Quote</a></li>
<li id="history" ><a href="/history/<%= userID %>">History</a></li>
<li id="watchlist" ><a href="/watchlist/<%= userID %>">Watchlist</a></li>
<li id="discussion" ><a href="/discussion/<%= userID %>">Discussion</a></li>
<li id="logout"><a href ="/logout/<%= userID %>">Logout</a></li>
</ul>
</div>
</nav>
</head>
```

5 Results

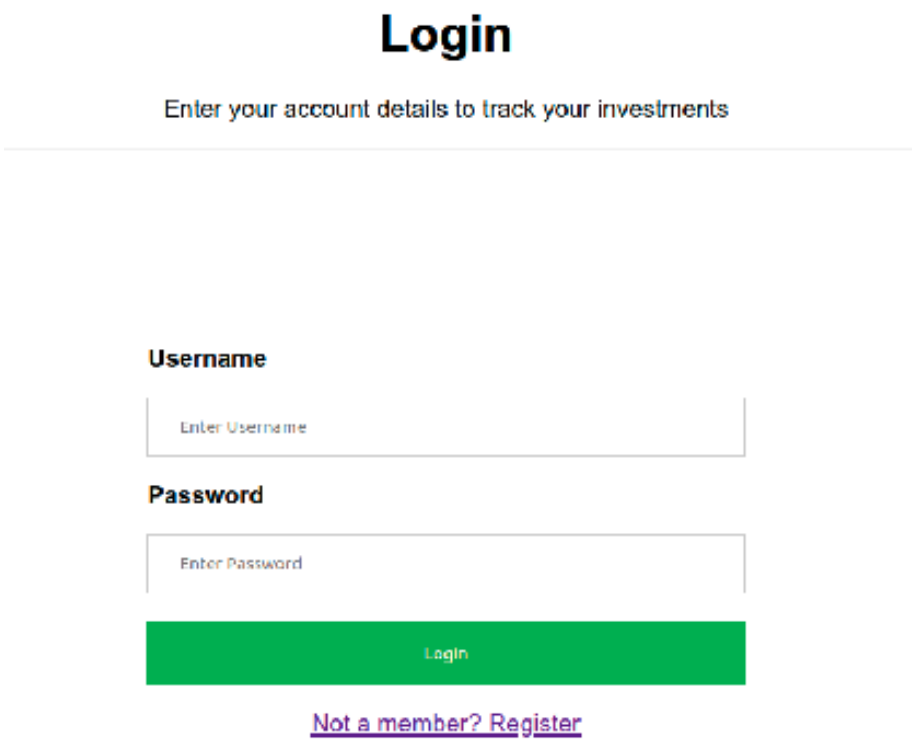
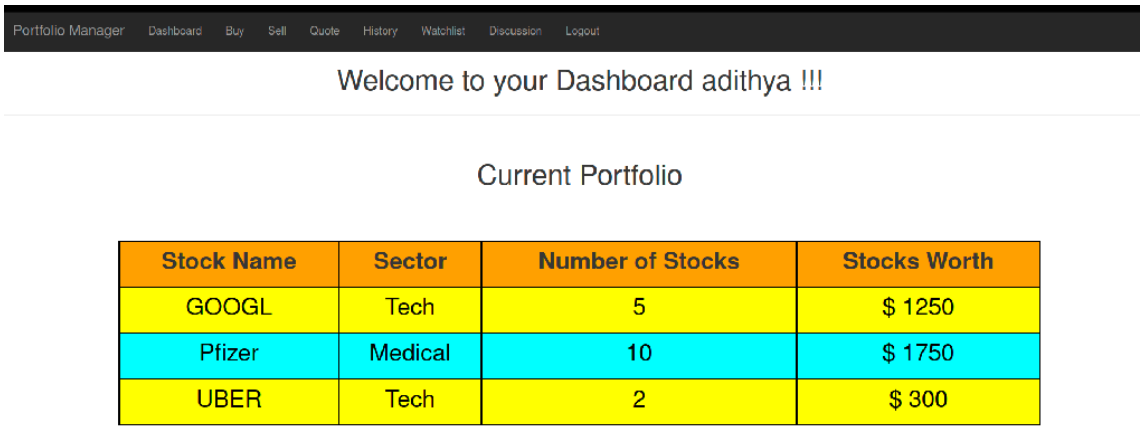


Figure 2: Login Facility



Stock Name	Sector	Number of Stocks	Stocks Worth
GOOGL	Tech	5	$ 1250
Pfizer	Medical	10	$ 1750
UBER	Tech	2	$ 300

Figure 3: User Portfolio

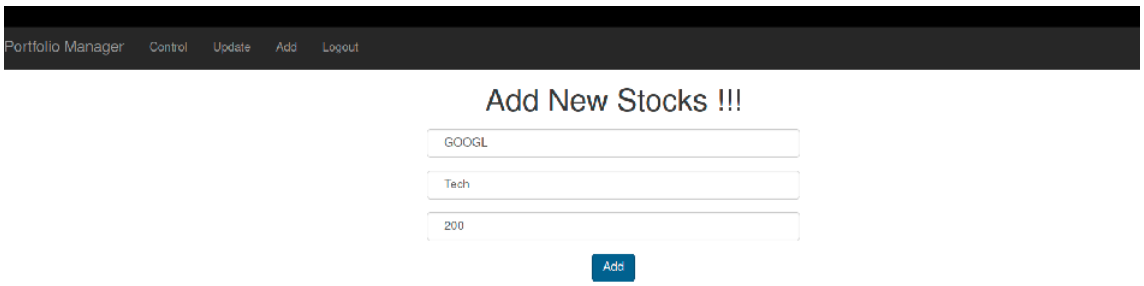


Figure 4: Admin Panel

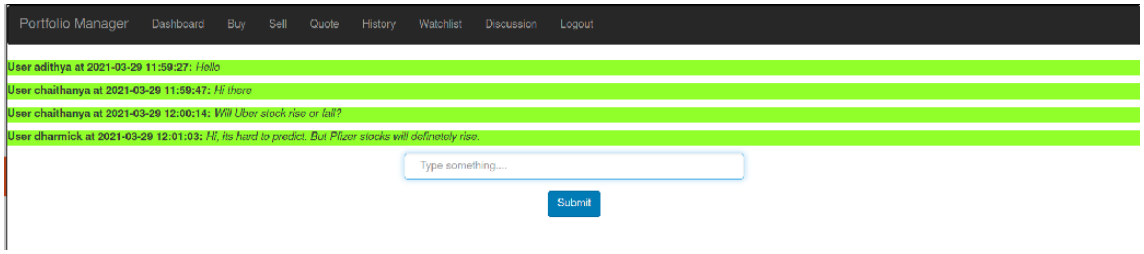


Figure 5: Discussions

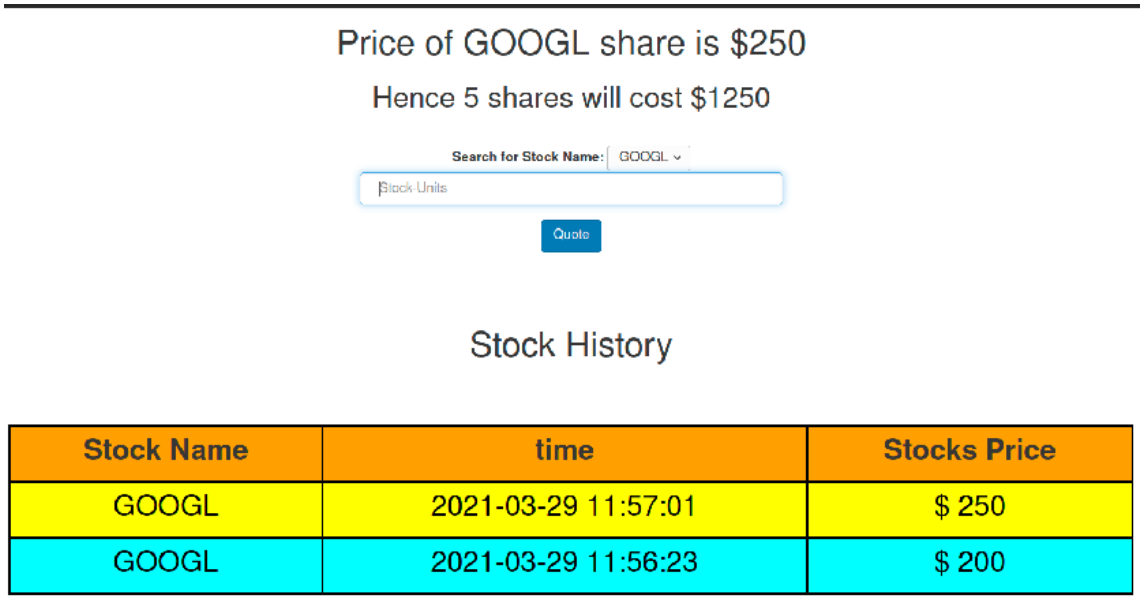


Figure 6: quote and history

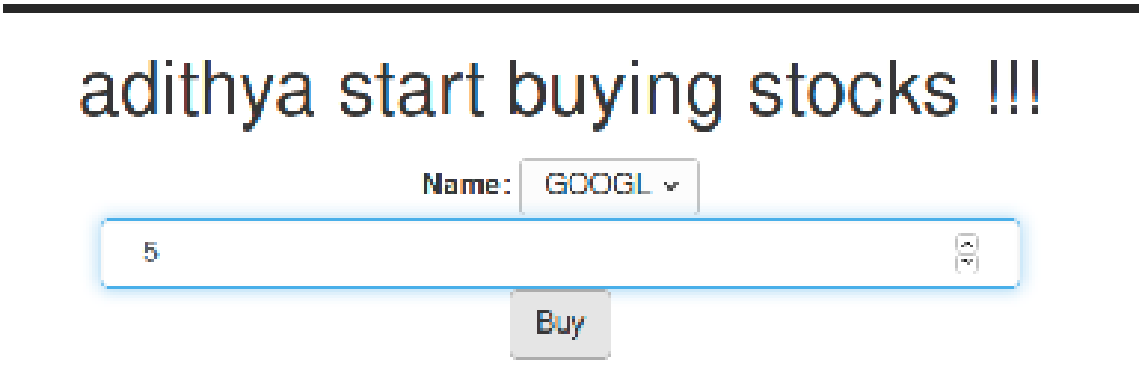
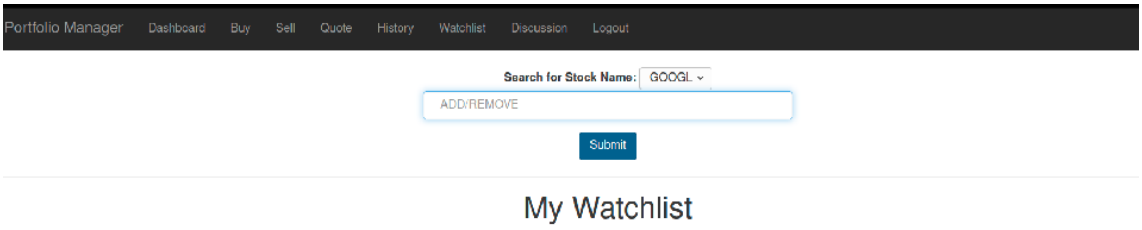


Figure 7: Buy/Sell



My Watchlist

Stock Name	Sector	Stocks Worth
GOOGL	Tech	$ 250
Pfizer	Medical	$ 175

Figure 8: Watchlist

```
mysql> show tables;
+--------------------+
| Tables_in_Platform |
+--------------------+
| Discussion         |
| StockPrice         |
| Stocks             |
| Transactions       |
| User               |
| WatchList          |
+--------------------+
6 rows in set (0.01 sec)
```

Figure 9: Tables

```
mysql> select * from Transactions;
+---------------+--------+-----------+-------+------------+---------------------+
| transactionID | userID | stockName | units | totalValue | transacted          |
+---------------+--------+-----------+-------+------------+---------------------+
|             1 |      2 | GOOGLE    |     5 |       1250 | 2021-03-29 08:36:44 |
|             2 |      2 | GOOGLE    |    -2 |       -500 | 2021-03-29 08:38:17 |
|             3 |      3 | ASSYAD    |     3 |        420 | 2021-03-29 12:18:57 |
|             4 |      3 | MSFT      |     6 |       1080 | 2021-03-29 12:20:33 |
|             5 |      3 | MSFT      |    -2 |       -360 | 2021-03-29 12:20:37 |
+---------------+--------+-----------+-------+------------+---------------------+
5 rows in set (0.00 sec)
```

Figure 10: Query1

```
mysql> with UserStocks(userID,units,stockname) as
    -> ( SELECT userID, units,stockname from Transactions natural join User having userID = 3)
    -> select sum(units)as num_stocks , sum(units)*unitprice as stocksworth, stockname,
    -> sector from UserStocks natural join Stocks  group by stockname ;
+------------+-------------+-----------+--------+
| num_stocks | stocksworth | stockname | sector |
+------------+-------------+-----------+--------+
|          3 |         420 | ASSYAD    | OIL    |
|          4 |         720 | MSFT      | TECH   |
+------------+-------------+-----------+--------+
2 rows in set (0.00 sec)
```

Figure 11: Query2

```
mysql> select * from StockPrice where stockName = 'GOOGLE' order by time desc;
+-----------+---------------------+-------+
| stockName | time                | Price |
+-----------+---------------------+-------+
| GOOGLE    | 2021-03-29 12:23:05 |   230 |
| GOOGLE    | 2021-03-29 12:23:00 |   200 |
| GOOGLE    | 2021-03-29 08:31:33 |   250 |
+-----------+---------------------+-------+
3 rows in set (0.00 sec)
```

Figure 12: Query3

6 References:

1. https://nodejs.org/en/docs/

2. https://cs50.harvard.edu/x/2020/tracks/web/finance/

3. https://medium.com/@ms.jeet1995/connecting-to-mysql-in-nodejs-4bad50ef6740

****** END ******