Mock Trade Platform and Stock Price Analysis

ERG3010 Data and Knowledge Management (2019 Fall)

Final Report

117010082 Han Yi

117010308 Xie Guozheng

117010374 Zhang Fangzhou

117010387 Zhang Wanxuan

118020081 Zhu Yaochen

118020043 Liu Xilin

(Sort by student ID)

December 22, 2019

Contents

1 Abstraction

In this report we will present a comprehensive description of our mock trade platform, from the first idea to the final product. Our website is based on a bootstrap template. Bootstrap is a powerful front-end framework library for web development, adapting websites to mobile, tablets and PC devices efficiently. For the back-end, we adapt MYSQL in Ubuntu 16.04. The design of our web interface utilizes HTML to build the web page frame, CSS works for web page style, and JavaScript to present the web dynamically. In addition, Natural Language Processing (NLP) is used to extract characteristics of the companies. Alongside the basic functions of buying and selling stocks, we use machine learning, specifically Long Short Term Memory (LSTM) model, trying to predict the future stock prices, filtering out the noise in the stock market. After finishing the local development collaborating on GitHub, we use Nginx to deploy our web application on the internet. We rent the server of Aliyun, which can be accessed anywhere in the world, everyone can participate in this trading game. In the end, we made a conclusion of our whole project and our expectation for the future work.

2 Introduction

In the complex and volatile financial environment, we are witnessing an increasing need of gaining practical experience for people who are eager to enter the attractive stock market to get their share. However, the risks are not affordable for most freshmen. We looked up several representative stock trade games and most of them used generated data, which is lack of realistic value. Moreover, the game rule is difficult to understand and the interface are a bit hard to recognize in the first time. To stimulate the whole process, it takes a long time to finish the game. Therefore, we decided to create an easier and more interesting mock trade platform for both freshmen and finance students who are interested in stock market. Historical daily stock information from 2000-2018 are collected from WRDS.

We prepare for our user one million dollar as initial capital to start stock trading, and hope this could help our users to get the basic sense of the true stock market.

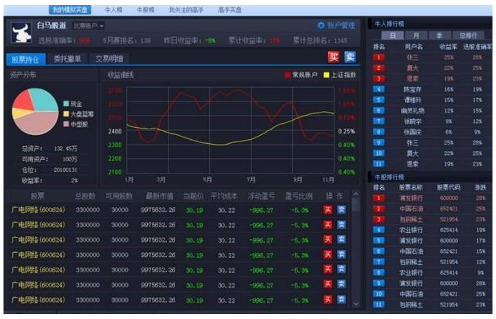
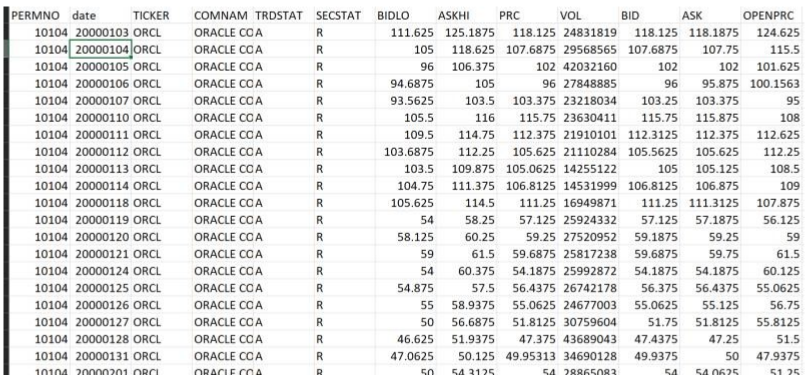


Figure 1: Traditional Mock Trade Game with obscure rules and complex graphs



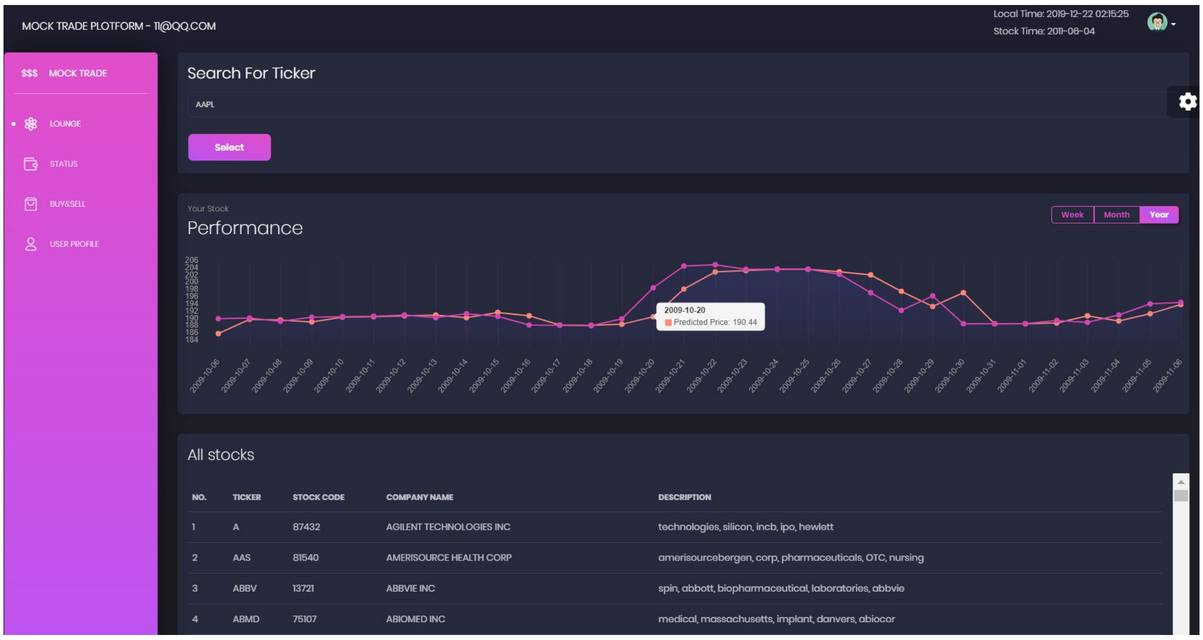
PERMNO	date	TICKER	COMNAM	TRDSTAT	SECSTAT	BIDLO	ASKHI	PRC	VOL	BID	ASK	OPENPRC
10104	20000103	ORCL	ORACLE CO	A	R	111.625	125.1875	118.125	24831819	118.125	118.1875	124.625
10104	20000104	ORCL	ORACLE CO	A	R	105	118.625	107.6875	29568565	107.6875	107.75	115.5
10104	20000105	ORCL	ORACLE CO	A	R	96	106.375	102	42032160	102	102	101.625
10104	20000106	ORCL	ORACLE CO	A	R	94.6875	105	96	27848885	96	95.875	100.1563
10104	20000107	ORCL	ORACLE CO	A	R	93.5625	103.5	103.375	23218034	103.25	103.375	95
10104	20000110	ORCL	ORACLE CO	A	R	105.5	116	115.75	23630411	115.75	115.875	108
10104	20000111	ORCL	ORACLE CO	A	R	109.5	114.75	112.375	21910101	112.3125	112.375	112.625
10104	20000112	ORCL	ORACLE CO	A	R	103.6875	112.25	105.625	21110284	105.5625	105.625	112.25
10104	20000113	ORCL	ORACLE CO	A	R	103.5	109.875	105.0625	14255122	105	105.125	108.5
10104	20000114	ORCL	ORACLE CO	A	R	104.75	111.375	106.8125	14531999	106.8125	106.875	109
10104	20000118	ORCL	ORACLE CO	A	R	105.625	114.5	111.25	16949871	111.25	111.3125	107.875
10104	20000119	ORCL	ORACLE CO	A	R	54	58.25	57.125	25924332	57.125	57.1875	56.125
10104	20000120	ORCL	ORACLE CO	A	R	58.125	60.25	59.25	27520952	59.1875	59.25	59
10104	20000121	ORCL	ORACLE CO	A	R	59	61.5	59.6875	25817238	59.6875	59.75	61.5
10104	20000124	ORCL	ORACLE CO	A	R	54	60.375	54.1875	25992872	54.1875	54.1875	60.125
10104	20000125	ORCL	ORACLE CO	A	R	54.875	57.5	56.4375	26742178	56.375	56.4375	55.0625
10104	20000126	ORCL	ORACLE CO	A	R	55	58.9375	55.0625	24677003	55.0625	55.125	56.75
10104	20000127	ORCL	ORACLE CO	A	R	50	56.6875	51.8125	30759604	51.75	51.8125	55.8125
10104	20000128	ORCL	ORACLE CO	A	R	46.625	51.9375	47.375	43689043	47.4375	47.25	51.5
10104	20000131	ORCL	ORACLE CO	A	R	47.0625	50.125	49.95313	34690128	49.9375	50	47.9375
10104	20000201	ORCL	ORACLE CO	A	R	50	54.3125	54	28865083	54	54.0625	51.25

Figure 2: Parts of our Stock Data

Next is to build up our website. We use Alibaba Cloud - Aliyun, to store our data and cloud functions, from which the front end could apply the cloud functions directly and offer visualization on our website.

3 Data Resources

S&P 500 companies' information are extracted from Wikipedia using web crawler. Stock information are collected from WRDS, a web-based business data research service backed by The Wharton School of the University of Pennsylvania. WRDS provides access to financial, accounting, banking, economics and marketing databases through a uniform interface, enabling comprehensive thought historical analysis and insight into the latest innovations in academic research. The data preprocessing included Data Cleaning, Data Transformation, and Data Integration. After all that, we put the cleansed data into MYSQL database.



All stocks

NO.	TICKER	STOCK CODE	COMPANY NAME	DESCRIPTION
1	A	87432	AGILENT TECHNOLOGIES INC	technologies, silicon, incb, ipo, hewlett
2	AAS	81540	AMERISOURCE HEALTH CORP	amerisourcebergen, corp, pharmaceuticals, OTC, nursing
3	ABBV	13721	ABBVIE INC	spin, abbott, biopharmaceutical, laboratories, abbvie
4	ABMD	75107	ABIOMED INC	medical, massachusetts, implant, danvers, abiocor

Figure 3: Home page of our website

4 Database Management

4.1 Relation Schema

Our data covers all the information of S&P 500 companies and their daily stocks prices during 2000-2018. There are four entities:

1. Company

   This table contains the basic information of the companies. It contains 4 attributes including: stock ticker, company name, trading status and security status. The primary key of this table is stock ticker.

2. Stock Price

   This table contains all the information of the daily stock prices including Ticker, Date, BIDLO (the lowest bid price), ASKHI (the highest asking price), PRC (the closing price), VOL (trading volume), BID (the bid price), ASK (the ask price), and OPENPRC (open price). The primary key of this table is (Ticker, Date).

3. User Account

   This table contains the information of the registered users, including User's ID, Password, Email address, Start Time and Balance. The primary key of this table is ID.

4. Transaction Record

   This table contains the information of users' transaction, including User's ID, Transaction ID, Time, Ticker, Price, Volume. The primary key of this table is (User ID, Transaction ID).

There are two relations:

1. Company stock price

   Connect Company and Stock Price which is one to many.

2. Users behavior

   Connect users' information and their transaction which is one to many.

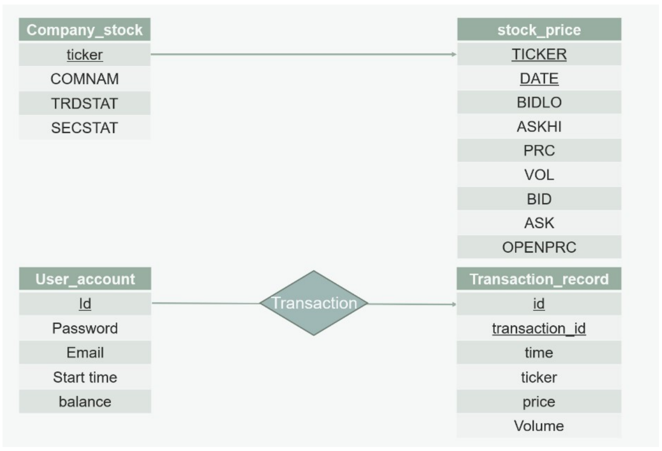


Figure 4: Relation Schema for our company information and user database

4.2 Platform

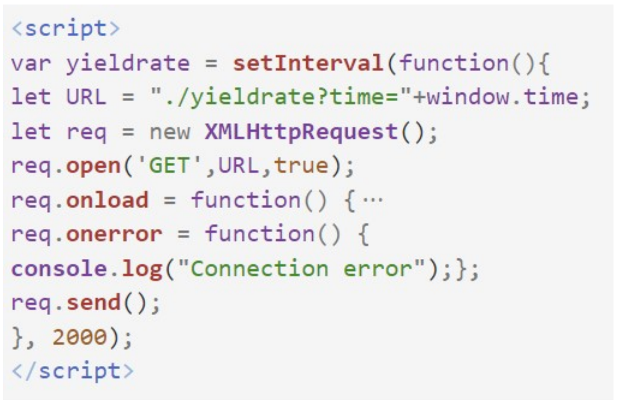
To store and manipulate our huge data efficiently, we decided to build our database on Alibaba Cloud - Aliyun, a cloud server provider. There are two reasons why we chose Alibaba Cloud: (1) Alibaba Cloud offers a set of fully managed database services that support open-source database engines. The database services monitor, backup, and recover our database to guarantee data stability and availability. (2) Alibaba Cloud encrypt data automatically to prevent attacks on the cloud, which is safe for our platform development.

4.3 Data Uploading

We convert our CSV file to SQL format and deployed on the cloud. Panda package in Python is the tool we use.

4.4 Cloud Functions Implementations

Since the stock prices are floating, we need to get the current prices and provide a dynamic graph to show the changes and the yield rate of the players. To achieve real-time update, we used setInterval() function to get real-time data from database every 2 seconds.



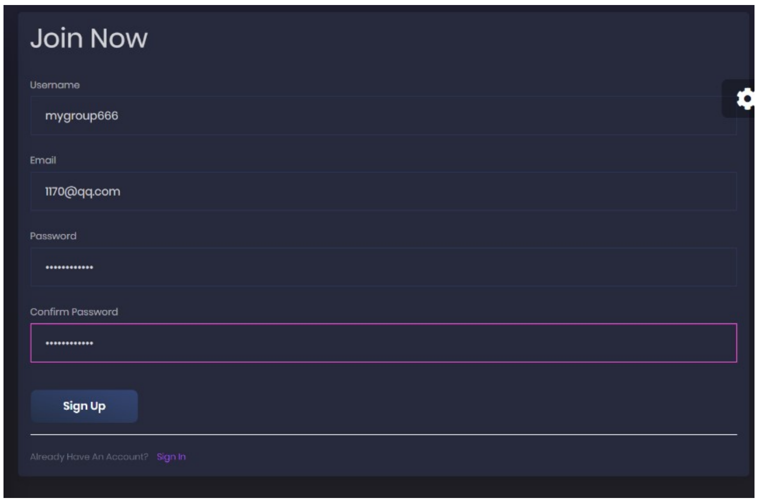
```
<script>
var yieldrate = setInterval(function(){
let URL = "./yieldrate?time="+window.time;
let req = new XMLHttpRequest();
req.open('GET',URL,true);
req.onload = function() {⋯
req.onerror = function() {
console.log("Connection error");};
req.send();
}, 2000);
</script>
```

5 Web Interface Design

In this part, web interface design and its functions will be introduced. Basically, there are 6 functional pages, each with different contents and functions.

1. Registration Page

   This page is for players to sign up. Once they register successfully, the game time is ticking.



Join Now

Username: mygroup666

Email: 1170@qq.com

Password: ••••••••••••

Confirm Password: ••••••••••••

Figure 5: Registration Page

2. Login Page

   This page is for players to log in. They can view, trade and see the prediction after logging in their accounts.

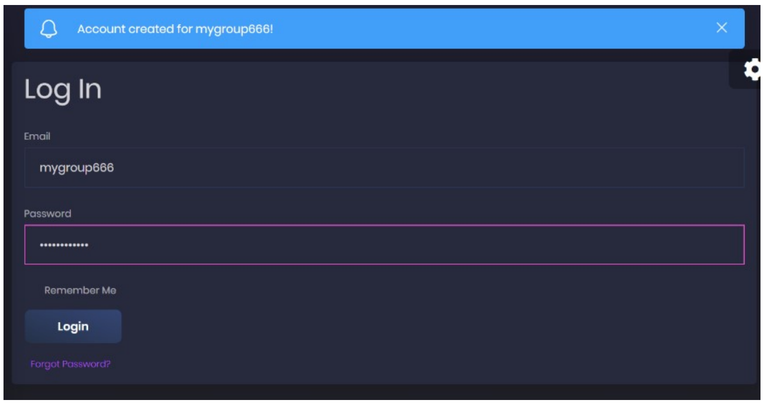

Account created for mygroup666!

Figure 6: Login Page

3. Stock Lounge Page

   This page contains two parts. The upper part shows the performance of a specific stock in some period.

   Users can type in the ticker of the stock to select a specific company to check the stock performance weekly, monthly, and yearly. It is highly effective because users don't need to type in the full ticker. They only need to type in the initial characters without case sensitive, like aa, then all the related results will be listed below, for instance, AAPL(Apple Computer Inc.) and AAS(American Health Cor.).

   The stock lounge page provides dynamic stock price data and companies information shown as below.

   The below part shows the companies information regarding to their ticker and basic descriptions, the descriptions are generated from NLP.

4. Buy & Sell Page

   This page contains 3 parts.

   The upper part contains the transaction operations, where users can buy and sell the stocks to the market freely while the future stock price will not be affected no matter how much he throws.

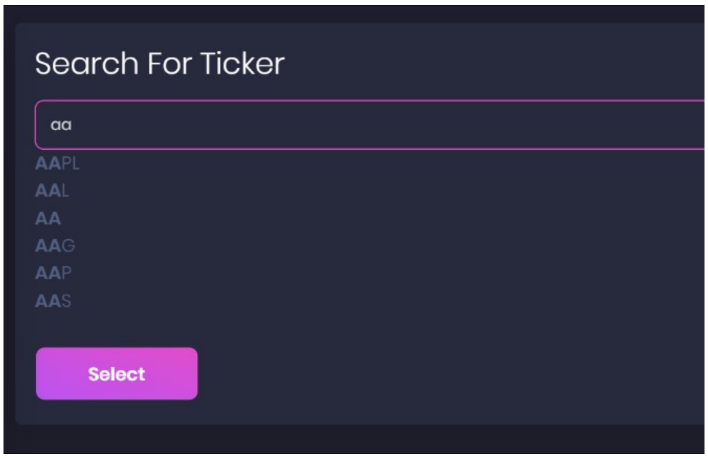


Figure 7: Search function

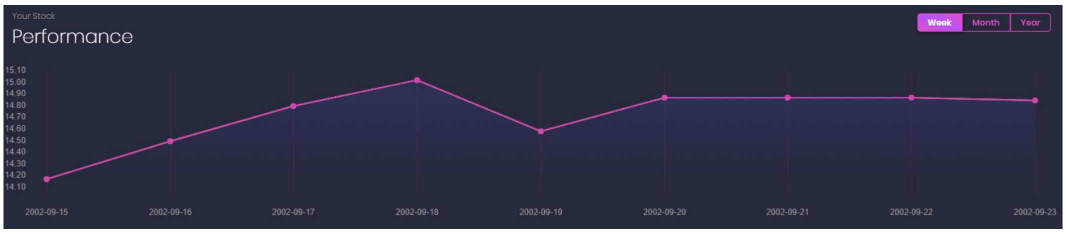


Figure 8: Stock AAPL weekly performance

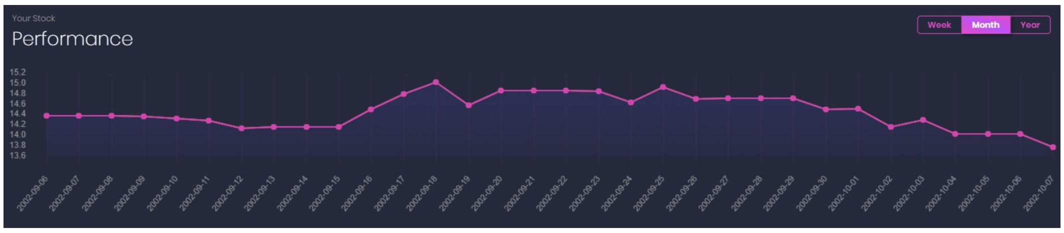


Figure 9: Stock AAPL monthly performance

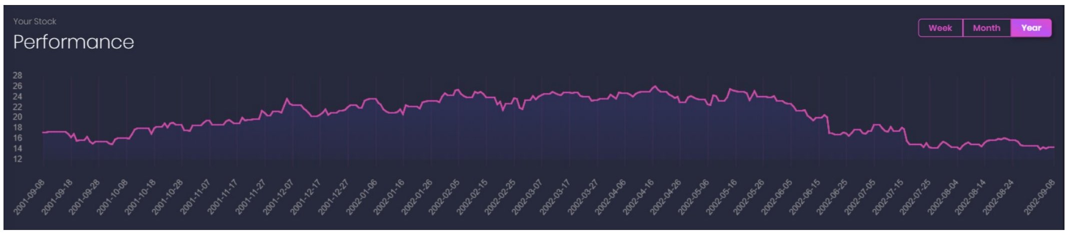
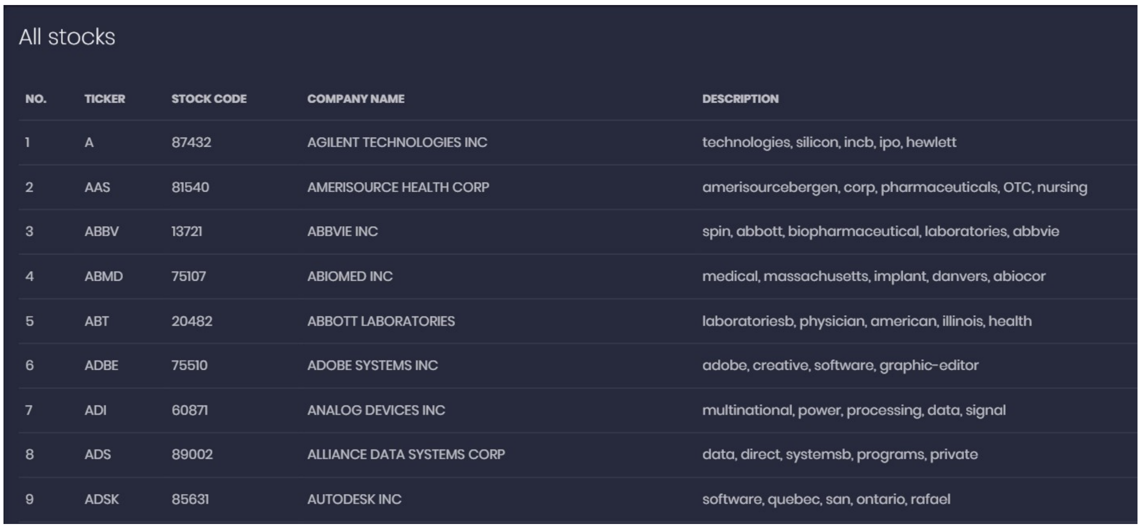


Figure 10: Stock AAPL yearly performance

The middle part is the transaction record, where users can see their past transaction clearly including price and the number of shares.

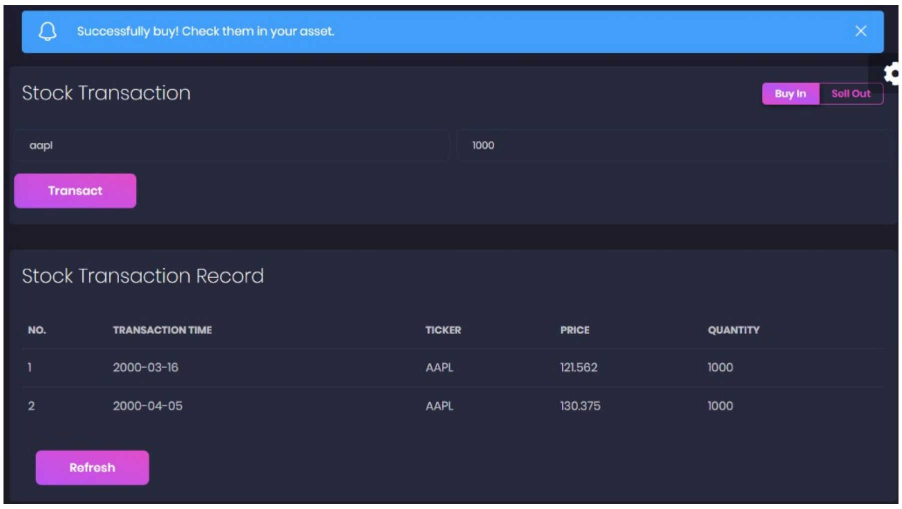
The last part is the recommendation stocks list. The current 10 highest yield stocks in the



All stocks

NO.	TICKER	STOCK CODE	COMPANY NAME	DESCRIPTION
1	A	87432	AGILENT TECHNOLOGIES INC	technologies, silicon, incb, ipo, hewlett
2	AAS	81540	AMERISOURCE HEALTH CORP	amerisourcebergen, corp, pharmaceuticals, OTC, nursing
3	ABBV	13721	ABBVIE INC	spin, abbott, biopharmaceutical, laboratories, abbvie
4	ABMD	75107	ABIOMED INC	medical, massachusetts, implant, danvers, abiocor
5	ABT	20482	ABBOTT LABORATORIES	laboratoriesb, physician, american, illinois, health
6	ADBE	75510	ADOBE SYSTEMS INC	adobe, creative, software, graphic-editor
7	ADI	60871	ANALOG DEVICES INC	multinational, power, processing, data, signal
8	ADS	89002	ALLIANCE DATA SYSTEMS CORP	data, direct, systemsb, programs, private
9	ADSK	85631	AUTODESK INC	software, quebec, san, ontario, rafael

Figure 11: Companies information module



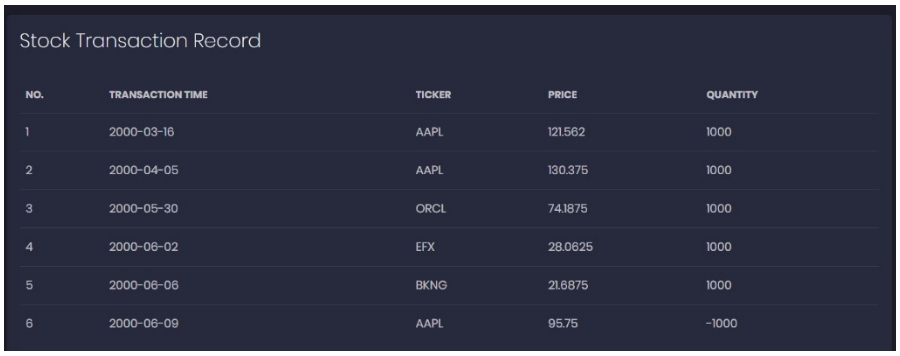
Successfully buy! Check them in your asset.

Stock Transaction

Stock Transaction Record

NO.	TRANSACTION TIME	TICKER	PRICE	QUANTITY
1	2000-03-16	AAPL	121.562	1000
2	2000-04-05	AAPL	130.375	1000

Figure 12: Stock Transaction

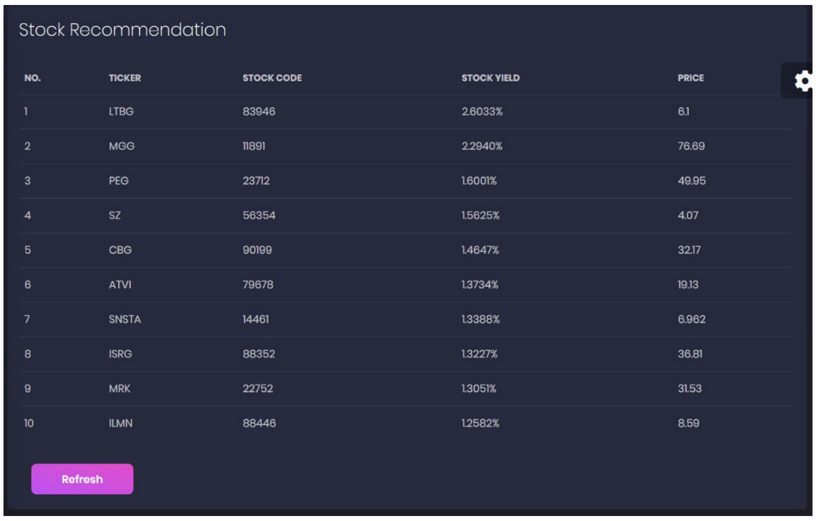


Stock Transaction Record

NO.	TRANSACTION TIME	TICKER	PRICE	QUANTITY
1	2000-03-16	AAPL	121.562	1000
2	2000-04-05	AAPL	130.375	1000
3	2000-05-30	ORCL	74.1875	1000
4	2000-06-02	EFX	28.0625	1000
5	2000-06-06	BKNG	21.6875	1000
6	2000-06-09	AAPL	95.75	-1000

Figure 13: Transaction Record

past month are provided to the players.



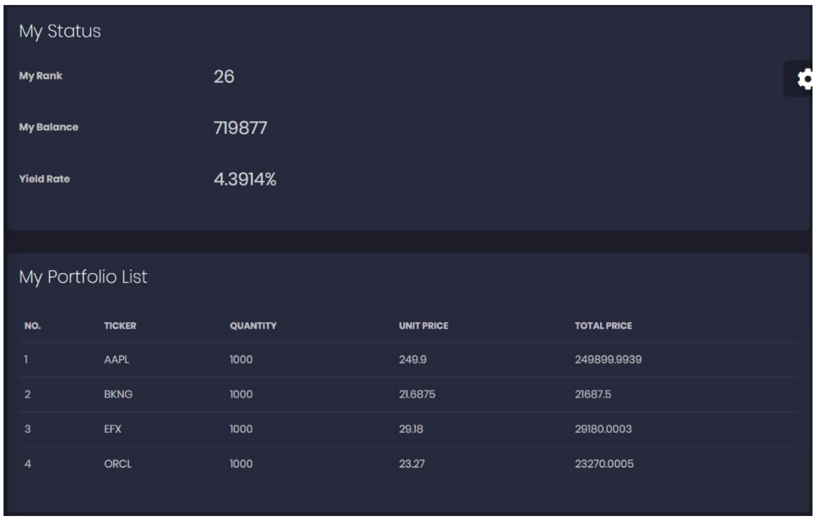
Stock Recommendation

NO.	TICKER	STOCK CODE	STOCK YIELD	PRICE
1	LTBG	83946	2.6033%	6.1
2	MGG	11891	2.2940%	76.69
3	PEG	23712	1.6001%	49.95
4	SZ	56354	1.5625%	4.07
5	CBG	90199	1.4647%	32.17
6	ATVI	79678	1.3734%	19.13
7	SNSTA	14461	1.3388%	6.962
8	ISRG	88352	1.3227%	36.81
9	MRK	22752	1.3051%	31.53
10	ILMN	88446	1.2582%	8.59

Figure 14: Stock Recommendation

5. Status Page

   This page consists of 2 parts. The upper part shows the basic information of the player, including the ranking, balance and current yield rate. The below part shows the shares that the player holds. Both the unit price and the total price are updated every 2 second.



My Status

My Rank	26
My Balance	719877
Yield Rate	4.3914%

My Portfolio List

NO.	TICKER	QUANTITY	UNIT PRICE	TOTAL PRICE
1	AAPL	1000	249.9	249899.9939
2	BKNG	1000	21.6875	21687.5
3	EFX	1000	29.18	29180.0003
4	ORCL	1000	23.27	23270.0005

Figure 15: Status Page

- My Rank is only based on users' current balance, which is ranked among all players.

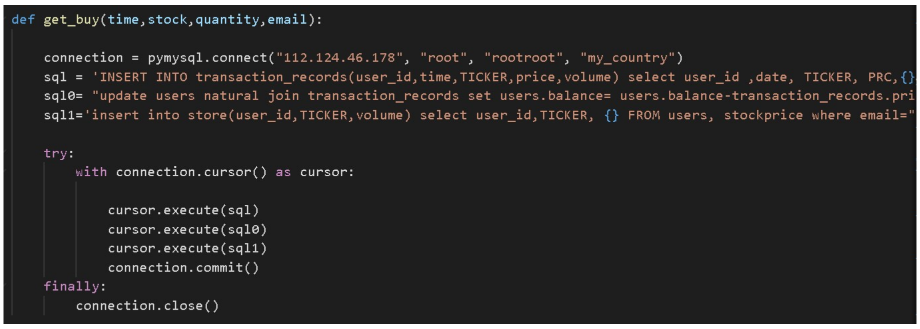
- The initial account is 1 million and My balance is changing by transaction.
- Yield Rate is calculated by the formula:

$$\frac{user'scurrentbalance + currentvalueofstocks–1000000}{1000000}$$

6 Data Visualization and Implementations

- Functions of transaction

  Buy & Sell Every time players call a transaction, the functions will find the corresponding price of the stocks and insert a row to the 'transaction record' table when it is affordable.



```
def get_buy(time,stock,quantity,email):

    connection = pymysql.connect("112.124.46.178", "root", "rootroot", "my_country")
    sql = 'INSERT INTO transaction_records(user_id,time,TICKER,price,volume) select user_id ,date, TICKER, PRC,{}
    sql0= "update users natural join transaction_records set users.balance= users.balance-transaction_records.pri
    sql1='insert into store(user_id,TICKER,volume) select user_id,TICKER, {} FROM users, stockprice where email="

    try:
        with connection.cursor() as cursor:

            cursor.execute(sql)
            cursor.execute(sql0)
            cursor.execute(sql1)
            connection.commit()
    finally:
        connection.close()
```

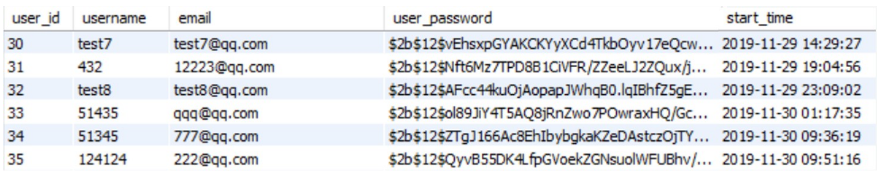
- Balance & Yield Rate

  Balance does not alter until players do transactions. Yield Rate is updating in every 2 seconds.
- Registration detection

  When registering, Non-email address format will be alarmed as well as the used ID and email which has already been used. This is the preservation on data integrity.
- Data injection

  If the selection bar is typed in string like X' or 'Y' = 'Y, the resulting statement will return none. We prevent some basic SQL injection on Flask side.
- Data security

  For data security, the users' password is hashed. Even if our database is hacked, the information will not be compromised.



user_id	username	email	user_password	start_time
30	test7	test7@qq.com	$2b$12$vEhsxpGYAKCKYyXCd4TkbOyv17eQcw...	2019-11-29 14:29:27
31	432	12223@qq.com	$2b$12$Nft6Mz7TPD8B1CiVFR/ZZeeLJ2ZQux/j...	2019-11-29 19:04:56
32	test8	test8@qq.com	$2b$12$AFcc44kuOjAopapJWhqB0.lqIBhfZ5gE...	2019-11-29 23:09:02
33	51435	qqq@qq.com	$2b$12$ol89JiY4T5AQ8jRnZwo7POwraxHQ/Gc...	2019-11-30 01:17:35
34	51345	777@qq.com	$2b$12$ZTgJ166Ac8EhIbybgkaKZeDAstczOjTY...	2019-11-30 09:36:19
35	124124	222@qq.com	$2b$12$QyvB55DK4LfpGVoekZGNsuolWFUBhv/...	2019-11-30 09:51:16

Figure 16: Sample record of the user data

7 Data Analysis

7.1 GAIN OR LOSS? Doing Prediction to Analyze Underlying Factors

In the beginning, we never thought that any stock forecasting model we were doing was completely effective. In view of the factors that affect stock price volatility, the financial community has not yet reached a firm conclusion. However, we can still see that many factors have indeed affected the trend of stock prices. Because of the existence of professional traders, which occupy most of the market composition. We have reasons to believe that there are some inherent patterns in the stock market data that can help people identify the rise and fall of stocks.

We hope to use the LSTM model to capture these specific laws in the data and give predictions of stock price changes. These predictions completely filter out the effects of irrational factors in the market and help users analyze stocks.

Traditional neural networks cannot interpret input sequences that depend on information and context. The information can be a previously appearing word in a sentence to allow the context to predict what the next word might be. Or it can be the time information of the sequence, it will allow context for time-based elements of the sequence. In short, in traditional neural networks, each input is stand-alone data vector and there is no concept of memory.

RNN has the Vanishing Gradient Problem, while LSTM solves the sequence and time problems by keeping the context of memory in its pipeline through neurons, so there is no problem of gradient disappearance that affects performance.

The following figure is the internal schematic of LSTM neuron, which includes pointwise operations that play a gating role on data input, output, and forgetting. It is used as the input of cell state, which is reserved in the cell state the long-term memory and contextual information of the network and input.

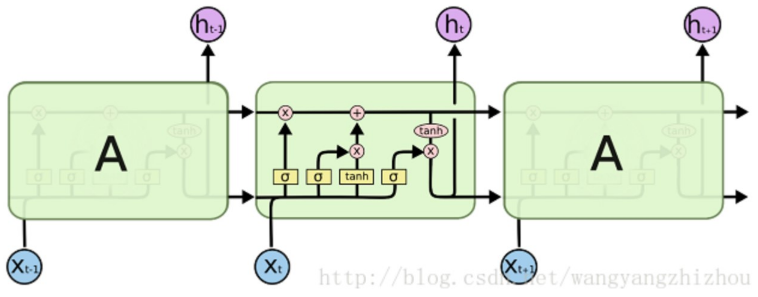
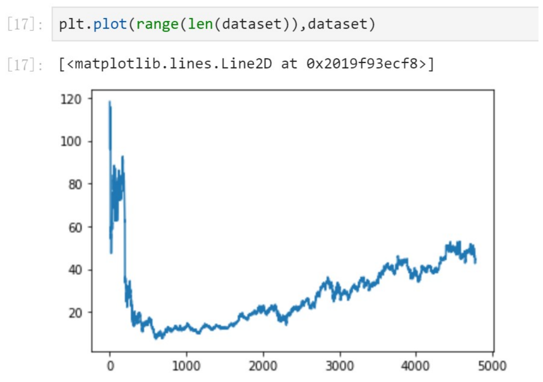


Figure 17: Internal schematic of LSTM neuron

Procedure: We use the stock price of PERMNO as 10104 as the data, this is the price curve



```
[17]: plt.plot(range(len(dataset)),dataset)

[17]: [<matplotlib.lines.Line2D at 0x2019f93ecf8>]
```

Figure 18: Price curve of stock 10104

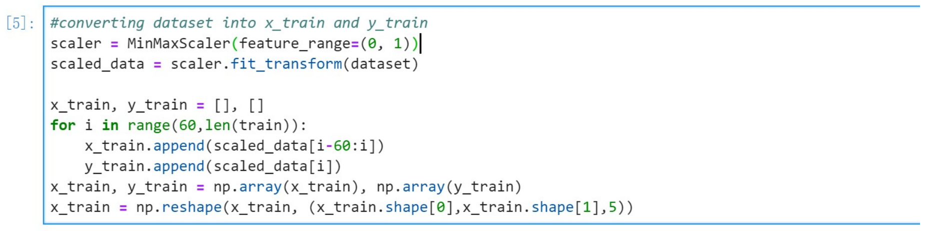
The closing price is the ever-changing absolute price of the stock market. This means that if data is not standardized, it will never converge. To address this, we will take a single data window and normalize the data to reflect the percentage change from that window (so the data at point i = 0 will always be 0). We will use the following equation for normalization and then de-normalization at the end of the prediction process to obtain the true data in the prediction.

$$Normalization : n_i = (\frac{p_i}{p_0}) - 1$$

$$De - Normalization : p_i = p_0(n_i + 1)$$

Since the data set is large enough, we take the first 0.4 of each stock as the training set.

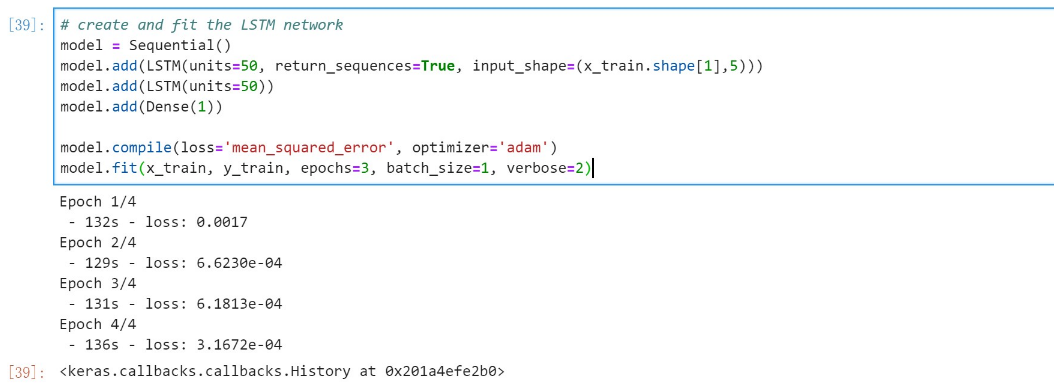
At the same time, in our model, the time step is 60 and epochs = 4. At the same time, in order to solve the problem that the size of the data set may be different for different stocks (some stocks may be shortened because of the suspension). We encapsulate the 60-day dataset into the same numpy.array input model.



```
[5]: #converting dataset into x_train and y_train
     scaler = MinMaxScaler(feature_range=(0, 1))
     scaled_data = scaler.fit_transform(dataset)

     x_train, y_train = [], []
     for i in range(60,len(train)):
         x_train.append(scaled_data[i-60:i])
         y_train.append(scaled_data[i])
     x_train, y_train = np.array(x_train), np.array(y_train)
     x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],5))
```

We use keras to build our model. The first two layers are LSTM models. Finally, we added a simple dense model to capture the linear features extracted by the first two layers.



```
[39]: # create and fit the LSTM network
      model = Sequential()
      model.add(LSTM(units=50, return_sequences=True, input_shape=(x_train.shape[1],5)))
      model.add(LSTM(units=50))
      model.add(Dense(1))

      model.compile(loss='mean_squared_error', optimizer='adam')
      model.fit(x_train, y_train, epochs=3, batch_size=1, verbose=2)

      Epoch 1/4
       - 132s - loss: 0.0017
      Epoch 2/4
       - 129s - loss: 6.6230e-04
      Epoch 3/4
       - 131s - loss: 6.1813e-04
      Epoch 4/4
       - 136s - loss: 3.1672e-04
[39]: <keras.callbacks.callbacks.History at 0x201a4efe2b0>
```

Then we input the test data set for prediction

7.2 BUY OR SELL? Using Natural Language Processing (NLP) to label the companies

Considering that users may not be very clear about all companies of S&P 500, we hope to extract the keywords described by the company through NLP.
First, we found the 'list of S&P 500' in Wiki, and got all companies' links from the table in the body. Crawled the company's homepage code pointed to by the link, and then used package RE to get the description we need.
Regarding the NLP algorithm, we use textrank to obtain 5 keywords for each company. The core ideas of the algorithm are as follows:

```
[44]: rms=np.sqrt(np.mean(np.power((valid-closing_price),2)))
      rms
```

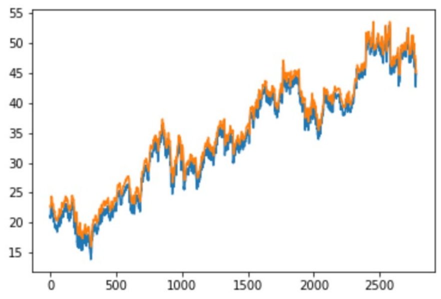
```
[44]: 1.385004454048496
```

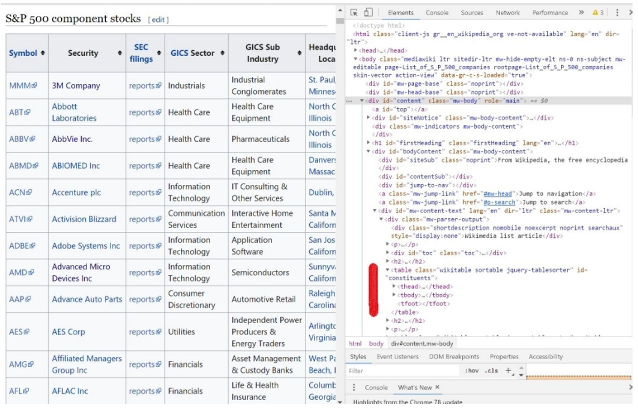
```
[45]: #for plotting
      train = new_data[:2000]
      valid = new_data[2000:]
      valid['Predictions'] = closing_price
      plt.plot(range(len(valid)),valid[['PRC','Predictions']])
```

```
c:\users\l2766\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:4: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-c
  after removing the cwd from sys.path.
```

```
[45]: [<matplotlib.lines.Line2D at 0x201a949dac8>,
       <matplotlib.lines.Line2D at 0x201a94b2588>]
```



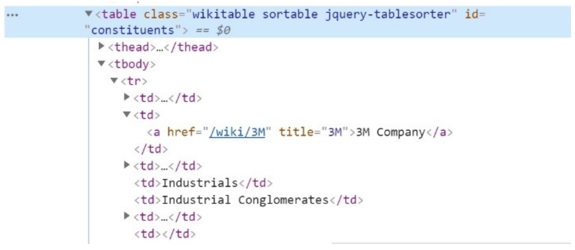


S&P 500 component stocks [edit]

Symbol	Security	SEC filings	GICS Sector	GICS Sub Industry	Headqu Loca
MMM	3M Company	reports	Industrials	Industrial Conglomerates	St. Paul, Minnes
ABT	Abbott Laboratories	reports	Health Care	Health Care Equipment	North C Illinois
ABBV	AbbVie Inc.	reports	Health Care	Pharmaceuticals	North C Illinois
ABMD	ABIOMED Inc	reports	Health Care	Health Care Equipment	Danvers Massac
ACN	Accenture plc	reports	Information Technology	IT Consulting & Other Services	Dublin,
ATVI	Activision Blizzard	reports	Communication Services	Interactive Home Entertainment	Santa M Californ
ADBE	Adobe Systems Inc	reports	Information Technology	Application Software	San Jos Californ
AMD	Advanced Micro Devices Inc	reports	Information Technology	Semiconductors	Sunnyv Californ
AAP	Advance Auto Parts	reports	Consumer Discretionary	Automotive Retail	Raleigh Carolina
AES	AES Corp	reports	Utilities	Independent Power Producers & Energy Traders	Arlingto Virginia
AMG	Affiliated Managers Group Inc	reports	Financials	Asset Management & Custody Banks	West Pa Beach, I
AFL	AFLAC Inc	reports	Financials	Life & Health Insurance	Columb Georgia

```
<!doctype html>
<html class="client-js gr__en_wikipedia_org ve-not-available" lang="en" dir=
"ltr">
 ▸<head>…</head>
 ▾<body class="mediawiki ltr sitedir-ltr mw-hide-empty-elt ns-0 ns-subject mw-
 editable page-List_of_S_P_500_companies rootpage-List_of_S_P_500_companies
 skin-vector action-view" data-gr-c-s-loaded="true">
    <div id="mw-page-base" class="noprint"></div>
    <div id="mw-head-base" class="noprint"></div>
…  ▾<div id="content" class="mw-body" role="main"> == $0
      <a id="top"></a>
     ▸<div id="siteNotice" class="mw-body-content">…</div>
      <div class="mw-indicators mw-body-content">
      </div>
     ▸<h1 id="firstHeading" class="firstHeading" lang="en">…</h1>
     ▾<div id="bodyContent" class="mw-body-content">
        <div id="siteSub" class="noprint">From Wikipedia, the free encyclopedia
        </div>
        <div id="contentSub"></div>
        <div id="jump-to-nav"></div>
        <a class="mw-jump-link" href="#mw-head">Jump to navigation</a>
        <a class="mw-jump-link" href="#p-search">Jump to search</a>
       ▾<div id="mw-content-text" lang="en" dir="ltr" class="mw-content-ltr">
         ▾<div class="mw-parser-output">
            <div class="shortdescription nomobile noexcerpt noprint searchaux"
            style="display:none">Wikimedia list article</div>
           ▸<p>…</p>
           ▸<div id="toc" class="toc">…</div>
           ▸<h2>…</h2>
           ▾<table class="wikitable sortable jquery-tablesorter" id=
           "constituents">
             ▸<thead>…</thead>
             ▸<tbody>…</tbody>
              <tfoot></tfoot>
            </table>
           ▸<h2>…</h2>
           ▸<p>…</p>
```

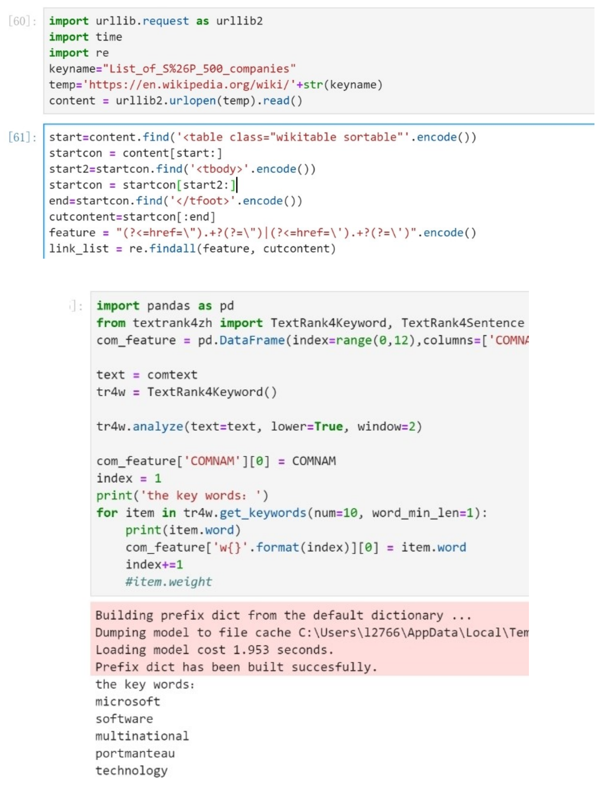
Figure 19: S&P 500 list in Wikipedia and the homepage code



```
…  ▾<table class="wikitable sortable jquery-tablesorter" id=
    "constituents"> == $0
     ▸<thead>…</thead>
     ▾<tbody>
       ▾<tr>
         ▸<td>…</td>
         ▾<td>
            <a href="/wiki/3M" title="3M">3M Company</a>
          </td>
         ▸<td>…</td>
          <td>Industrials</td>
          <td>Industrial Conglomerates</td>
         ▸<td>…</td>
          <td></td>
```

If a word appears after many words, then that word is more important.

A word followed by a word with a high TextRank value will increase the TextRank value of the word accordingly.



```
[60]: import urllib.request as urllib2
      import time
      import re
      keyname="List_of_S%26P_500_companies"
      temp='https://en.wikipedia.org/wiki/'+str(keyname)
      content = urllib2.urlopen(temp).read()

[61]: start=content.find('<table class="wikitable sortable"'.encode())
      startcon = content[start:]
      start2=startcon.find('<tbody>'.encode())
      startcon = startcon[start2:]
      end=startcon.find('</tfoot>'.encode())
      cutcontent=startcon[:end]
      feature = "(?<=href=\").+?(?=\")|(?<=href=\').+?(?=\')".encode()
      link_list = re.findall(feature, cutcontent)
```

```
]: import pandas as pd
   from textrank4zh import TextRank4Keyword, TextRank4Sentence
   com_feature = pd.DataFrame(index=range(0,12),columns=['COMNA

   text = comtext
   tr4w = TextRank4Keyword()

   tr4w.analyze(text=text, lower=True, window=2)

   com_feature['COMNAM'][0] = COMNAM
   index = 1
   print('the key words: ')
   for item in tr4w.get_keywords(num=10, word_min_len=1):
       print(item.word)
       com_feature['w{}'.format(index)][0] = item.word
       index+=1
       #item.weight

   Building prefix dict from the default dictionary ...
   Dumping model to file cache C:\Users\12766\AppData\Local\Ten
   Loading model cost 1.953 seconds.
   Prefix dict has been built succesfully.
   the key words:
   microsoft
   software
   multinational
   portmanteau
   technology
```

Figure 20: Key words results of Microsoft stock

8 Future Work

Our website is a basic mock trade platform and there is still a lot we can do to provide a better game experience. Therefore, in the future, call & put functions can be added in stock trading parts to improve the transaction efficiency. Candlestick chart (k line) can also be applied to show the thorough market fluctuation. On the other hand, we can use PERMNO to analyze the contributing factors of the stock price.

9 Tasks Assignment (In Alphabetical Order)

Han Yi: Web development (Lounge page design)

Liu Xilin: Web spider, Data cleaning and machining learning (NLP and neural network)

Xie Guozheng: Web development (Buy&Sell page design)

Zhang Fangzhou: Schema design and report writing

Zhang Wanxuan: Web development (Lounge and Status page design)

Zhu Yaochen: Web development (Stock data collection, template adaptation, game mechanics, framework design, website chart, register and login page, application deployment)

10 Reference

https://wrds-sol1.wharton.upenn.edu/

https://demos.creative-tim.com/black-dashboard/examples/map.html

https://www.youtube.com/watch?v=Z1RJmh_OqeA&t=2532s

https://www.youtube.com/watch?v=rJesac0_Ftw

11 Appendix

You are welcome to visit our website:

http://112.124.46.178:5000/

Our Source Code:

https://github.com/paradoxXD/my_country