2008

SOFTWARE ENGINNERING OF WEB APPLICATIONS

GROUP # 5

WEB BASED STOCK FORECASTERS

This report describes the Stock Prediction System titled “TIMING THE MARKET” developed by our team which has various modules like Data Mining, Web Services, Neural Network Based Stock Prediction and Technical Indicators.

SUBMITTED BY:

AMARINDER CHEEMA
ATEET VORA
CHETAN JAIN
PUNEET KATARIA
RONAK SHAH
SIDDHARTH WAGH

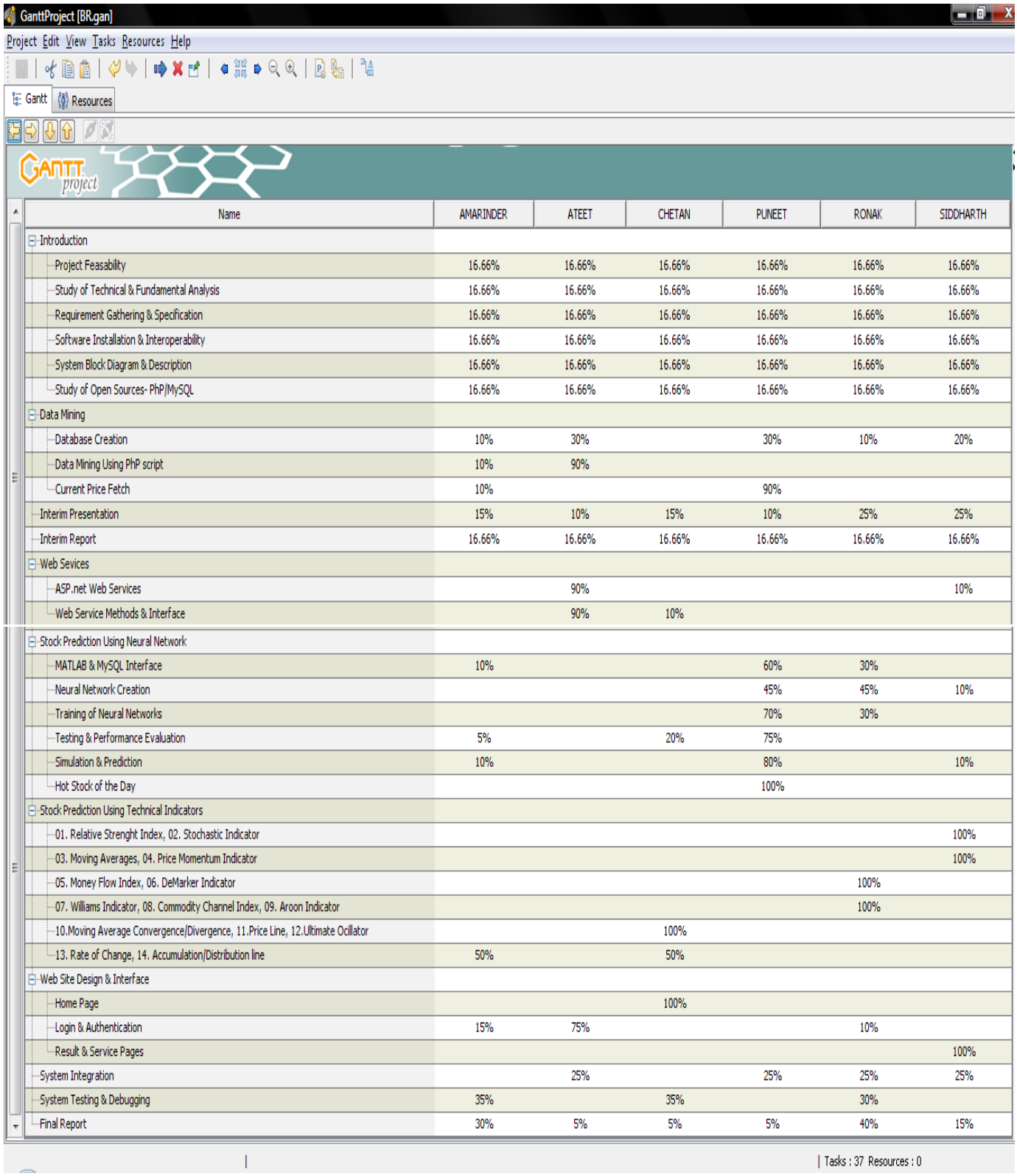
7 May, 2008

ACKNOWLEDGEMENT

It is a moment of great pleasure and satisfaction for us to express our sense of profound gratitude to all people who have contributed in making our project work a rich experience.

We convey our sincere thanks Prof. Ivan Marsic for giving us an opportunity to take up the project work and also helping us in all phases of the project work and providing our team sound encouragement. We are also thankful to him for the technical guidance, help and facilities provided to us for successful completion of our project work.

BREAKDOWN OF RESPONSIBILITIES



Name	AMARINDER	ATEET	CHETAN	PUNEET	RONAK	SIDDHARTH
Introduction						
Project Feasability	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Study of Technical & Fundamental Analysis	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Requirement Gathering & Specification	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Software Installation & Interoperability	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
System Block Diagram & Description	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Study of Open Sources- PhP/MySQL	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Data Mining						
Database Creation	10%	30%		30%	10%	20%
Data Mining Using PhP script	10%	90%				
Current Price Fetch	10%			90%		
Interim Presentation	15%	10%	15%	10%	25%	25%
Interim Report	16.66%	16.66%	16.66%	16.66%	16.66%	16.66%
Web Sevices						
ASP.net Web Services		90%				10%
Web Service Methods & Interface		90%	10%			
Stock Prediction Using Neural Network						
MATLAB & MySQL Interface	10%			60%	30%	
Neural Network Creation				45%	45%	10%
Training of Neural Networks				70%	30%	
Testing & Performance Evaluation	5%		20%	75%		
Simulation & Prediction	10%			80%		10%
Hot Stock of the Day				100%		
Stock Prediction Using Technical Indicators						
01. Relative Strenght Index, 02. Stochastic Indicator						100%
03. Moving Averages, 04. Price Momentum Indicator						100%
05. Money Flow Index, 06. DeMarker Indicator					100%	
07. Williams Indicator, 08. Commodity Channel Index, 09. Aroon Indicator					100%	
10.Moving Average Convergence/Divergence, 11.Price Line, 12.Ultimate Ocillator			100%			
13. Rate of Change, 14. Accumulation/Distribution line	50%		50%			
Web Site Design & Interface						
Home Page			100%			
Login & Authentication	15%	75%			10%	
Result & Service Pages						100%
System Integration		25%		25%	25%	25%
System Testing & Debugging	35%		35%		30%	
Final Report	30%	5%	5%	5%	40%	15%

TABLE OF CONTENTS

TABLE OF FIGURES

SUMMARY OF CHANGES

- The report now mentions the number of companies for which we have developed our Stock Prediction Model which is 20. (Not mentioned in previous report)

- The PHP script imports data every 15 minutes because certain indicators implement formula which requires current stock volume and price to give correct prediction result.

- The system block diagram has been modified and a brief description is provided for more clarity which was missing in earlier report.

- The use case diagram has been modified as per requirement.

- The report now includes UML – Class, Interaction and Activity Diagrams for each module of our system.

- Web services Class Diagrams with explanation of classes is now a part of the report which was missing earlier.

- Database snapshot has now been replaced with a database schema diagram.

- Web services earlier written in Java have now been modified to ASP.NET due to interface issues.

- References to text and diagrams have been duly provided where ever required.

GLOSSARY OF TERMS

Artificial Intelligence
The field of computer science dedicated to producing programs that attempt to mimic the processes of the human brain.

utocorrelation
The correlation between the values of a time series and previous values of the same time series.

Back-Propagation Network
A feed forward multilayered neural network that is a commonly used neural network paradigm.

Bear Market
A securities market characterized thus based on declining prices.

Bull Market
A securities market characterized thus on rising prices.

Buy and Hold
The acquisition of a tradable for the long term rather than quick turnover.

Charts
A display or picture of a security that plots price and/or volume (the number of shares sold).

Confidence Level
The degree of assurance that a specified failure rate is not exceeded.

Data mining
Process of sorting through large amounts of data and picking out relevant information.

Epoch
Presentation of the set of training (input and/or target) vectors to a network and the calculation of new weights and biases. Note that training vectors can be presented one at a time or all together in a batch.

Fundamental Analysis
The analytical method by which only the sales, earnings and the value of a given tradable's assets may be considered.

Gantt charts
Bar charts illustrating the start and finish dates of the terminal elements and summary elements of a project

Hidden layer
Layer of a network that is not connected to the network output (for instance, the first layer of a two-layer feed-forward network).

Input layer
Layer of neurons receiving inputs directly from outside the network.

Learning rule
Learning rule in which weights and biases are adjusted by error-derivative (delta) vectors back-propagated through the network. Back-propagation is commonly applied to feed forward multilayer networks. Sometimes this rule is called the generalized delta rule.

Momentum Indicator
A market indicator utilizing price and volume statistics for predicting the strength or weakness of a current market and any overbought or oversold conditions, and to note turning points within the market.

Moving Average
A mathematical procedure to smooth or eliminate the fluctuations in data and to assist in determining when to buy and sell.

Neural Network
An artificial intelligence program that is capable of learning through a training process of trial and error.

Neuron
Basic processing element of a neural network. Includes weights and bias, a summing junction, and an output transfer function. Artificial neurons, such as those simulated and trained with this toolbox, are abstractions of biological neurons.

Output layer
Layer whose output is passed to the world outside the network.

Pattern recognition
Task performed by a network trained to respond when an input vector close to a learned vector is presented. The network “recognizes” the input as one of the original target vectors.

Regression
A mathematical way of stating the statistical linear relationship between one independent and one dependent variable

Relative Strength Index An indicator invented by J. Welles Wilder and used to ascertain overbought/oversold and divergent situations.

Resilient back-propagation
Training algorithm that eliminates the harmful effect of having a small slope at the extreme ends of the sigmoid squashing transfer functions.

Resistance Line
On a chart, a line drawn indicating the price level at which rising prices have stopped rising and have moved sideways or reversed direction.

Skew
A descriptive measure of lopsidedness in a distribution.

Stochastic Oscillator
An overbought/oversold indicator that compares today's price to a preset window of high and low prices. These data are then transformed into a range between zero and 100 and then smoothed.

Support Line
On a chart, a line drawn, indicating the price level at which falling prices have stopped falling and have moved sideways or reversed direction.

Technical Analysis
A form of market analysis that studies demand and supply for securities and commodities based on trading volume and price studies using charts and modeling techniques.

Tick
The minimum fluctuation of a trading instrument.

Training
Procedure whereby a network is adjusted to do a particular job. Commonly viewed as an offline job, as opposed to an adjustment made during each time interval, as is done in adaptive training.

Trend
The general drift, tendency or bent of a set of statistical data as related to time.

Vector
Column vector of weights coming from a neuron or input.

Volume
The shares which are traded for a given market or tradable within a specified time period.

Web service
Software system designed to support interoperable Machine to Machine interaction over a network.

1. INTRODUCTION

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on a financial exchange. The successful prediction of a stock's future price could yield significant profit. The stock market is not an efficient market. Herding behavior is common among investors, all investors do not get all information at the same time and the time it takes to evaluate information before they act differs between investors. Many investors do not show rational behavior. Greed and fear are strong feelings and may result in panic sales and stock market bubbles. Hence, to regulate the stock market to obtain maximum profit or achieve a certain objective in general without falling prey to inconsistencies, predicting stock behavior is a pressing requirement.

Prediction methodologies fall into two broad categories: fundamental analysis and technical analysis. Fundamental analysis of a business involves analyzing its income statement, financial statements and health, its management and competitive advantages, and its competitors and markets. It is more subjective compared to technical analysis. Fundamental analysis maintains that markets may misprice a security in the short run but that the "correct" price will eventually be reached. Profits can be made by trading the mispriced security and then waiting for the market to recognize its "mistake" and recalculate the security price. On the other hand, Technical Analysis is an approach that uses information of past stock behavior in order to forecast future price movements. Within the technical analysis community there exist several schools with different techniques, but they all have in common that they use price and volume history. A basic thought is that it takes time before the market reacts upon new information and that pattern often occur in price behavior which makes forecasting possible.

There are several factors that explain why technical analysis works:

1. Most speculators on the market act upon fundamental analysis, so that kind of facts influence stock prices strongly. But all operators do not get this information at the same time. When there are positive news of a company, those acting immediately can buy shares for a lower price than those getting the news later.
2. Large investors such as mutual funds and banks are often not placing their whole block orders at the same time when they are buying larger quantities of securities because this would risk triggering an unnecessary high price advance. Instead, the orders are spread over a period that can last several weeks. The resulting increased purchase pressure may result in a steady advancing trend under the period the purchases continue.
3. It is more psychological stressing to go against the trend than to follow it. People are herding animals and like to do as others are doing. This is why a rising stock price is a signal in itself that the price will advance even more. Of course one has to be careful with stocks that have been rocketing, because they will often recoil.

Hence, we limit our focus to technical analysis which has time and again proved its supremacy over other methods. There are many tools available to investors using technical analysis but none of them removes entirely the element of chance from investment decisions. Large trading organizations can employ sophisticated computer systems and armies of analysts. We, as students, attempt to employ a simple set of formidable techniques to achieve the same result for the benefit of small-time investors who cannot afford to hire experts or buy costly softwares to make their investment decisions.

1.1 Project Goals and Requirements:

- **Selection of target customers:** Our customers are small-time daily or weekly investors who trade on an individual basis and who do not have the time or resources to avail of commercial forecasting services or hire agents.
- **Selection of information to be tracked:** We aim to collect and use 10 indicators or patterns for performing technical analysis and provide predictions. We also plan to provide services like RSS feeds, current stock quotes, price charts, recommendations and alerts to help our customer in making a wise investment decision.
- **Data collection:** Our major data source is the Google Finance service which provides us with daily stock prices for an entire year. We mine current prices of stocks from Yahoo Finance.
- **Charting:** Based on the information in the database, we plan to display the stock prices to the end-user using charts.
- **Machine learning:** The logic to recognize trends in market for past one year and make wise decisions based on statistical inference should be coded in the machine in the form of efficient algorithms. We plan to use technical indicators and neural networks to frame such algorithms.
- **Implement Web services:** We aim to design web services to connect to prediction models that track different stocks as queried by user and issue forecasts about price movement of a given stock.
- **Design web interface:** A user-friendly web interface needs to be created and hosted to aid users to get valuable information and timely recommendations and tips about dealing with their stock options.

1.2 System requirements:

- The system should collect data from web sources like Yahoo Finance, Google finance at a time interval of 15 minutes and update its database from a set of 25 companies.
  http://finance.google.com
  http://finance.yahoo.com

- The system should maintain a host of web services that fairly link different service modules with client interface.

- Different prediction models need to be built that give a fair recommendation to the customer whether to buy/sell/hold stock. The system should train itself from a set of past data and simulate on a remaining set of test data to improve its precision.

- The system should aggregate all indications given by technical indicators and try to provide a recommendation that supports the more robust neural network system.

- The system should track various stock movements simultaneously that might not be owned currently by customers and send alerts if it notices a favorable tilt in the graph that hints at immediate purchase.

- A friendly user interface should be designed for customers and access to specific services should be restricted to only registered users to dissuade hackers from destabilizing the system.

- Various customer profiles should be maintained by the system and confidential customer data should be protected at all costs from prying Softwares.

- The system should regularly send emails or short messages to the customer informing him about the current trends and future scenarios. It could feed the user current news and conduct surveys to judge its performance.

- The system may provide in-detail risk analysis of user portfolios to generate Sharpe Ratio to evaluate their current and future risk standing in the market and take appropriate corrective measures.

1.3 Statistical comparison of our product with current software's in market:

Softwares	Attributes					
	Charting	Verification	Built-in indicators	Data feed	Online/ Download	Alerts
Optimal Trader	Yes	No	15	EOD, Delayed	D	No
Ninja Trader	Yes	No	100	EOD, RT, Delayed	D	Yes
Stocker	No	Yes	13	Delayed	D	No
TA- Lib	No	No	125	No	D	No
Meta Stock	Yes	Yes	200	EOD, RT, Delayed	D	Yes
"Timing the Market"	**Yes**	**Yes**	**14**	**EOD, Delayed**	**OL**	**No**

Fig. 1: Comparison of software's (Courtesy: Wikipedia – Technical Analysis)

Terminology:

Charting: Drawing elaborate charts based on stock information stored in database.
Verification: Providing recommendations supporting the fundamental prediction
Built-in indicators: Various indicators used for studying trends like Dow, S&P 500 etc.
Data feed: How the data is extracted from web sources

i. EOD: End of day
ii. RT: Real-time
iii. Delayed: After a specific time interval

Online/Download: Whether the software is available online or is to be downloaded.
Alerts: Email, web, SMS services.

2. SYSTEM DESCRIPTION

2.1 System Block Diagram:

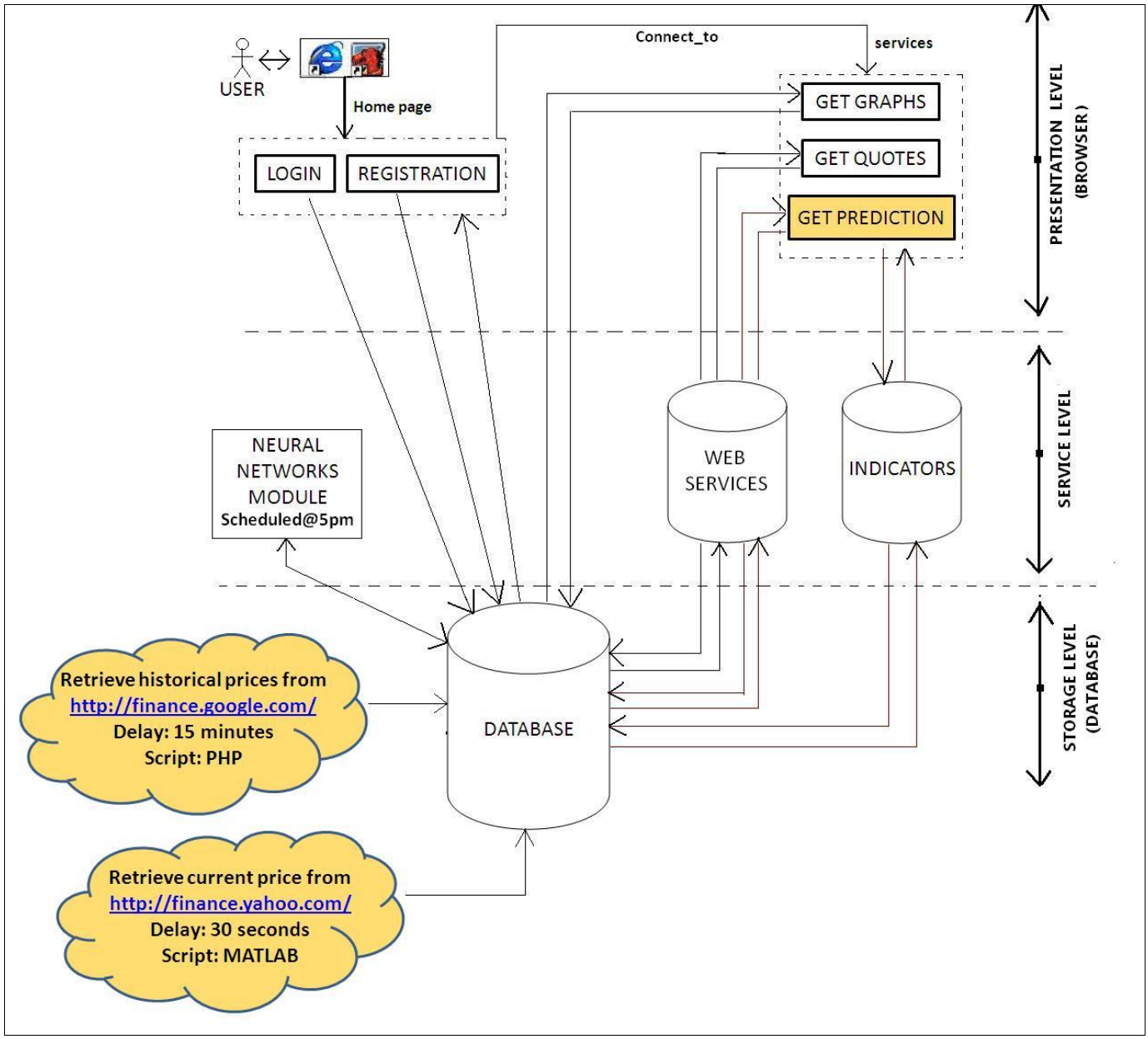


Fig 2: System Block Diagram

2.2 System description and general working:

As shown in Fig: 2, the system under consideration has three levels: presentation level, services level and storage level. The user gives the input in the form of the name of organization and desired service and gets the required output through the interface in the presentation level. Registration and authentication is required to get access to our services. The services level performs various actions on this data using Neural Networks and Indicator algorithms. Web Services are employed to provide connectivity between the prediction module and end-user. The storage level lies at the bottom of other levels with a database that stores and records the entire data.

The user is provided with the “home page” when he opens the application. The user is asked to provide a correct combination of username and password. This pair of username and password is sent to the database for verification. If the combination is correct, the user is directed to the “service page”. If the user is new to the system then he is allowed to register as a new user. The user selects the “registration” button and is then allowed to fill in the registration form, when this is done, the corresponding information is sent and stored into the database and the user has now got the username and password to login into the system.

There are three key options that a user can select according to his requirement. These are:

- Get Quote
  This service gets the current price for a particular stock from the database. These prices are ticked every 30 seconds from http://finance.yahoo.com using MATLAB.
- Get Graphs
  These graphs are dynamically generated using JPGraph software using historical prices gathered by MATLAB at the end-of-day.
- Get Prediction
  This command gets buy-sell recommendations for users using Web services or PHP scripts using Indicator algorithms, as well as predicted price for a particular stock.

Once the user enters the home page, he can select a stock from the drop down list and then select one of these options accordingly.

If a user wants to see the graph of a particular stock, the information is directly sent to the database, corresponding data is fetched, required actions are performed on it and the required graph is sent back to the top layer that is the interface. If the user wants to get the quotes of a particular stock, he selects that stock and clicks the “quotes” button. In this case the Web Service is called which takes the data from the data base, performs the required actions and send the required data to the user.

If the user wants to get the prediction of a stock, the user again selects the stock from the list, clicks the “Prediction” button. There are two ways in which the system works: one is using the Web Services and the other one is using the Indicators which can be invoked by the user.

Based on the method selected (Web Services or the Indicators), data is again fetched from the database and the desired results are sent back to the user through the services level.

The database is kept up to date with the latest stock data. This is because the PhP data mining script has been scheduled to run every 15 minutes. After every 15 minutes, new data is updated into the database. Google Finance (http://finance.google.com) is being used for fetching the historical data for our system.

In the above figure the top cloud represents the data mining application of our project where our php script get historical stock data which includes the following: High, Low, Close, Open and volume for a list of companies and stores it in the database (MySQL). The cloud at the bottom represents a Matlab code that connects to Yahoo Finance and gets the current stock price for a Company.

Implementation of Indicators:

The indicators are implemented on the client side. They are a part of our application since they provide additional support to the Neural Network based prediction model. They are mainly operation on data over a 9-25 days range. This gives us an indication that they are good options available for a very short term investor or even a day to day trader.

They do not call the web services. A PHP code has been written to implement the logic of all the indicators. The PhP code is embedded in the html pages. But do keep in mind that since the Indicators also need access to the historical data in the database they do connect directly to the database (MySQL). We followed this model because the indicators we short term hints for trading and felt that it would be a right mix to have a few things implemented in web service and few directly on the client side. It makes our entire application more dynamic and light weight. It was also important to us to do a few things differently since learning different things was quite exciting and important to us.

2.3 Use Cases:

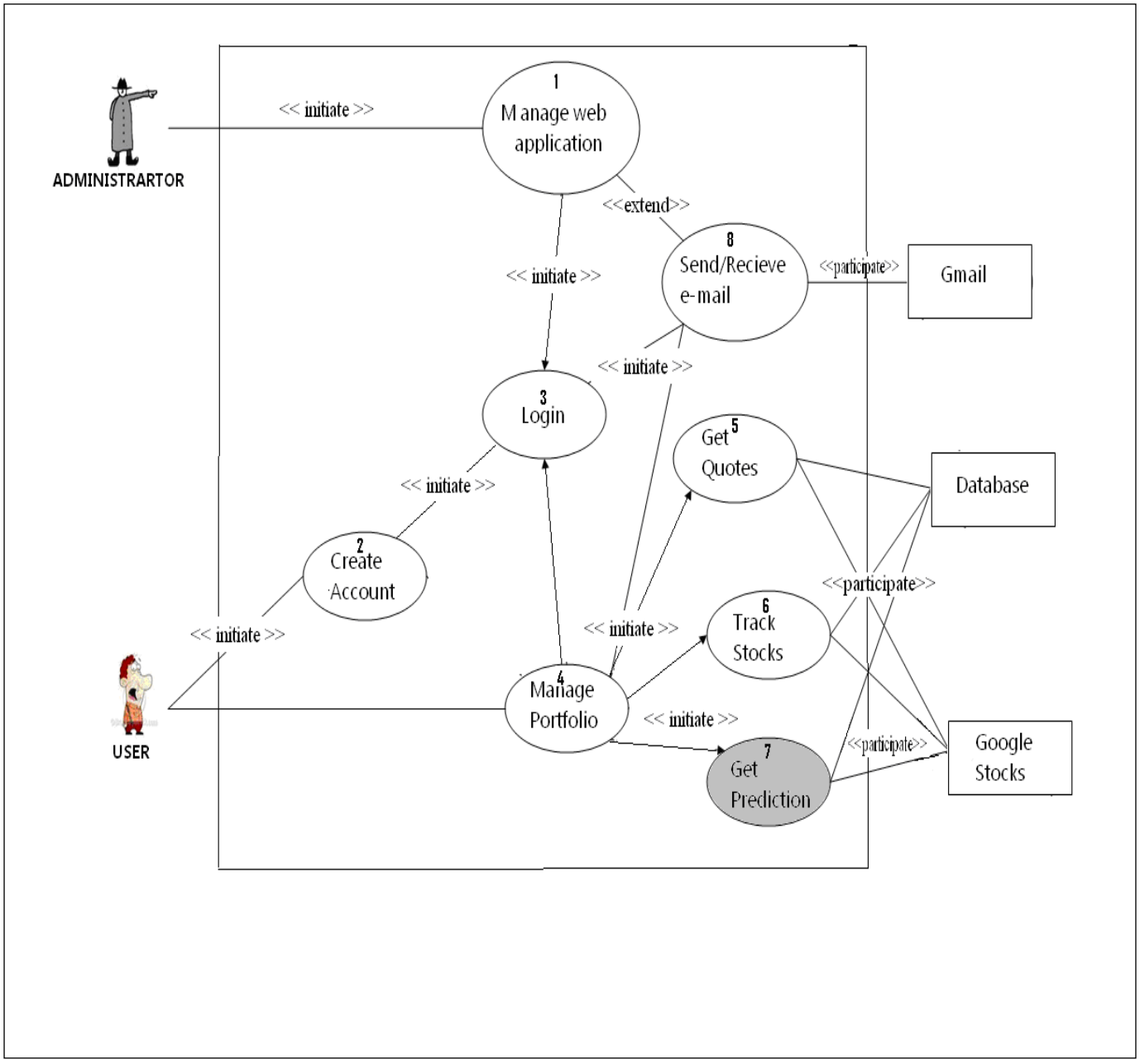


Fig3: Use Case Diagram

Use Case Description:

2.3.1 Use Case Casual Description

USE CASE	USE CASE NAME	ACTIVE PARTICIPANTS	DESCRIPTION
1	Web-Service management	Administrator	The Administrator will be maintaining and managing the web-service and the server
2	Create Account	User	The user will be able to create a new account if he is a new user
3	Login	User/ Administrator	By providing the correct combination of user name and password any user or the administrator is allowed to login to his respective accounts
4	Account/Portfolio management	User	This will allow a user to manage his account, check details of his account and is allowed to perform several other functions
5	Get Quotes/Graphs	User	The User will be allowed to get quotes/graphs for selected stocks
6	Track Stocks	User	The user will be allowed to track selected stocks for a specified amount of time before actually buying that stock
7	Get Stock Prediction	User	The user will be allowed to get the prediction of selected stocks
8	Send/Receive e-mails	User/ Administrator	Both the user and the administrator will be allowed to send and receive e-mails whenever required.

2.3.2 Functional Specification Requirement:

ACTOR	ACTOR GOALS	USE CASE
ADMINISTRATOR	To Login into the Admin account to view the application.	Use Case# 3
ADMINISTRATOR	Maintenance and Management of web service and the server	Use Case# 1
ADMINISTRATOR	To send/receive emails/opinions to/from the users	Use Case# 8
USER	To create a new account and login into his account	Use Case# 2
USER	To login into his account via the login page and see his account information	Use Case# 3
USER	To manage the account and check its details	Use Case# 4
USER	To get Quotes, Track Stocks, get Prediction	Use Case# 5, Use Case# 6,
GMAIL	To provide e-mail facility to the administrator and the user	Use Case# 8
DATABASE	To provide/store all necessary information regarding the stocks and the players	Use Case# 1, Use Case# 2, Use Case# 3, Use Case# 4, Use Case# 5, Use Case# 6, Use Case# 7, Use Case# 8

Note: The functionality provided by UC#6 and UC#8 could not be implemented due to time constraints.

USE CASE# 1 DESCRIPTION:

USE CASE# 1	Web-Service management
INITIATING ACTOR: Administrator	
ACTOR'S GOALS: Maintenance and management of web application and server	
PARTICIPATING ACTORS: None	
PRE CONDITIONS: The administrator should be properly authenticated by the system after he provides the correct user name and password	
POST CONDITIONS: Check the details of the application and make changes if required	
SUCCESS SCENARIO:	
1) Opens the Login Page	
2) Fills the correct Username	
3) Fills the correct Password	
4) Checks the details and make modifications if required	

USE CASE # 2 DESCRIPTION:

USE CASE# 2	Create Account
INITIATING ACTOR: User	
ACTOR'S GOALS: To create a new account	
PARTICIPATING ACTORS: Database	
PRE-CONDITIONS: The user does not have an account	
POST-CONDITIONS: The user has an account and can use the application as and when required	
SUCCESS SCENARIO:	
1) The user opens main page	
2) Selects new user option	
3) Fills in the registration form	
4) Select the create account option	

USE CASE # 3 DESCRIPTION:

USE CASE# 3 Login
INITIATING ACTOR: User/ Administrator
ACTOR'S GOALS: To login into the account
PARTICIPATING ACTORS: User / Administrator
PRE-CONDITIONS: Login status: not logged in
POST-CONDITIONS: Login status: logged in by giving correct username and password
SUCCESS SCENARIO: 1) The user/ administrator opens login page
2) Fills in the correct user name
3) Fills in the correct password
4) Select the Login option

USE CASE # 4 DESCRIPTION:

USE CASE# 4 Account/Portfolio Management
INITIATING ACTOR: User
ACTOR'S GOALS: To manage the account and check the details
PARTICIPATING ACTORS: Database
PRE-CONDITIONS: Login status: not logged in
POST-CONDITIONS: Login status: logged in by giving correct username and password and use the services
SUCCESS SCENARIO: 1) The user opens Login page
2) Fills in the correct user name and password
3) Check the details of the portfolio and make decisions accordingly

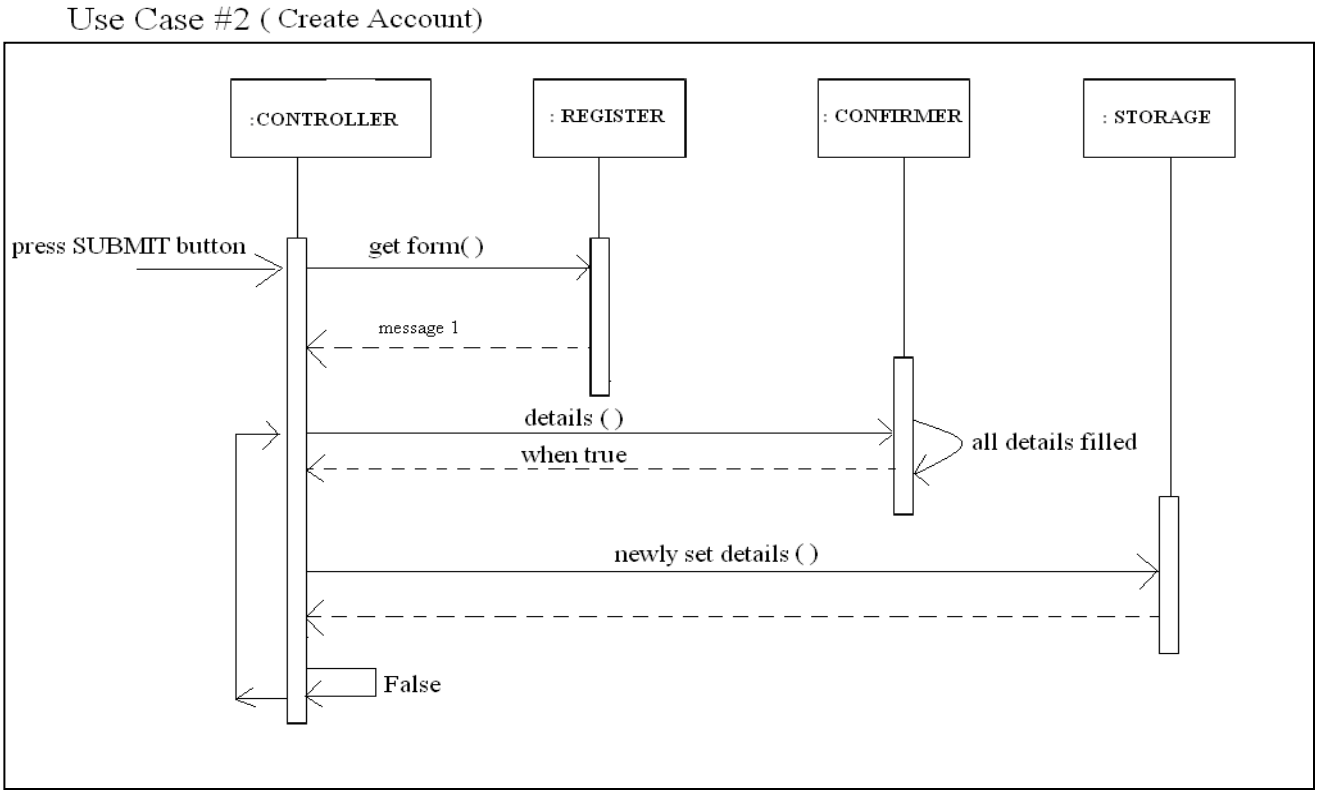
USE CASE # 5 DESCRIPTION:

USE CASE# 5 Get Quotes/Graphs
INITIATING ACTOR: User
ACTOR'S GOALS: To get the quotes and graphs of selected stocks
PARTICIPATING ACTORS: None
PRE-CONDITIONS: Login status: user has logged in
POST-CONDITIONS: Get the required quotes
SUCCESS SCENARIO: 1) The user opens the Login page
2) Fills in the correct username and password
3) Select the Company for which he wants the quotes/ graphs
4) Select the get quoted option 5) Check the quotes and the graphs obtained

USE CASE # 7 DESCRIPTION:

USE CASE# 7 Get Stock Prediction
INITIATING ACTOR: User
ACTOR'S GOALS: To get the prediction of selected stocks
PARTICIPATING ACTORS: Database
PRE-CONDITIONS: Login status: user has logged in
POST-CONDITIONS: Gets the required prediction (Buy, Sell or Hold the stock)
SUCCESS SCENARIO: 1) The user opens the Login page
2) Fills in the correct username and password
3) Select the Company for which he wants the prediction
4) Select the get prediction option 5) Check the predictions obtained and make decisions accordingly

2.3.3 Interaction Diagrams of key Use Cases:



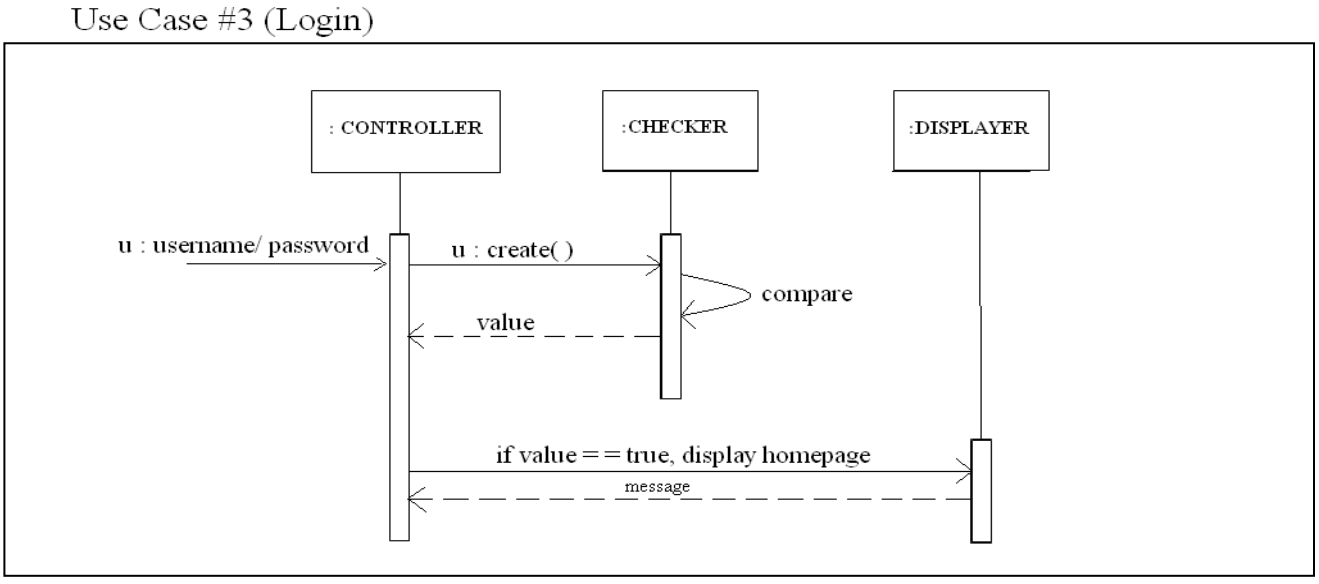


Fig 4: Use Case #2 and #3 Interaction Diagram

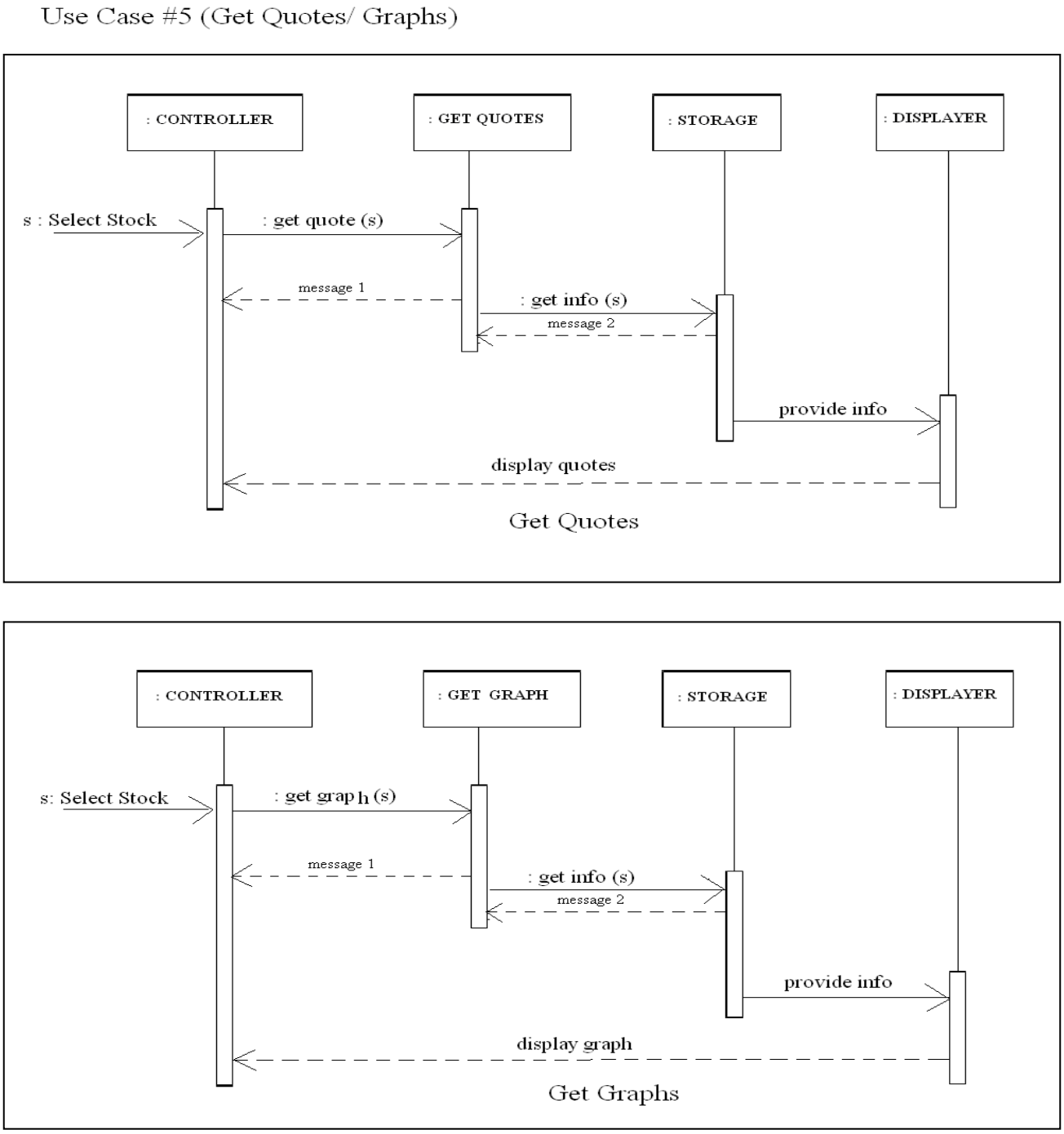


Fig 5: Use Case #5 Interaction Diagram

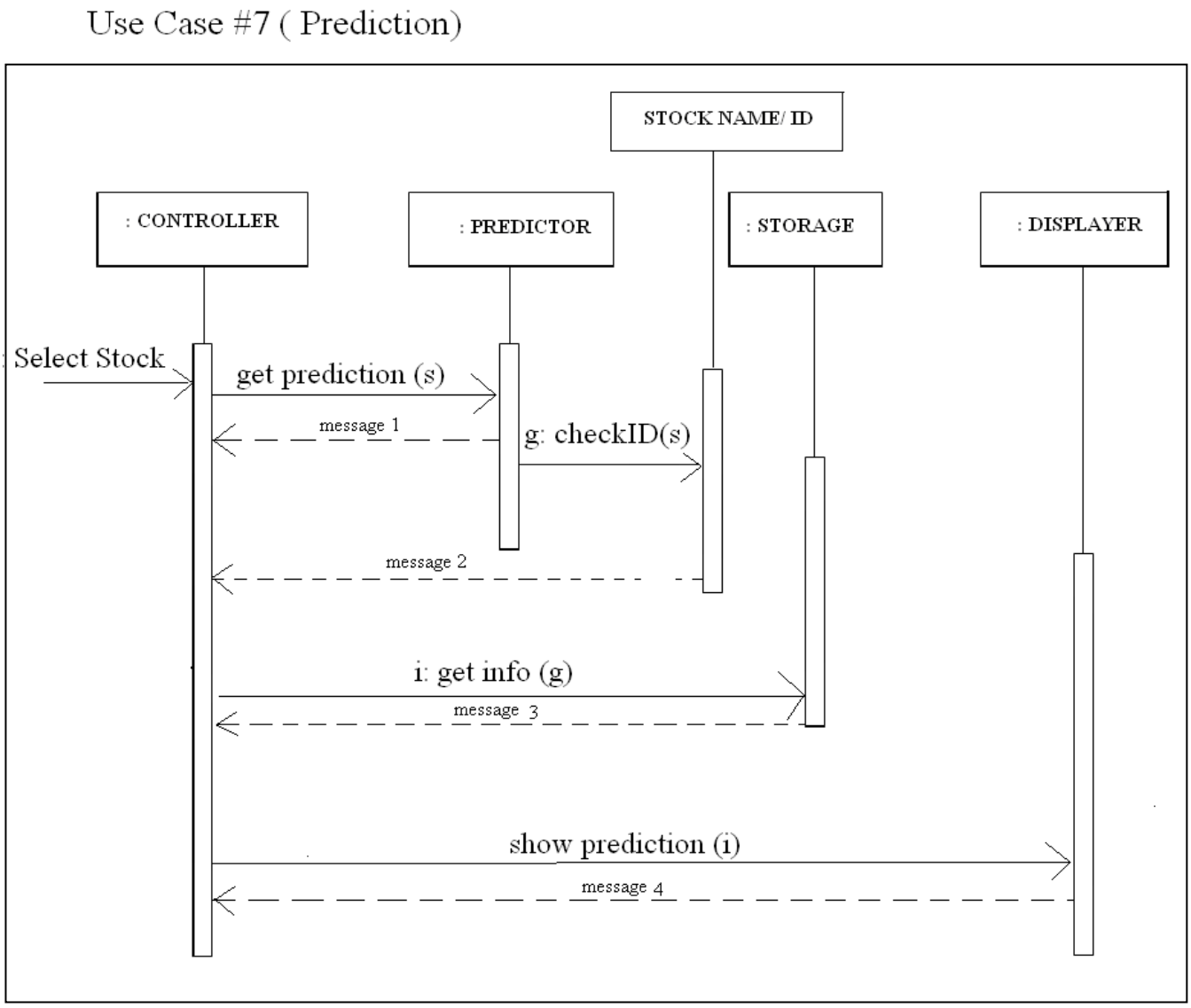


Fig 6: Use Case #7 Interaction Diagram

Notes: The diagram starts from the controller. The controller gets a particular stock(s) and passes it to the Predictor. The predictor may call the web services or use the indicators to get the prediction based on the unique name and ID of the stock. Data is fetched from the storage, calculations are performed and finally again sent back to the storage and the final result: buy or sell are sent to the displayer which displays it on the user interface.

2.4 Activity Diagrams:

Use Case #1 (Manage Account):

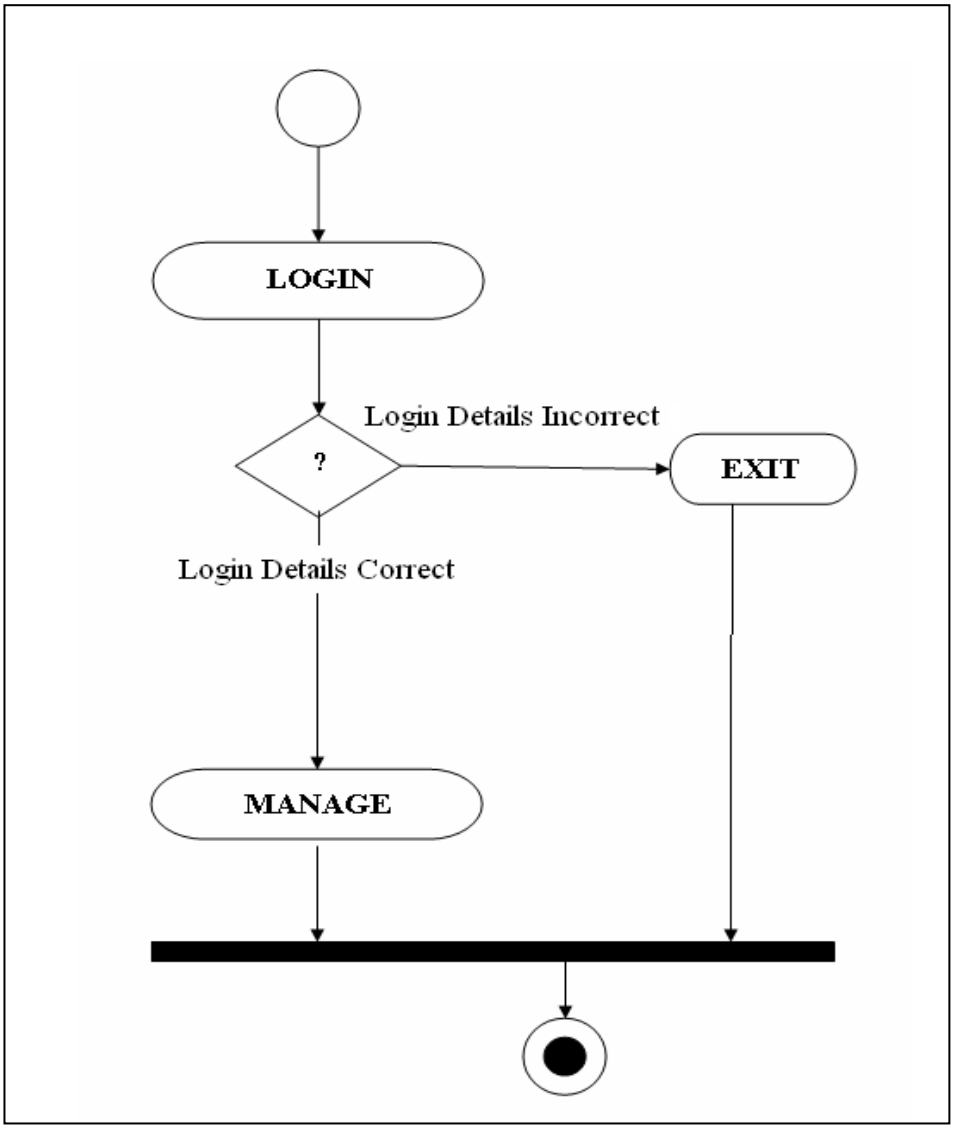


Fig 7: Use Case #1 Activity Diagram (Manage Account)

Steps:

a. Administrator enters into the home page.
b. Fills in the user name and password.
c. If the username/password is correct, he enters into service page and can manage and maintain it.

Use Case #2 (Create Account):

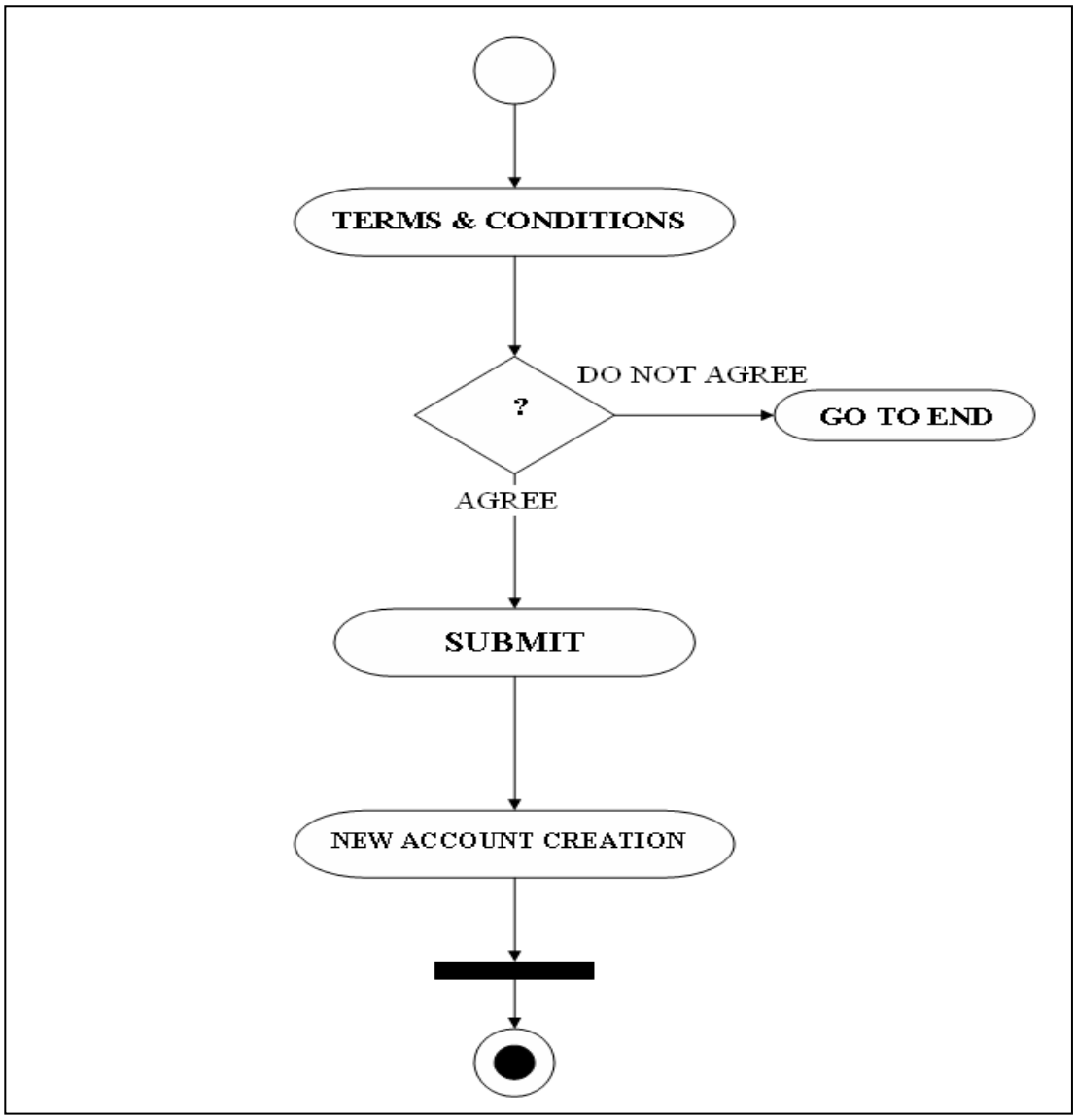


Fig 8: Use Case #2 Activity Diagram

Steps:

a. User enters into the registration page.
b. Gets the terms and conditions.
c. If agreed, he fills the form.
d. Press the submit button and gets registered as a new user.

Use Case #3 (Login):

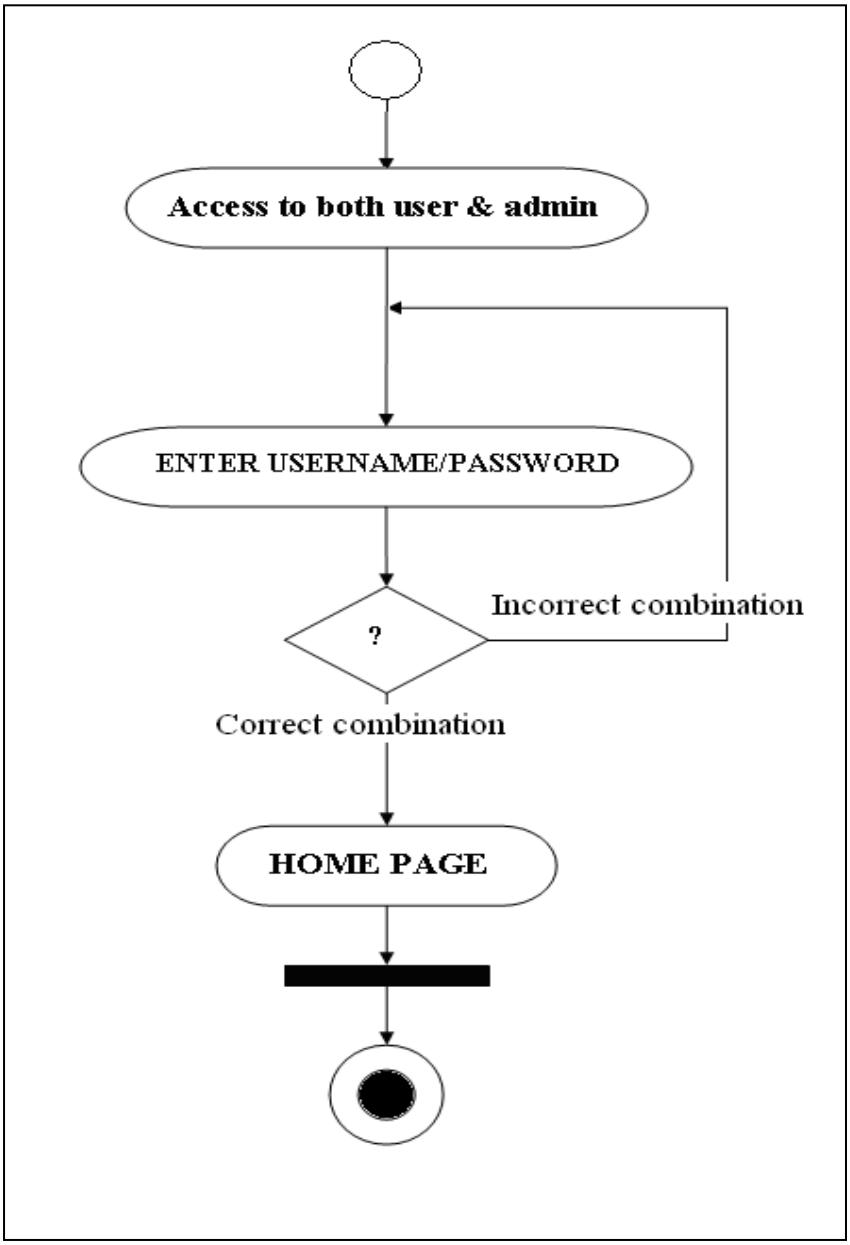


Fig 9: Use Case #3 Activity Diagram

Steps:

a. The user/administrator is in home page.
b. The user provides the username/password.
c. If combination is correct, service page is displayed.
d. If combination is incorrect, the user is allowed to re-enter the username/password.

Use Case #5 (Get Quotes/ Graphs):

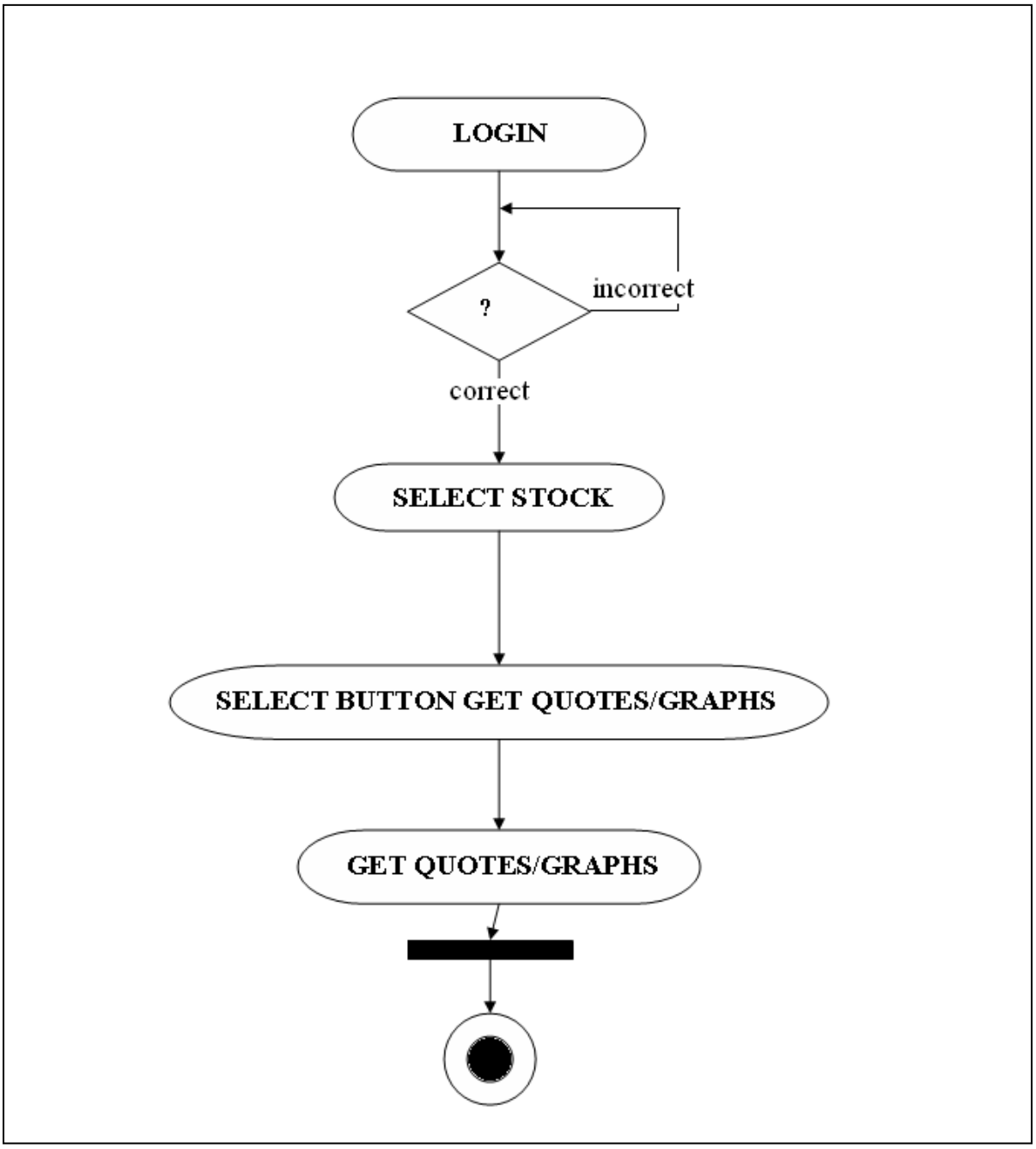


Fig 10: Use Case #5 Activity Diagram

Steps:

a. The user after successful login can select any stock from the drop down box
b. Click the Get Quotes/ Get Graphs button
c. The corresponding Quotes/ Graphs are displayed.

Use Case #7 (Get Prediction):

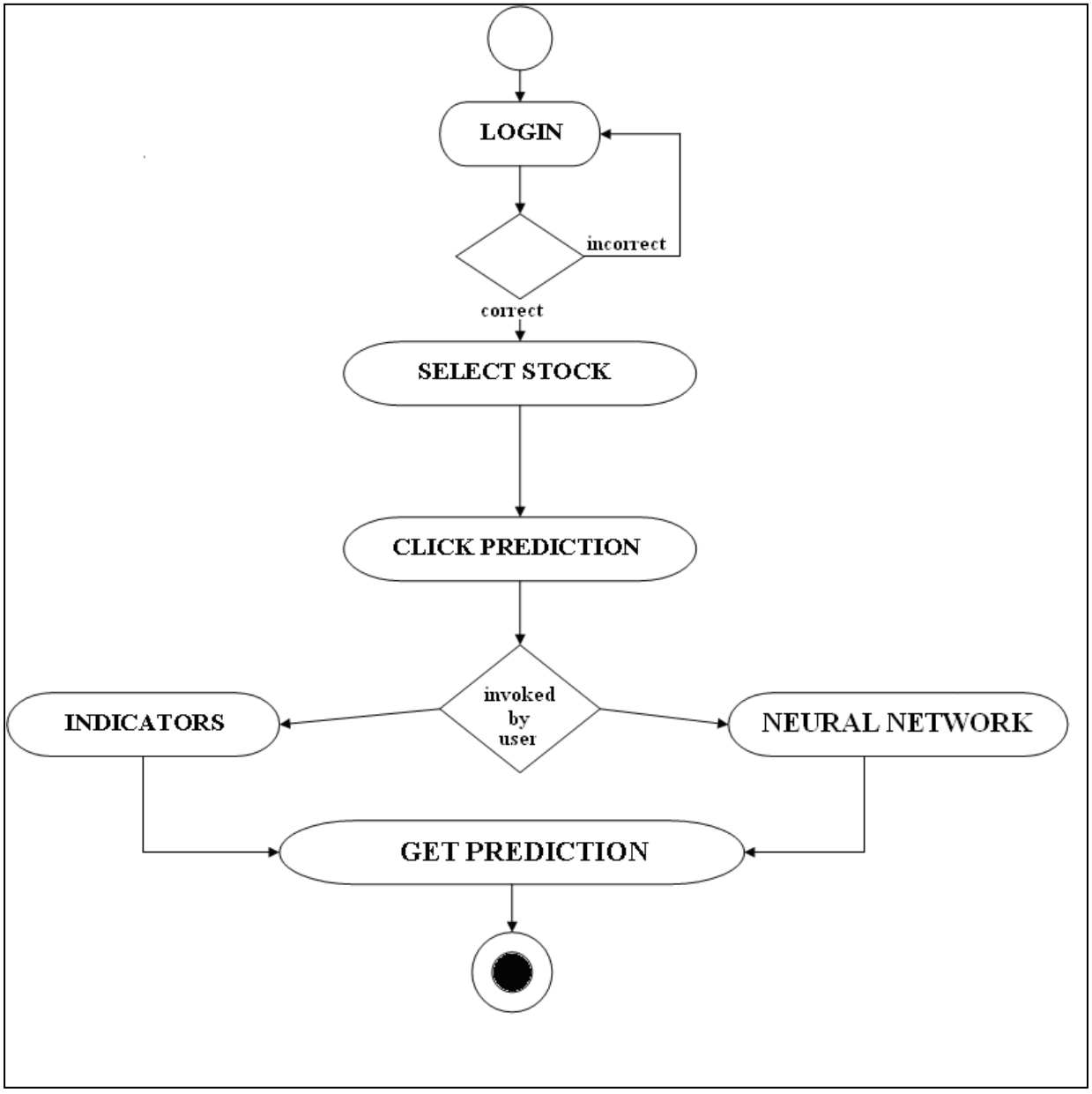


Fig 11: Use case #7 Activity Diagram

Steps:

a. User logins and select a stock from the drop down box
b. The user can get the prediction of the selected stock by clicking Get Prediction. In this case Web Services will be called to provide the prediction, or the user can also invoke the indicators to get same prediction and confirm the result.
c. The prediction is displayed.

2.5 System Class Diagram:

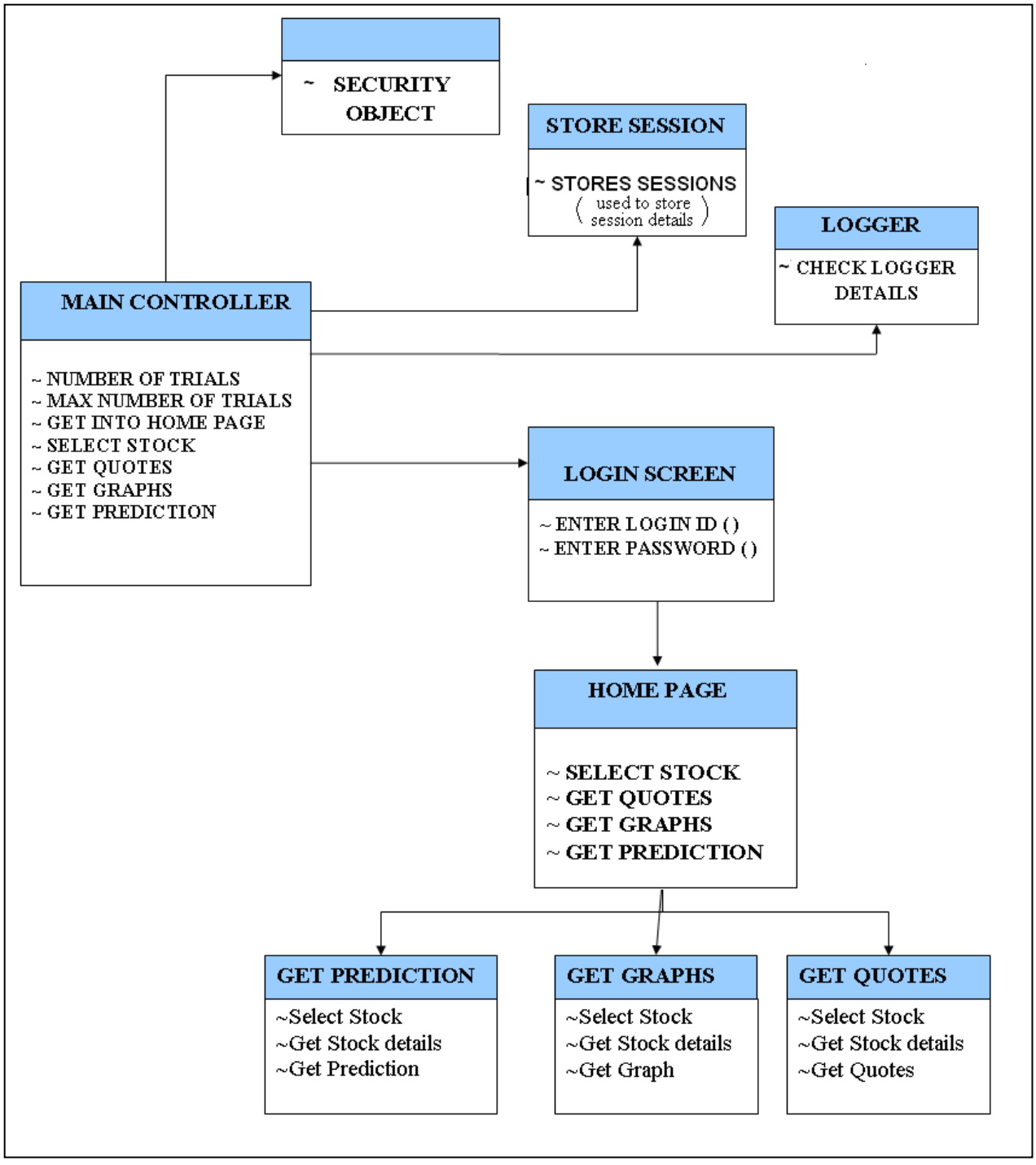


Fig 12: System Class Diagram

2.5.1 Class Diagram Description:

NAME	DOCUMENTATION
Main controller	Main controller class controls all the activities of the system. All the classes connected to operations to be done by the user, like logging in, opening WebPages and Accessing them through this control.
Security object	The security object checks the login details and password of the user.
Login screen	The screen which appears to the user before log in the Details. It will prompt the user to enter the details, which is Checked by security object class.
Home Page	It contains a list of all the company names. The user can select any stock from the drop down box and proceed as Required.
Session store	It will store the session details of the user, every time he logs in
Logger	It logs the details of admin, every time he access the pages.

2.5.2 Class Diagram Attributes and their Description:

Main Controller	
Number of trials	It describes about the number of trials the user can make in login page, while entering the password.
Max number of trials	It describes the maximum number of trails the user can make before getting his system locked.
Get info from Home page	It provides information about the system
Get into Home page Select Stock	It allows to use the home page as required It allows to select the required stock from the list of available stocks
Get Quotes	It provides quotes of the stock selected
Get Graphs Get Prediction	It provides graphs of the stock selected It provides the prediction of the stock selected

Login Screen	
enter loginID	This attribute prompts the user to enter login details.
enter password	This attribute prompts the user to enter password details.

Security Object	
check login	This attribute checks the user entered login
check password	This attribute checks the user entered password

Session store	
Store session	This attribute stores the session

Logger	
log transactions	This attribute logs all the transactions

Services Page	
Select Stock	This attribute is used to select a stock from a given list of stocks
Get Quotes	This attribute is used to get the quotes of a selected stock
Get Graphs	This attribute is used to get the quotes of a selected stock
Get Prediction	This attribute is used to get the quotes of a selected stock

2.DATA MINING

3.1 Need for data mining:

Data mining is the process of sorting out large amounts of data and picking up relevant information. We need to collect data for the following reasons:

1. The project goal of predicting stocks requires huge amount of stock data for each company, for training our neural network model.

2. The data collected needs to be updated at regular intervals to make our system robust and accurate. What update means in this context is that some of the fields of the database like the volume, high and low stock price can change a number of times in a day. The client should thus get the result for his required indicators (which usually do perform certain calculations based on the volume of shares traded and the high and low stock price) bases on the variations in one particular day.

   When updating it does not clear all the previous days' data. It just checks if it has yesterday's historical data is present in the database and only updates the latest values for the current day. This takes place intraday. Now if I was unable to run the code for a complete one day and start running the script the next day. Then I would definitely need data of the day I had skipped. So the code now while iterating through the dates finds a date missing then it automatically inserts the value into the database. Thus it never so happens that a day is missed because we forgot to run our script on a particular day.

3. We have also implemented indicators for the sake of short term investors and hence they require historical data of a particular security, to get accurate stock price indication.

4. Our client web portal is 'Login' based hence we need to perform user authentication.

5. Intermediate results have to be stored, in order to systematically represent the results on the web pages.

Thus it is evident that a relational database would satisfy all our needs, for the same we have developed powerful script to import data from the World Wide Web.

3.2 Data Mining - An overview:

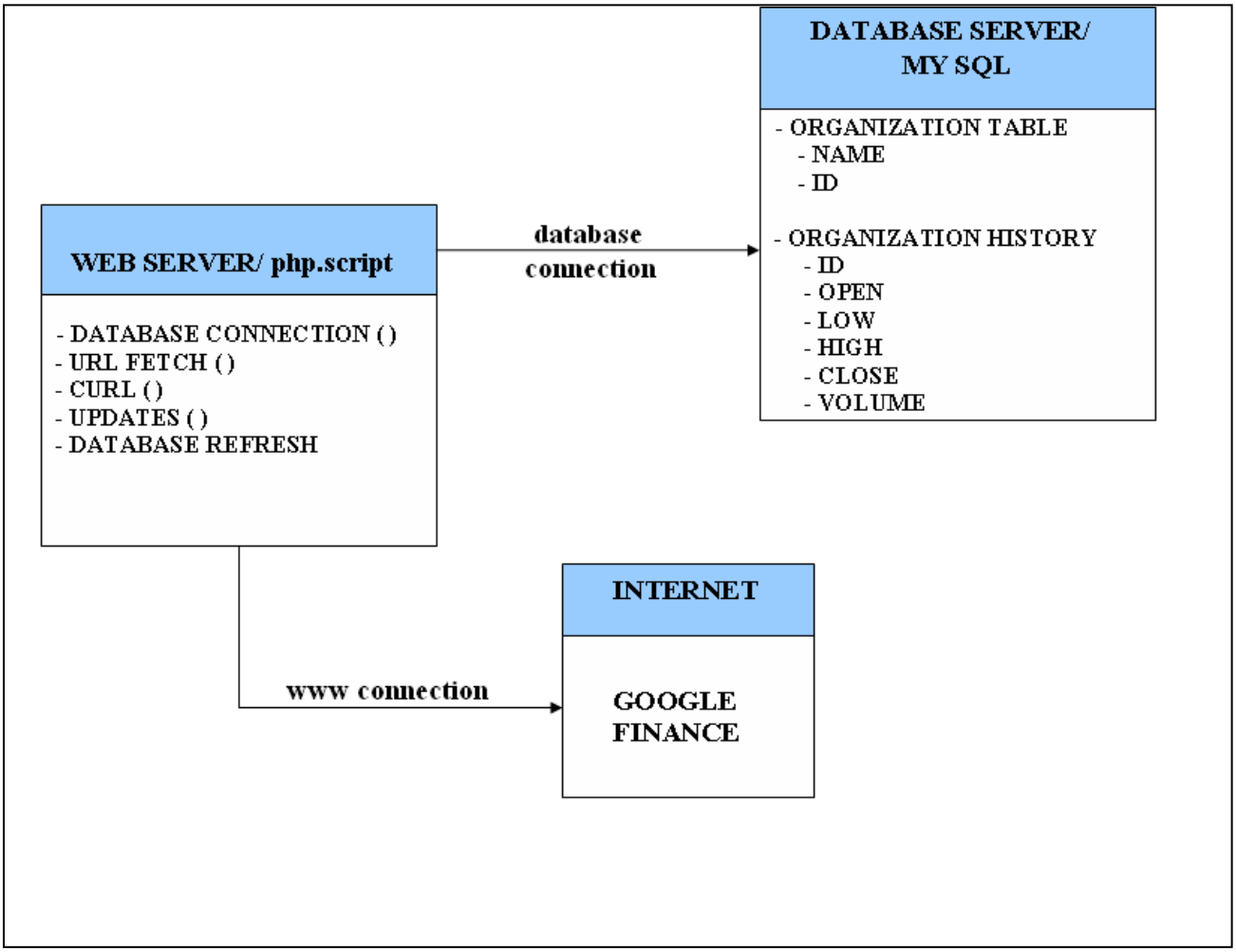


Fig 13: Class Diagram of Data Mining

Database Connection: The PhP script has to connect to the database so that it can store the historical data which it gets from the World Wide Web (Google Finance). This there is a need for a connection to be established.

URL Fetch: This is basically a part of the data mining PhP code which basically is able to dynamically create the URLs of Google finance. So that we can use those URLs to

import the data from the Google website and place it in a text file. This text file is then read by the PhP code and data retrieved is stored into the database for each of the organization

CURL: PHP has a very powerful library of calls that are specifically designed to safely fetch data from remote sites. It's called CURL. This library, usually included with PHP, allows you to retrieve other pages, and also makes it possible to define dozens of different options.

Updates: The refresh option in the PhP code brings in the latest historical data. This data has to be updated in the database. Hence the update method of the PhP code does the same.

Database Refresh: This is the ability of the data mining PhP code to execute after every 15 minutes.

Class Diagram Description:

NAME	DOCUMENTATION
WEB SERVER	Internet Information Services(IIS) Server is the web server which hosts our application.
DATABASE SERVER	We have a MySQL database server
INTERNET SERVER	IIS

The above diagram gives an overview of how our data is extracted from the World Wide Web and stored into our database.

3.3 Features of the Fetch script:

- The PhP script to import data from the Google Finance website is located on the webserver and it connects to the MySQL database. It uses the Curl library functions and hence the code behaves like a web spider and imports the data from the mentioned website and dumps the data into a comma separated file (.csv).

- The file imported from the website has data for each organization for a period of one year. The data includes Open, High, Low, Close Prices and Volume of stocks.

- The most dynamic feature of the php script is that there is no need for manual entry of data at any point of time. The script has the ability to dynamically create URLs from where it needs to pull data from. And hence it is very fast, efficient and portable.

  PhP has a curl() library which can import the data from the web and put it in a CSV file. The curl function takes the input parameter as the URL from where the data has to be fetched. Now we need historical data for all the organizations in a single table. And the script has to be completely automatic.

  The Google Finance URL for a particular company, say CISCO looks something like this.

  http://finance.google.com/finance/historical?cid=99624&startdate=May+17%2C+2007&enddate=May+15%2C+2008&output=csv

  One can notice that the URL has the *startdate* and *enddate* mentioned in the URL. Hence I can change the dates as per the date range I require. Plus if we look carefully then there is a parameter called *cid*. This cid is a unique ID allotted to each organization by Google Finance. Now we are also using the same ID as the primary key in our master table (organizationtable).

  Thus what my PhP code does is it iterates through the entire IDs which it retrieves from the master table and creates a different URL for each company based on the ID I have in the master table which also matches the ones that Google finance uses. Thus I do not need to hardcode anything in the PhP script. These URLs are then passed to the CURL function as an input parameter, which

returns the entire historical data for one company in a CSV file. This data is then read by the PhP code and stored in the database.

Note this entire operation takes place in a single loop and thus I shall only require one text file at a time. This saves server space as well. Thus creating URLs for importing data from google finance is efficient and helpful.

- We have included an automatic refresh capability in the script. Since some features in our application, mainly our indicators which are explained later in the text require current prices of the stock and also the current volume. Hence we have provided with an auto refresh function. The current volume is also obtained from google finance. It is a part of the historical data which the PhP script imports.

- The PhP code also takes into consideration weekends and hence does not run the script or try and update the database.

3.4 Current Price Fetch:

One of the services provided to the user is **get quotes**. The user can select one of the 20 companies provided in the drop down box and request for the current price for that particular stock. When the client requests for the current price of a company, the getquotes () web method of the ASP.Net web services is called which returns the requested current price from the MySQL Database.

The data mining PHP script brings in historical values for the companies but it does not provide the current price. Hence we use Matlab to fetch this current price. We use an object “connect” to connect to Yahoo (URL: *'http://quote.yahoo.com'*) and a timer is set which allows the code to run every 30 seconds. Now fetch() is used to bring current prices for all the companies which are refreshed every 30 seconds. Matlab then connects to the database and updates the table (current price column with its corresponding price). Since we need the current price of a particular company with minimum delay as compared to the real world value, thus we are updating the Matlab code which gets the current stock price every 30 seconds. We must keep in mind that the historical data is updated every 15 minutes only.

3.5 DB Schema Design

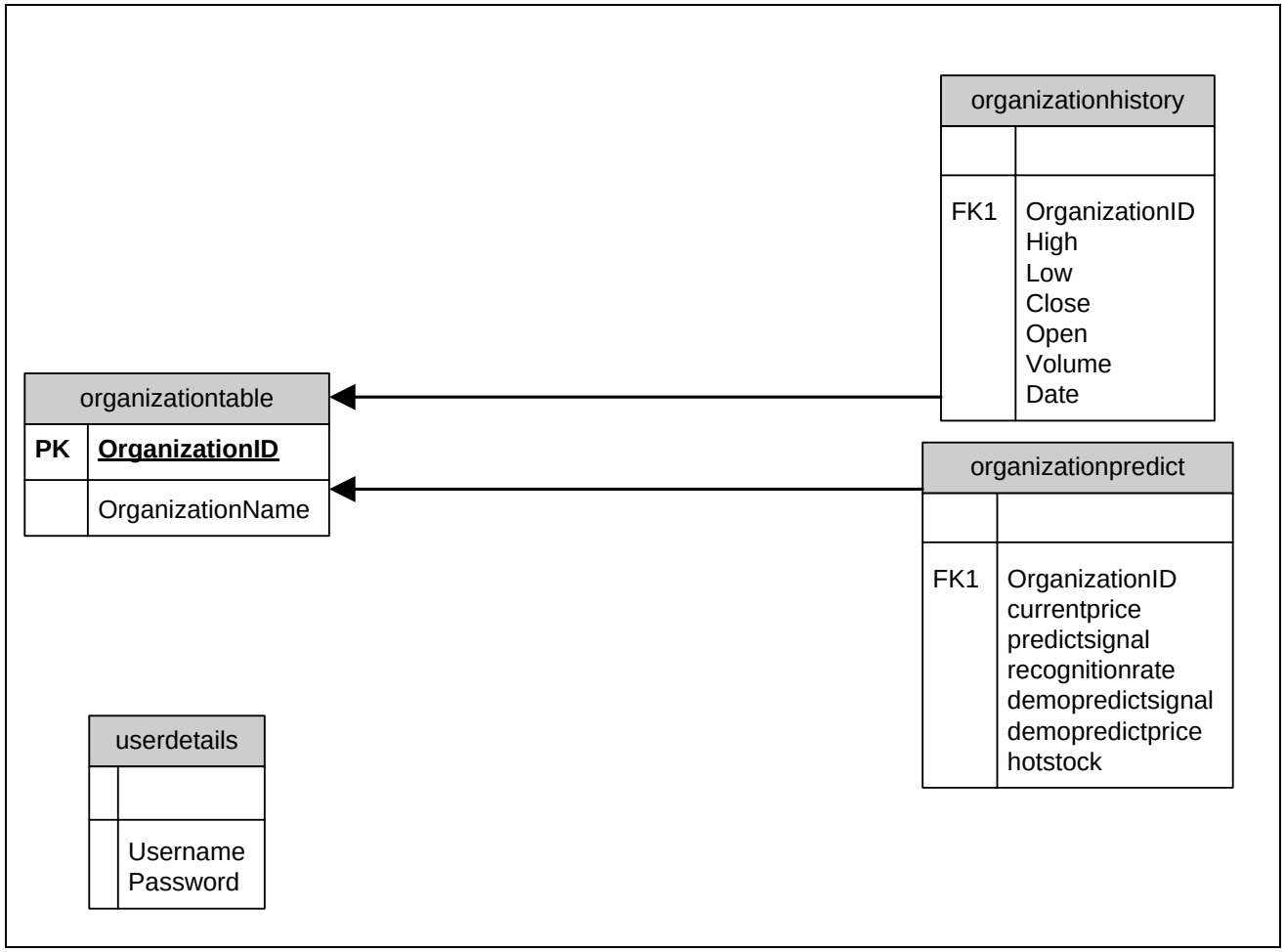


Fig 14: Database Schema

The database schema provided above is a complete snapshot of MySQL tables along with the relationships. To predict stocks it was necessary to import historical stock data which can be processed by our neural network prediction model.

The ***organizationtable*** is our parent table which has the list of all the companies for which we are going to predict stocks. Each entry in the table is unique and hence linked to a primary key. This primary key is the same as the ID allotted to the particular company by Google Finance.

The PhP script dumps the data into the ***organizationhistory*** table which is shown above in the schema design diagram. The fields of the table are self explanatory.

The ***organizationpredict*** table contains 5 fields. The OrganizationID is mapped to the OrganizationID in the master table which also has the name of the organization. This table is updated by the Matlab code which implements the Neural Network stock prediction. This table is then accessed by our asp.Net web services when the client requests for a particular service

The field *currentprice* stores the current price of the particular stock which the MATLAB code gets from Yahoo Finance. The *predictsignal* field stores a 1(indicating a "buy") and 0(indicating a "sell"). The *recognitionrate* provides performance evaluation for each company. This is explained in section 5.5. The demopredictprice and demopredictsignal fields were created for the convenience of verifying our system prediction during the demonstration(explained in detail in section 5.4 step 9.). The hotstock field contains the OrganizationID of a single company (explained in detail in section 5.4 step 8.)

3. WEB SERVICES

Web services are a standardized way for one application to invoke a method of another application. These applications may reside on different computers that are some way connected, such as by a local area network or, more commonly, by the Internet. With Web services you have two *actors*, a client and a server. The server makes some set of methods available. These methods, when called, will perform some action and/or return some data. The client invokes one of the available methods of the server. Like methods in a class library, the methods of a Web service can accept an arbitrary number of input parameters, and can optionally return a result.

The fundamental properties of any service are:

- Medium by which the service is performed,
- The protocol by which the messages are sent to and from the service, and
- The protocol for the message formats.

With Web services, the medium used is a computer network, such as the Internet or a LAN. The messages are typically transported from a client to a server using the HTTP protocol. HTTP, which stands for **H**yper**T**ext **T**ransfer **P**rotocol, is the protocol used to request/receive Web pages from a Web server to a browser. Web services can use other transport protocols, however, such as SMTP, but HTTP is used in the vast majority of cases. The message protocol used by Web services is SOAP. SOAP, or **S**imple **O**bject **A**ccess **P**rotocol, is an XML-encoded messaging protocol that is used to marshal the input parameters and return values of a Web service method.

4.1 Procedure for using Web services:

Web services transport their messages using HTTP, which means these messages are transmitted over port 80, an open port for Web server firewalls. Web service messages are transmitted as SOAP-formatted messages. SOAP messages are in XML format, meaning they are simply text, and not complex binary data. SOAP is designed to serialize both simple and complex data types, meaning SOAP can be used to easily transmit complex data from the client to the server and back again. Finally, the protocols used by Web services - SOAP and HTTP - are open and well-known protocols, meaning Web services can be easily implemented on any platform. In fact, one of the great benefits of Web services is their interoperability. A Web service created with the J2EE platform can be consumed by a client created with the .NET platform, and vice-a-versa.

Due to their interoperability, in enterprises Web services are commonly used as a unified means to access disparate systems. For example, a large enterprise may have data in a variety of systems, accessible through various platforms. HR data might be stored in an Oracle database and accessed via a J2EE application; sales data may be in a MS-SQL Server database accessed by a VB6 application; employee data may be in a DB2 database and accessed by a legacy software system. Clearly, it would be advantageous to be able to access all of this data uniformly from a single application. One option would be to rewrite all of these old applications so that a common data store and access program is used. This approach is undesirable for a number of reasons, such as the cost and time it would take to replace the legacy software with new software. Rather, it might make more sense to provide Web service "front-ends" to each of these data stores and application interfaces. Then, a new application can uniformly access data from disparate resources all through a simple, standard Web service interface.

4.1.1 A High-Level View of Web Services:

Web services provide a means for a client application to invoke a service of a server application. The challenges surrounding this process are as follows:

- How is the service invoked? That is, how does the client specify the Web service method he wished to call?
- How are the input parameters serialized? That is, if the input parameters involve complex data types, how are these data types represented in the message?
- How is the method's response serialized?

The answers to these questions are spelled out in the SOAP specification. SOAP is the message protocol used for Web services. A SOAP message is an XML-formatted message. The figure below shows a high-level view of the interaction between a client and a Web service. That is, a client begins the interaction by sending an XML-formatted message to the server specifying the function he wants to invoke, along with the input parameters. Upon receiving this information, the server invokes the specified method, and then returns the methods result in an XML-formatted message.

This simple interaction between a client and a server sums up what Web services are in a concise manner.

4.1.2 Consuming a Web Service:

In order for a client to be able to consume a Web service, the client must know a bit about the Web service - namely, it must know where the Web service "lives" and how to invoke its methods (by knowing how to serialize and de-serialize the messages it passes to and receives from the Web service). This information is contained wholly in the Web service's WSDL document. In addition to knowing how to invoke the Web service's methods, the client must also go through the motions involved in creating the SOAP message to send to the Web service, making the actual HTTP request, and then deserializing the HTTP response. These steps will need to be repeated for each Web service a client wishes to consume.

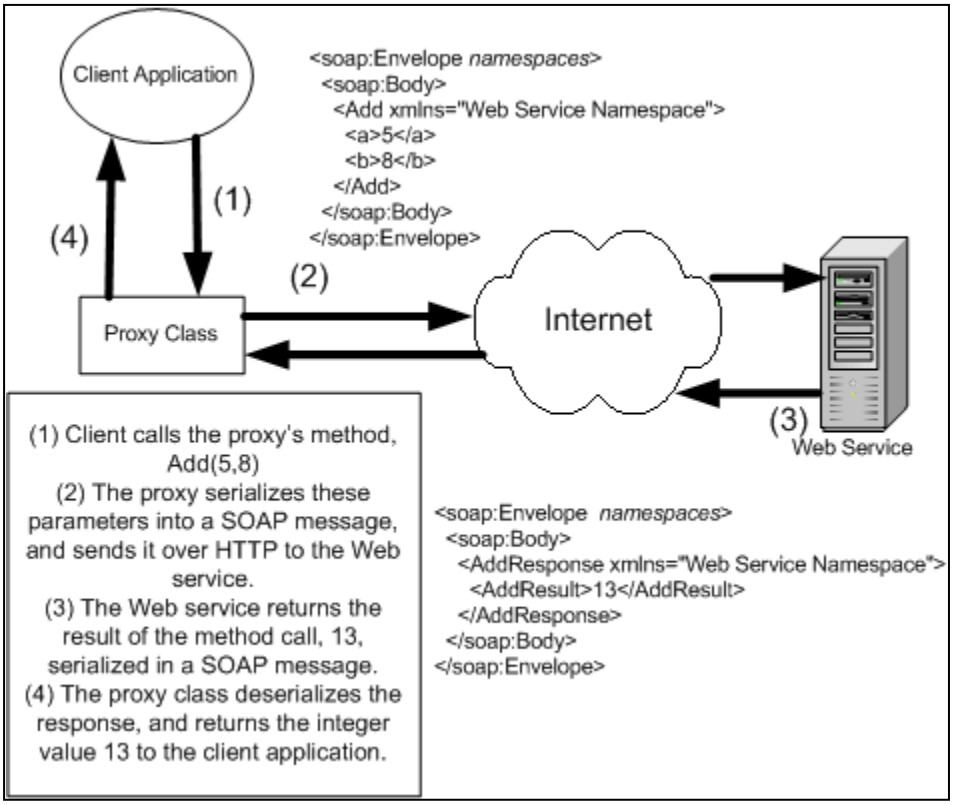


Fig 15 : Basic Web Service Description

A proxy class is a class that encapsulates the complexity of calling a Web service and exposes this complexity through a simplified interface. From the client application's perspective, the Web service is simply a local component - the client doesn't have to worry about the specifics of how to serialize a SOAP message, or how to make an HTTP request. In order to create the proxy class, it is important that you have access to the Web service's WSDL document.

4.1.3 Benefits of using ASP.net for creating Web Services:

ASP.NET allows you to build and publish Web services using familiar programming construct, such as methods, primitive types, and user-defined complex types. The .NET Framework also contains infrastructure and tools to create Web service clients that can call any standards-compliant Web service.

With ASP.NET, you can build Web services that use industry-wide standards for implementation. Since Web services are built on industry-wide standards, they can be communicated with across the Web using any client on any platform that adheres to these standards. Specifically, Web services employ the industry standards listed in the following table.

Industry standard	**Use in Web services created using ASP.NET**
XML	The text format used when communicating with Web services using the SOAP protocol. When communicating with Web services using the HTTP-GET and HTTP-POST protocols, XML is used to encode responses.
SOAP	An XML-based message exchange protocol used for communication between Web services and their clients.
Web Services Description Language (WSDL)	Describes the contract of messages that a Web service can interpret when communicating with a Web service client.
XSD	Provides a universal type system, allowing data types to be defined and passed across platforms. For a Web service, XSD defines the structure and data types for the XML encapsulated within a SOAP message sent to and from a Web service.
application/x-www-form-url encoded	A MIME type used for encoding parameters on a URL. This encoding is used for encoding request parameters to Web services using the HTTP-GET and HTTP-POST protocols.

With Web services you can take advantage of the features of ASP.NET to build Web services that adhere to industry-wide standards. Specifically, ASP.NET not only takes advantage of performance enhancements found in the .NET Framework and the common language runtime, it has also been designed to offer significant performance

improvements over ASP and other Web development platforms. All ASP.NET code is compiled, rather than interpreted, which allows early binding, strong typing, and just-in-time (JIT) compiling to native code, to name a few of its benefits. ASP.NET is also easily factorable, meaning that developers can remove modules (session modules, for instance) that are not relevant to the application they are developing.

4.2 Web Service inclusion in project at hand:

Since we are also building a client server web application model, we needed to separate certain functionalities based on programming technology implemented.

The application we have developed has a very powerful MATLAB based Neural Network Model. This model simulates the historical prices of stocks and provides the user with correct information of its future trends and price. Our goal was to make our prediction model accessible on the World Wide Web. As already explained before, our published ASP.NET web service allows user to connect to it through a LAN network or the internet.

Overview:

We created an ASP.NET web service using Visual Studio 2005. Our web service contains a single web service class which has a list of web methods. The web methods are based on the various different services the client requests. For example it could be:

1. Print Graphs.
2. Show current Stock Quotes.
3. Predict stock using Neural N/W Model.
4. Predict stocks for short term investment using various Indicators.

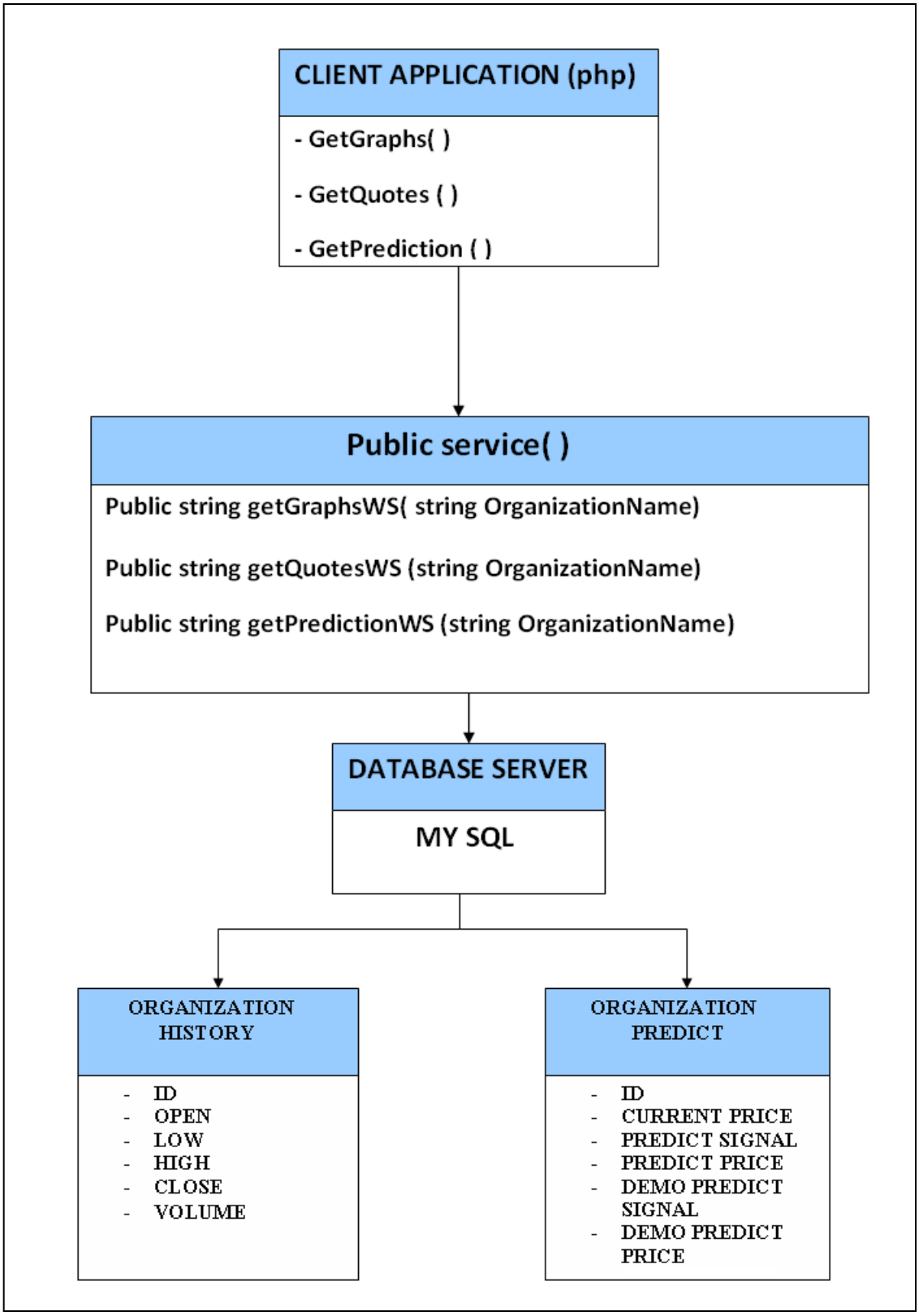


Fig 16: Class Diagram for Web Services

4.2.1 Detailed Explanations of our Web Service

We now list all the different web methods present in our Web Service class, giving a detailed explanation of each with the exact signature.

a. getQuotesWS()

The method signature is as given below:

```
[WebMethod]
public string getQuotesWS(string organization_name)
{}
```

Hence this web method has an input parameter which is the name of the Organization selected from the dropdown menu on the client php page. The return value which is of string type returns to current stock price for the selected organization. This value is fetched from the MySQL database using ODBC connections to the database by the C# code of the getQuotesWS method.

b. getPredictionWS()

The method signature is as given below:

```
[WebMethod]
public string getPredictionWS(string organization_name)
{}
```

This method too has an input parameter which is the name of the Organization selected by the user from the client page.

c. getgraphsWS()

The method signature is as given below:

```
[WebMethod]
public string[] getGraphsWS(string organization_name)
{}
```

This method also has the same Organization Name as the input parameter and returns the string array which contains the closing prices of the organization selected. This array of closing price is then used to create graphs for the users of the historical prices using php.

Thus the graph created will be Date Vs Closing Price for a selected organization. The data range for the Prices shown in the graph is over an entire period of one year. The closing price for the one year is returned to the client through the web service but the logic to plot the graph is on the client PhP page. Hence we can that the graphs are plotted on the fly.

We have used JPGraphs, which is an Object Oriented Graph creation library in PhP. This provides easy means to create simple and dirty graphs.

4.3 Understanding WSDL:

WSDL, or **W**eb **S**ervice **D**escription **L**anguage, is an XML-formatted document that formally defines the Web service. Specifically, the WSDL document spells out in precise terms:

- The data types of the messages a Web service will accept and send (these message types are spelled out using XML Schema)
- What message format is accepted (SOAP, HTTP-POST, HTTP-GET, etc.)
- For SOAP messages, the style and encoding of the SOAP message, and
- The Web service's endpoint (the URL the Web service resides at)

View of our Web Service Description

Below is a snapshot of the WSDL generated by our web service. It is followed by a SOAP request and response which is exchanged by the client application and the web server hosting the Web Service. The WSDL is self explanatory and hence not talked about in detail. It helps in understanding the complete framework.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tm=
"http://microsoft.com/wsdl/mime/textMatching/" xmlns:soapenc=
"http://schemas.xmlsoap.org/soap/encoding/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap12=
"http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://tempuri.org/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
      <s:element name="getQuotesWS">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="organization_name" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="getQuotesWSResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="getQuotesWSResult" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="getPredictionWS">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="organization_name" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="getPredictionWSResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="getPredictionWSResult" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="getQuotesWSSoapIn">
    <wsdl:part name="parameters" element="tns:getQuotesWS"/>
  </wsdl:message>
  <wsdl:message name="getQuotesWSSoapOut">
    <wsdl:part name="parameters" element="tns:getQuotesWSResponse"/>
  </wsdl:message>
  <wsdl:message name="getPredictionWSSoapIn">
    <wsdl:part name="parameters" element="tns:getPredictionWS"/>
  </wsdl:message>
  <wsdl:message name="getPredictionWSSoapOut">
    <wsdl:part name="parameters" element="tns:getPredictionWSResponse"/>
  </wsdl:message>
  <wsdl:portType name="ServiceSoap">
    <wsdl:operation name="getQuotesWS">
```

```
    <wsdl:input message="tns:getQuotesWSSoapIn"/>
    <wsdl:output message="tns:getQuotesWSSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="getPredictionWS">
    <wsdl:input message="tns:getPredictionWSSoapIn"/>
    <wsdl:output message="tns:getPredictionWSSoapOut"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ServiceSoap" type="tns:ServiceSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="getQuotesWS">
    <soap:operation soapAction="http://tempuri.org/getQuotesWS" style="document"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getPredictionWS">
    <soap:operation soapAction="http://tempuri.org/getPredictionWS" style="document"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="ServiceSoap12" type="tns:ServiceSoap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="getQuotesWS">
    <soap12:operation soapAction="http://tempuri.org/getQuotesWS" style="document"/>
    <wsdl:input>
      <soap12:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getPredictionWS">
    <soap12:operation soapAction="http://tempuri.org/getPredictionWS" style="document"/>
    <wsdl:input>
      <soap12:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="Service">
  <wsdl:port name="ServiceSoap" binding="tns:ServiceSoap">
    <soap:address location="http://localhost/StockPredictorWS/Service.asmx"/>
```

```
    </wsdl:port>
    <wsdl:port name="ServiceSoap12" binding="tns:ServiceSoap12">
      <soap12:address location="http://localhost/StockPredictorWS/Service.asmx"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

4.4 SOAP Request and Response:

```
POST /StockPredictorWS/Service.asmx HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd=
"http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <getPredictionWS xmlns="http://tempuri.org/">
      <organization_name>string</organization_name>
    </getPredictionWS>
  </soap12:Body>
</soap12:Envelope>

HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd=
"http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <getPredictionWSResponse xmlns="http://tempuri.org/">
      <getPredictionWSResult>string</getPredictionWSResult>
    </getPredictionWSResponse>
  </soap12:Body>
</soap12:Envelope>
```

5. STOCK PREDICTION USING NEURAL NETWORKS

Predicting the stock market is not a simple task, mainly as a consequence of the close to random-walk behavior of a stock time series. Different techniques are being used in the trading community for prediction tasks. In recent years the concept of neural networks has emerged as one of them. We have developed an efficient tool for stock market forecasting based on Neural Networks.

Machine learning:

The machine learning module consists of two stages (repeated many times). First, we present the system with the input data and obtain the output. Second, we adjust the system, to make output closer to what it should be. The first stage is referred as feed-forward, the second as back-propagation.

5.1 Neural Networks technology:

Prediction of stock market returns is an important issue in finance. Nowadays artificial neural networks (ANNs) have been popularly applied to finance problems such as stock exchange index prediction. An ANN model is a computer model whose architecture essentially mimics the learning capability of the human brain. The processing elements of an ANN resemble the biological structure of neurons and the internal operation of a human brain.

The use of ANN comes in because they can learn to detect complex patterns in data. It is able to work parallel with input variables and consequently handle large sets of data swiftly. The principal strength with the network is its ability to find patterns and irregularities as well as detecting multi-dimensional non-linear connections in data. In mathematical terms, they are universal non-linear function approximators meaning that given the right data and configured correctly; they can capture and model any input-output relationships.

This not only removes the need for human interpretation of charts or the series of rules for generating entry/exit signals but also provides a bridge to fundamental analysis as that type of data can be used as input. In addition, as ANNs are essentially non-linear statistical models, their accuracy and prediction capabilities can be both mathematically and empirically tested. This quality is extremely useful for modeling dynamical systems, e.g. the stock market.

5.1.1 Artificial Neural Network Approach:

One of the most important factors in constructing a neural network is deciding on what the network will learn. The goal of most of these networks is to decide when to buy or sell securities based on previous market indicators. The challenge is determining which indicators and input data will be used, and gathering enough training data to train the system appropriately.

We are using Multi Layer Feed-Forward network topology for our project. Input layer is composed of a set of inputs that feed input patterns to the network. The inputs that we provide to our input layer is the sets of 'closing prices' of a company. Following the input layer there will be at least one or more intermediate layers, often called hidden layers. Hidden layers will then be followed by an output layer, where the results can be achieved (Figure-). In feed-forward networks all connections are unidirectional. These networks, also known as back-propagation networks, are mainly used for applications requiring static pattern classification. The main advantage of these networks is their ease of use and approximation of any input/output map. The main disadvantage is that they train slowly and require lots of training data.

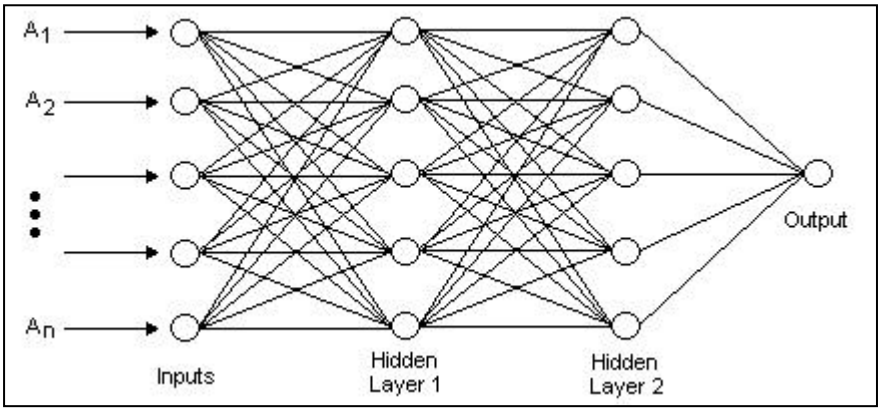


Fig 17: Multi Layer Feed Network Topology

5.1.2 Network Training and Testing:

Training a network involves presenting input patterns in a way so that the system minimizes its error and improves its performance. A multilayer feedforward network uses backpropagation training algorithm. Backpropagation is the process of backpropagating errors through the system from the output layer towards the input layer during training. Backpropagation is necessary because hidden units have no training target value that can be used, so they must be trained based on errors from previous

layers. The output layer is the only layer which has a target value for which to compare. As the errors are backpropagated through the nodes, the connection weights are changed. Training occurs until the errors in the weights are sufficiently small to be accepted. It is interesting to note that the type of activation function used in the neural network nodes can be a factor on what data is being learned.

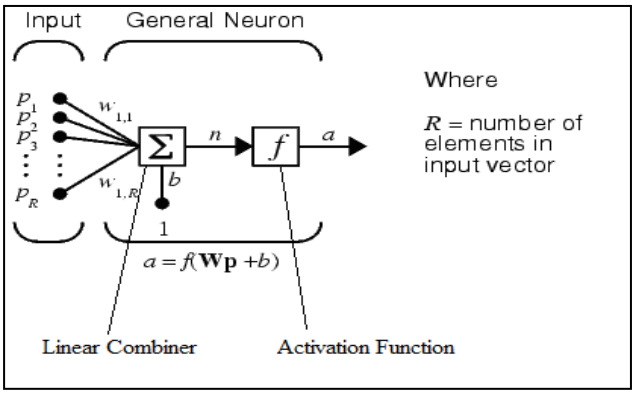
This process is performed internally by neural network toolbox in MATLAB by using the 'train' function. The function just accepts input parameters on which it trains network. As mentioned before we provide sequences of closing price of a particular company as an input parameter to the 'train' function. We are providing 180 sequences, each sequence having 10 different closing prices which is randomly picked up. The train procedure involves training the network on most of the patterns (usually around 90%) and then testing the network on the remaining patterns. The network's performance on the test set is a good indication of its ability to generalize and handle data it has not been trained on. The system is trained on k-1 sets and tested on the remaining k set times, using a different test set each time.

Conditions for halt in training process:

- The maximum number of epochs (repetitions) is reached.
- The maximum amount of time is exceeded.
- Performance is minimized to the goal.
- The performance gradient falls below minimum performance gradient .
- Validation performance has increased more than maximum validation failures since the last time it decreased (when using validation).

5.1.3 Neuron Models:

Below are the architectures of the network that are most commonly used with the backpropagation algorithm - the multilayer feedforward network. The routines in the Neural Network Toolbox can be used to train more general networks; they are briefly explained below.



An elementary neuron with R inputs is shown alongside. Each input is weighted with an appropriate w. The sum of the weighted inputs and the bias forms the input to the transfer function f. Neurons may use any differentiable transfer function f to generate their output.

Fig 18: Basic Neuron Model

A **Linear Combiner**, which is a function that takes all inputs and produces a single value. A simple way of doing it is by adding together the Input (p1,p2…pR) multiplied by the Synaptic Weight (w1,w2….wR).

By applying the **Activation function**, it will take any input from minus infinity to plus infinity and squeeze it into the -1 to 1 or into 0 to 1 interval.

After the neuron in the first layer received its input, it applies the Linear Combiner and the Activation Function to the inputs and produces the Output. This output, as you can see from figure 19, will become the input for the neurons in the next layer. So the next layer will feed forward the data, to the next layer. And so on, until the last layer is reached.

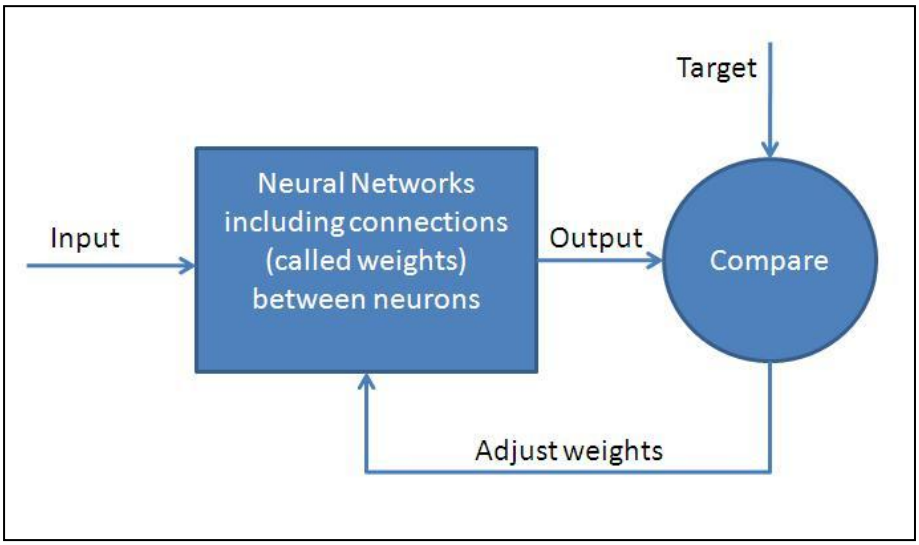


Fig19: Basic Functionality of Neural Network

When we work with yesterday's price, we not only know the price for the "day - 1", but also the price we are trying to predict, called the 'Desired output' of the Neural Net. When we compare the two values, we can compute the Error:

dError = dDesiredOutput - dOutput;

Now we can adjust this particular neuron to work better with this particular input. For example, if the dError is 10% of the dOutput, we can we can increase all synaptic weights of the neuron by 10%.

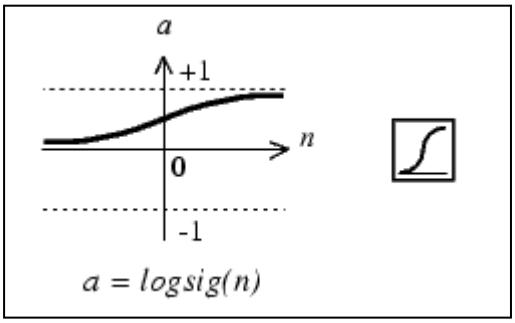
The problem with this approach is that the next input will require a different adjustment.

Hence we perform the following operation:
dNewWeight = dOldWeight * dAdjustment * dLearningRate.

The learning rate (dLearningRate) is of importance of a single pattern. For example, we can set it to 0.01, then it will take 100 patterns to make a 10% adjustment.

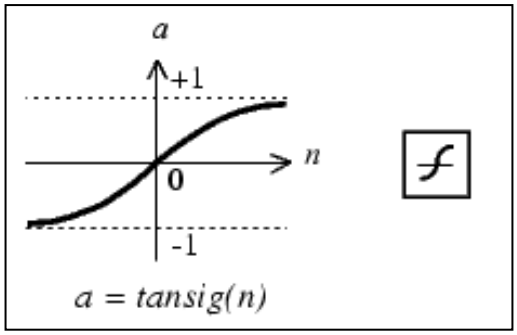
- **Types of Neuron Models (logsig, tansig, purelin):**

Multilayer networks often use the log-sigmoid transfer function logsig.



The function **logsig** generates outputs between 0 and 1 as the neuron's net input goes from negative to positive infinity.

Fig 20: Log Sigmoid Transfer Function



Alternatively, multilayer networks may use the tan-sigmoid transfer function **tansig**.

Fig 21: Tan Sigmoid Transfer Function

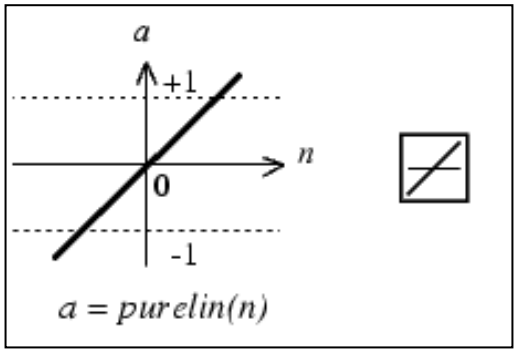


Fig 22:Linear Transfer Function

Occasionally, the linear transfer function **purelin** is used in backpropagation networksIf the last layer of a multilayer network has sigmoid neurons, then the outputs of the network are limited to a small range. If linear output neurons are used the network outputs can take on any value.
Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear and linear relationships between input and output vectors.

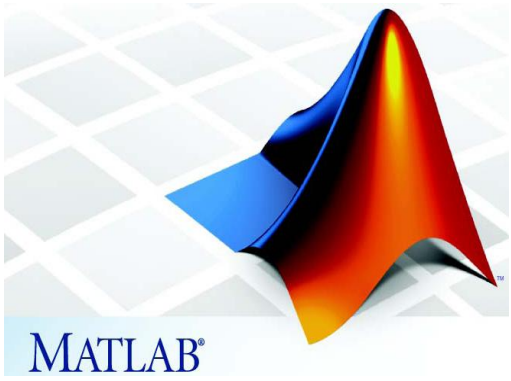
5.2 Matlab(v7.1) Neural Network Toolbox:

The Matlab toolbox allows us to initialize a new network with the selected neurons.

To train the network we need to enter the following command:

Net = train (net, Inputs, Targets);

The toolbox provides various train functions each built with a different feature and optimized for specific applications



- **Batch Training (train)**

  In batch mode the weights and biases of the network are updated only after the entire training set has been applied to the network.

- **Batch Gradient Descent (traingd)**

  The weights and biases are updated in the direction of the negative gradient of the performance function.

- **Resilient Backpropagation (trainrp)**

The purpose of the resilient back-propagation (Rprop) training algorithm is to eliminate these harmful effects of the magnitudes of the partial derivatives. Only the sign of the derivative can determine the direction of the weight update; the magnitude of the derivative has no effect on the weight update. We use resilient back-propagation for our project.

The following code recreates the previous network and trains it using the Rprop algorithm.

```
p = training input;
t = training target;
net = newff(p,t,3,{},'trainrp');
net = train(net,p,t);
```

During training, as in function fitting, the training window opens. Thiswindow displays training progress. To interrupt training at any point, click Stop Training.

- **Simulation**

We can then call the simulate function which tests the neural network with a set of testing inputs and compares the simulated output and the actual output.

```
Y = sim(net,P)
```

5.3 Basic working of our model:

The model (neural network) accepts, as input, a sequence of given length N. The system can determine if at least one of future prices - within an observation window of fixed length M - will be higher or lower than current price.

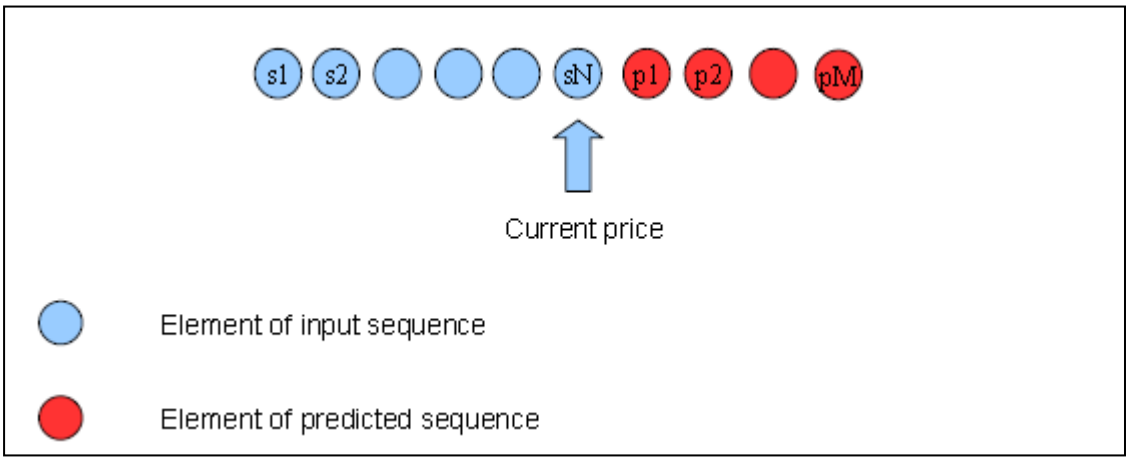


Fig 23: Basic Working of our Model

A buy signal is given if at least one of red circles is greater than current price. A sell signal is given if at least one of red circles is lower than current price. Formally a buy signal is verified if (p1>sN) OR (p2>sN) OR ... OR (pM>sN), a sell signal is verified if (p1<sN) AND (p2<sN) AND .... AND (pM<sN).

The neural network memorizes or learns the sequences of the form s1,s2….sN and also memorizes the buy signal or sell signal. Now during the test phase the neural network takes sequences, predicts the output based on the training and then verifies if the predicted value matches with actual value. This where performance evaluation is done and if the neural network seems to be giving satisfactory performance then the final sequence is given to it and the network is asked to predict the future value.

5.4 Detailed description / Program algorithm

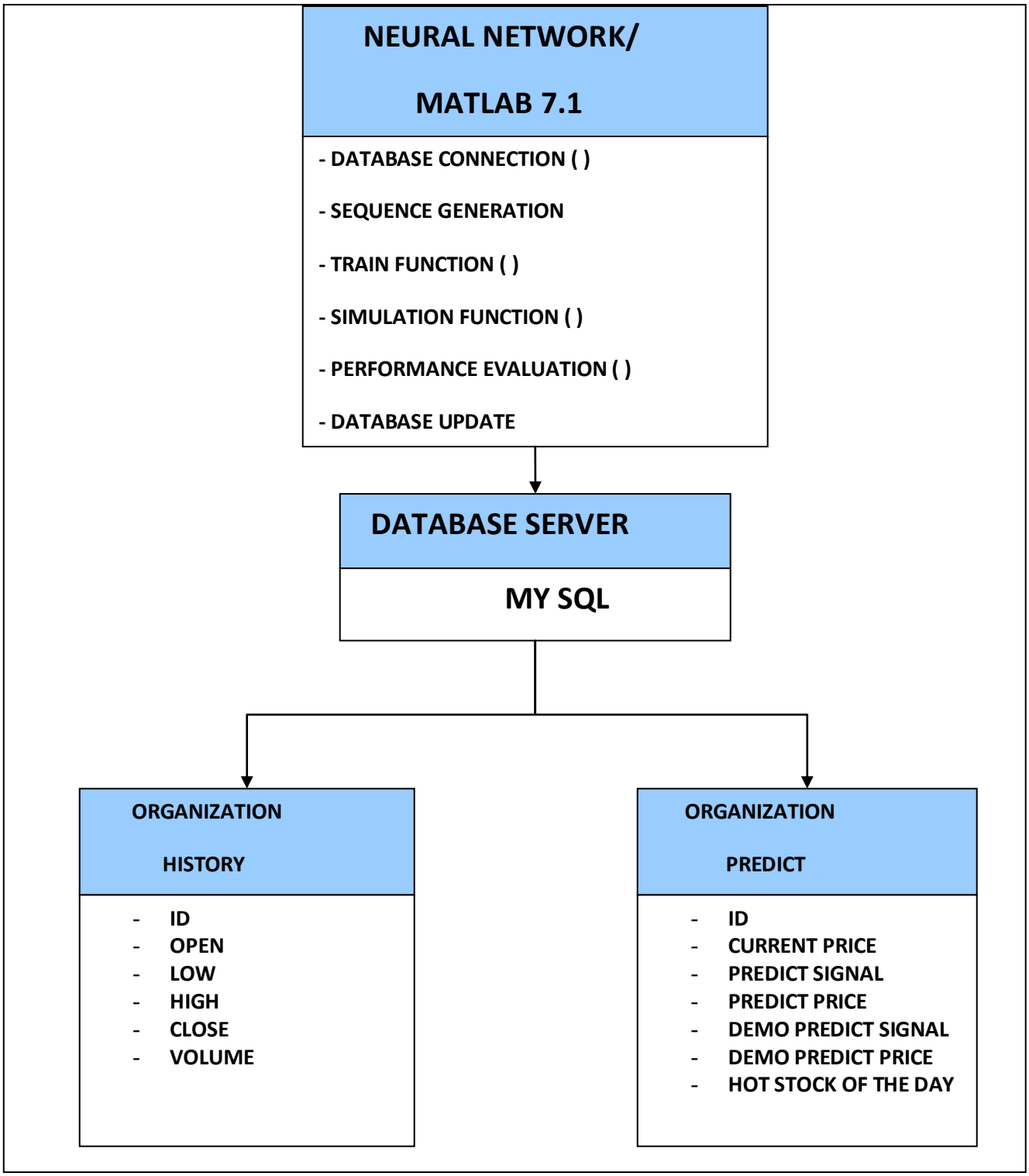


Fig 24: Class Diagram for Stock Prediction

Class Diagram Description:

NAME	DOCUMENTATION
NEURAL NETWORK	Matlab neural network is used for training the system with the historical data. This will enable us to make prediction for future value. The network is trained and predicted value is obtained everyday at 5:00pm
DATABASE SERVER	Matlab connects to Mysql Database and historical data is provided. This database has updated values provided by PHP script
ORGANIZATION HISTORY	Company ID and Close values of each company are available for the neural network for training
ORGANIZATION PREDICT	
	The predicted value by neural network is inserted in organization history as buy or sell signal and its corredponding predicted price.

Steps:

1. A timer is set for the Matlab prediction module. We have initiated the timer such that the Matlab code starts everyday at **5:00 pm** (The Nasdaq closes at 4:00pm). Hence everyday matlab will run through the prediction process and make prediction for the next day. This way we ensure that the recommendation/ prediction for the next day is available to the client before the start of the next day's trading session.

2. Initially MATLAB connects to MySQL database. This will enable MATLAB to pick up the historical values of the particular company from the MySQL database. We will be working on 'close prices' of a particular security. Once closing price of each company is available we initiate 20 neural networks, one for each company.

3. Using a for loop, we select one company at a time. Then we select randomly k prices for training and remaining for testing of that company.

4. For training, generate k sequences from the k prices, each sequence having N closing prices. This generated sequence is the training input for the network. (We have assigned 180 values for training and remaining for testing, from data of 1

year of closing prices of a particular stock. Thus we are generating 180 sequences for training each sequence having N=10 closing prices.)

5. Generate training target, training target is a sequence of M closing prices. (We have assigned M=3, i.e. three targeted prices following immediately the input sequence)

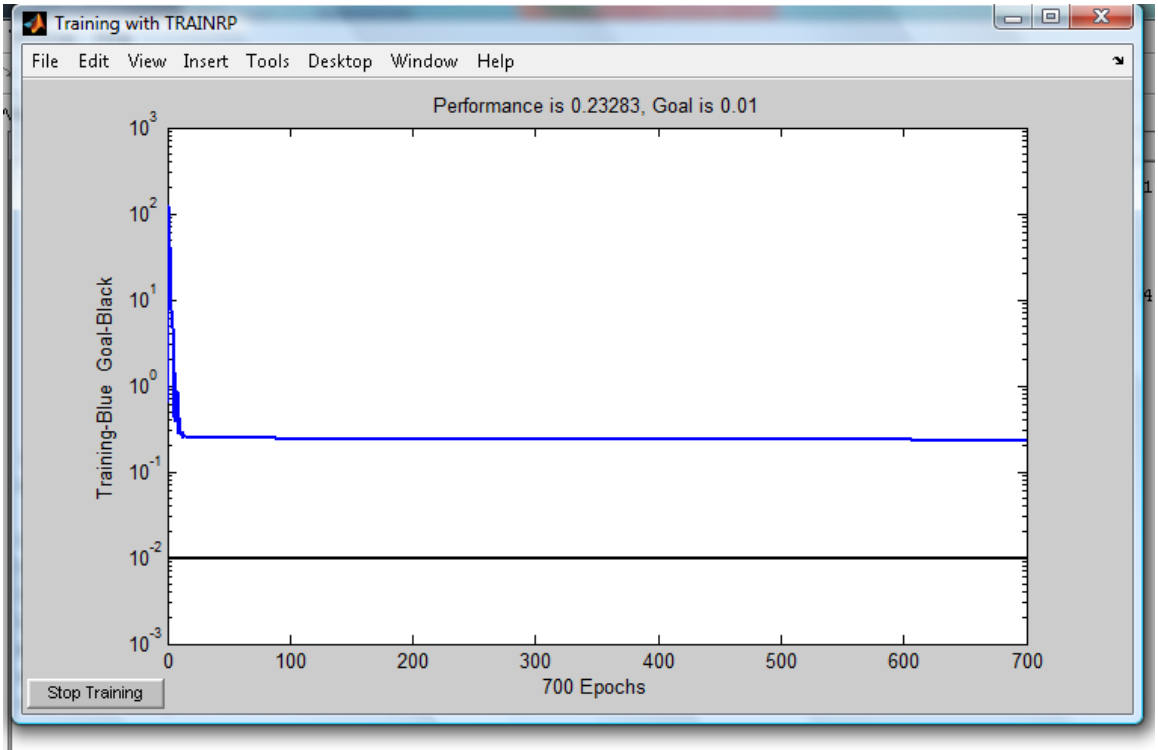
6. Train the neural network on following condition, if P1> SN OR P2> SN....OR PM>SN, then generate a buy signal else generate sell signal (buy signal=1, sell signal =0). (Alongwith this we also calculate the difference in the last price and the maximum of target price and difference of last price and minimum of target price. The moving average of all these accumulated prices forms the basis for the neural network to calculate the predicted price.)

7. Now during the test phase the neural network takes sequences, predicts the output based on the training and then verifies if the predicted value matches with actual value.

8. Once test and train are over, the neural network predicts the future signal 1(buy signal) or 0 (sell signal) and the predicted price. These values are updated in the database and the process (step 2 to step 7) repeats for the next company. Thus we have the predict signal and predicted price for all the companies. The code then searches for the stock/ company for which the neural network predicts the highest rise in the predicted price. This is done so that we can inform the client of the stock which is going to make the maximum profit for the next day. We call this the **"HOT STOCK OF THE DAY"**. This information may also be provided to the clients through email alerts.

9. (ONLY FOR DEMO) We have also created an arrangement so that professor can select and day of his choice so that system can be set to go that many days back in time and asked to predict the price for that day and then professor can check if the predicted price for that day was atleast as close as possible to the actual price of that day. This has only been done for convenience of verifying the system during demo.

SNAPSHOT OF THE TRAINING PROCESS:




```
Please wait... training in progress.
TRAINRP, Epoch 0/2500, MSE 0.616253/0.01, Gradient 13.2879/1e-006
TRAINRP, Epoch 100/2500, MSE 0.242579/0.01, Gradient 0.0212978/1e-006
TRAINRP, Epoch 200/2500, MSE 0.23941/0.01, Gradient 0.0152663/1e-006
TRAINRP, Epoch 300/2500, MSE 0.237618/0.01, Gradient 0.0111531/1e-006
TRAINRP, Epoch 400/2500, MSE 0.235707/0.01, Gradient 0.0101599/1e-006
TRAINRP, Epoch 500/2500, MSE 0.234529/0.01, Gradient 0.00975449/1e-006
```

The blue line indicates the training curve. The black line is the goal line.

We use TRAINRP (explained earlier in section 5.2). Epoch, MSE, and Gradient are used as parameters to stop the training process. We use a 3 layer neural network (Input layer, Hidden layer and Output Layer) and purelin neurons at all the layers.

The created network looks like this:

net = Neural Network object:

architecture:

numInputs: 10

numLayers: 3

biasConnect: [1; 1; 1]

inputConnect: [3x10 boolean]

layerConnect: [0 0 0; 1 0 0; 0 1 0]

outputConnect: [0 0 1]

targetConnect: [0 0 1]

numOutputs: 1 (read-only)

numTargets: 1 (read-only)

numInputDelays: 0 (read-only)

numLayerDelays: 0 (read-only)

sub object structures:

inputs: {10x1 cell} of inputs

layers: {3x1 cell} of layers

outputs: {1x3 cell} containing 1 output

targets: {1x3 cell} containing 1 target

biases: {3x1 cell} containing 3 biases

inputWeights: {3x10 cell} containing 1 input weight

layerWeights: {3x3 cell} containing 2 layer weights

functions:

adaptFcn: 'trains'

initFcn: 'initlay'

performFcn: 'mse'

trainFcn: 'trainrp'

parameters:

adaptParam: .passes

initParam: (none)

performParam: (none)

trainParam: .epochs, .show, .goal, .time,

.min_grad, .max_fail, .delt_inc, .delt_dec,delta0, .deltamax, .mc

weight and bias values:

IW: {3x10 cell} containing 1 input weight matrix

LW: {3x3 cell} containing 2 layer weight matrices

b: {3x1 cell} containing 3 bias vectors

other:

userdata: (user stuff)

Recommendation : When the user clicks getpredict, this is the kind of output that is expected

Predict Signal: Buy or sell

Amount: The predicted price of the requested company

Time Horizon: The system predicts that the predicted price is expected to be achieved in next **3 days**. Hence 3 days is the time frame that we provide to the client.

5.5 Performance Evaluation

We calculate the confidence level of the predicted of each individual company separately.

This is done with the help of the testing phase. We define a parameter called “Recognition Rate” which is

$$\textbf{Recognition rate} = \frac{\text{Total number of correct detected patterns}}{\text{Total number of total detected patterns}}$$

The Recognition Rate has been found to be in the range from 50% to 90% for different companies which is a fairly good performance heuristic.

These values are also placed in the database and made available to the client when he does ask for the prediction for a particular company. In this way the user is made aware of how accurate the system is or how much should he rely on that prediction.

6. STOCK PREDICTION USING TECHNICAL INDICATORS

Definition: A technical indicator is any class of metrics whose value is derived from generic price activity in a stock or asset.

Technical indicators look to predict the future price levels, or simply the general price direction, of a security by looking at past patterns. Collectively called "technical's", they are distinguished by the fact that they do not analyze any part of the fundamental business, like earnings, revenue and profit margins. Technical indicators are used most extensively by active traders in the market, as they are designed primarily for analyzing short-term price movements. To a long-term investor, most technical indicators are of little value, as they do nothing to shed light on the underlying business. The most effective uses of technicals for a long-term investor are to help identify good entry and exit points for the stock by analyzing the long-term trend.

We select a distinguished choice of indicators which together provide us with recommendation\s for buying, selling or holding stocks. They are as follows:

6.1 Relative Strength Index (RSI):

RSI is a type of momentum oscillator (see glossary).The RSI compares the magnitude of a stock's recent gains to the magnitude of its recent losses and turns that information into a number that ranges from 0 to 100. It takes a single parameter, the number of time periods to use in the calculation. A popular method of analyzing the RSI is to look for a divergence in which the security is making a new high, but the RSI is failing to surpass its previous high. This divergence is an indication of an impending reversal. When the Relative Strength Index then turns down and falls below its most recent trough, it is said to have completed a "failure swing". The failure swing is considered a confirmation of the impending reversal.

Prediction Algorithm:

1. Initialize Tc = today's closing price, yc = yesterday's closing price
2. Set peak parameters : Upclose = Downclose = 0
3. If (tc > yc) set : upclose = upclose + tc
   Else if (tc < yc) set : downclose = downclose + tc
4. Repeat step 3 for 'x' consecutive days
5. RSI = 100 – 100 / (1 + upclose / downclose)
6. If (RSI > 50), then predict increase in impending closing price
   Else if (RSI < 50), predict decrease in impending closing price

Source:
http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:relative_strength_in

6.2 Stochastic Oscillator:

It is a type of momentum indicator. It was Introduced by George Lane in the 1950s, the Stochastic oscillator compares the closing price of a commodity to its price range over a given time span. Prices tend to close near their past highs in bull markets, and near their lows in bear markets. Transaction signals can be spotted when the stochastic oscillator crosses its moving average. Two stochastic oscillator indicators are typically calculated to assess future variations in prices, a fast (%K) and slow (%D). Comparisons of these statistics are a good indicator of speed at which prices are changing or the Impulse of Price. We use the fast stochastic oscillator for our prediction. %K calculates the ratio of two closing price statistics: the difference between the latest closing price and the lowest closing price in the last N days over the difference between the highest and lowest closing prices in the last N days.

Prediction algorithm:

1. Initialize tc = today's closing price
2. Find hn : highest price with past N days
3. Find ln: lowest price within past N days
4. Calculate K = (tc – ln) / (hn – ln)
5. Calculate % K = K * 100
6. If (%K > 80), predict increase in impending closing price
   Else if (%K < 20), predict decrease in impending closing price

Source:
http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:stochastic_oscillato

6.3 Moving averages:

Moving averages are generally used to measure momentum and define areas of possible support and resistance.They are used to emphasize the direction of a trend and to smooth out price and volume fluctuations, or "noise", that can confuse interpretation. Typically, upward momentum is confirmed when a short-term average (e.g.15-day) crosses above a longer-term average (e.g. 50-day). Downward momentum is confirmed when a short-term average crosses below a long-term average. The most common way to interpreting the price moving average is to compare its dynamics to the price action. When the instrument price rises above its moving average, a buy signal appears, if the price falls below its moving average, what we have is a sell signal.

This trading system, which is based on the moving average, is not designed to provide entrance into the market right in its lowest point, and its exit right on the peak. It allows to act according to the following trend: to buy soon after the prices reach the bottom, and to sell soon after the prices have reached their peak.

Prediction algorithm:

1. Initalize tma = today's moving average, yma = yesterday's moving average to 0
2. Find tma as average of closing prices of last N days.
3. Find yma as average of closing prices of last N days starting from yesterday.
4. If (tma > yma), predict increase in impending closing price
   Else if (tma < yma), predict decrease in impending closing price

Source:
http://stockcharts.com/help/doku.php?id=chart_school:technical_indicators:moving_averages

6.4 Price momentum oscillator (PMO):

To find PMO, The current price is compared against a previous price to measure the magnitude of price change over a specific period of time. The rate of change curve is smoothed and plotted against a horizontal line equal to one. Indicator values much greater or less than one signal strong price momentum in the same direction, while values near one reflect little momentum. Though this is similar to RSI and stochastic oscillator, it deals more with instant changes rather than gradual change of prices over a certain period.

Prediction algorithm:

1. Initialize tc = today's closing price, ndac = closing price N days ago
2. Set pmo = tc – tdac
3. If (pmo > 0) predict increase in impending closing price
   Else if (pmo < 0), predict decrease in impending closing price.

6.5 Money flow index (MFI):

Money Flow Index (MFI) is the technical indicator, which indicates the rate at which money is invested into a security and then withdrawn from it.[7] Construction and interpretation of the indicator is similar to Relative Strength Index with the only difference that volume is important to MFI.

When analyzing the money flow index one needs to take into consideration the following points:

- Divergences between the indicator and price movement. If prices grow while MFI falls (or vice versa), there is a great probability of a price turn;
- Money Flow Index value, which is over 80 or under 20, signals correspondingly of a potential peak or bottom of the market.

Prediction Algorithm:

1. Define a "Typical Price" (TP) for the specified period:
   TP = (HIGH + LOW + CLOSE) / 3
2. Calculate "Money Flow" value (MF):
   MF = TP x VOLUME
3. If "Typical Price" is higher than the preceding one then "Money Flow" is positive.
   If "Typical Price" is lower than the preceding one then "Money Flow" is negative.
4. Calculate "Positive Money Flow" and "Negative Money Flow":
   - "Positive Money Flow" is the total value of all positive money flows for the specified time period.
   - "Negative Money Flow" is the total value of all negative money flows for the specified time period.
5. "Money Ratio" (MR) is calculated as follows:
   MR = POSITIVE MONEY FLOW / NEGATIVE MONEY FLOW
6. Then using "Money Ratio", calculate "Money Flow Index":
   MFI = 100 - (100 / (1 + MR))
7. If MFI > 80, predict BUY
   ElseIf MFI < 20, predict SELL

Where:
HIGH - current high price
LOW - current low price
CLOSE - current close price
VOLUME - current volume of security

Source:
http://www.metaquotes.net/techanalysis/indicators/money_flow_index

6.6 DeMarker Indicator:

DeMarker Technical Indicator is based on the comparison of the current bar maximum with the previous one. If the previous price maximum is lower than that of the current, the respective difference between two is registered. If it is higher or equal to the current, the nought value is registered. The differences received for N periods are then summarized used as the numerator of the DeMarker. Then the numerator is divided by

the same value plus the sum of differences between the price minima of the previous and the current bars. If the price minimum of the current period is higher than that of the previous, the nought value is registered.

When the indicator is below 30, the upward price reversal should be expected. When the indicator rises above 70, the downward price reversal should be expected. Using longer periods when calculating the indicator helps to catch the long term market tendency. Indicators based on short periods let you plan the transaction time so that it falls in with the major trend.

Prediction Algorithm:

1. Calculate DeMax (i):
   If HIGH(i) > HIGH(i - 1), then DeMax(i) = HIGH(i) - HIGH(i - 1), else DeMax(i) = 0

2. Calculate DeMin (i)
   If LOW(i) < LOW(i - 1), then DeMin(i) = LOW(i - 1) - LOW(i), else DeMin(i) = 0

3. Calculate DeMarker indicator :
   DMark (i) = SMA (DeMax, N) / (SMA (DeMax, N) + SMA (DeMin, N))

4. If DMark<30, predict BUY
   Elself DMark>70, Predict SELL

Where:
HIGH (i) - current maximum price;
LOW (i) - current minimum price;
HIGH (i - 1) - previous maximum price;
LOW (i - 1) - previous minimum price;
SMA - simple moving average;
N - number of periods used in the calculation

Source:
http://www.metaquotes.net/techanalysis/indicators/demarker/

6.7 William's percent range

Williams' Percent Range Technical Indicator (%R) is a dynamic technical indicator, which determines whether the market is overbought/oversold. Williams' %R is very similar to the Stochastic Oscillator. The only difference is that %R has an upside down scale and the Stochastic Oscillator has internal smoothing.

To show the indicator in this upside down fashion, one places a minus symbol before the Williams Percent Range values (for example -30%). One should ignore the minus symbol when conducting the analysis. Indicator values ranging between 80 and 100% indicate that the market is oversold. Indicator values ranging between 0 and 20% indicate that the market is overbought. As with all overbought/oversold indicators, it is best to wait for the security's price to change direction before placing your trades. For example, if an overbought/oversold indicator is showing an overbought condition, it is wise to wait for the security's price to turn down before selling the security.

An interesting phenomenon of the Williams Percent Range indicator is its uncanny ability to anticipate a reversal in the underlying security's price. The indicator almost always forms a peak and turns down a few days before the security's price peaks and turns down. Likewise, Williams Percent Range usually creates a trough and turns up a few days before the security's price turns up.

Prediction Algorithm:

1. Calculate %R = (HIGH(i-n)-CLOSE)/(HIGH(i-n)-LOW(i-n))*100
2. If %R<20, predict BUY
   ElseIf %R>80, predict SELL

Where:
CLOSE - today's closing price;
HIGH(i-n) - the highest high over a number (n) of previous periods;
LOW(i-n) - the lowest low over a number (n) of previous periods.

Source:
http://www.metaquotes.net/techanalysis/indicators/williams_percent/

6.8 Commodity Channel Index:

Commodity Channel Index Technical Indicator (CCI) measures the deviation of the commodity price from its average statistical price. High values of the index point out that the price is unusually high being compared with the average one, and low values show that the price is too low. In spite of its name, the Commodity Channel Index can be applied for any financial instrument, and not only for the wares.

There are two basic techniques of using Commodity Channel Index:

1. **Finding the divergences:** The divergence appears when the price reaches a new maximum, and Commodity Channel Index can not grow above the previous maximums. This classical divergence is normally followed by the price correction.

2. **As an indicator of overbuying/overselling:** Commodity Channel Index usually varies in the range of ±100. Values above +100 inform about overbuying state (and about a probability of correcting decay), and the values below 100 inform about the overselling state (and about a probability of correcting increase).

Prediction Algorithm:

1. Find a typical price: add high, low and close of each bar and divide it by 3:
   TP = (HIGH + LOW + CLOSE) / 3
2. Calculate the simple moving average of the typical prices over n-periods:
   SMA (TP, N) = SUM (TP, N) / N
3. Subtract SMA (TP, N) from the typical prices (TP) of each preceding n periods:
   D = TP - SMA (TP, N)
4. Calculate simple moving average of the absolute D values over n periods:
   SMA (D, N) = SUM (D, N) / N
5. Multiply SMA (D, N) by 0.015
   M = SMA (D, N) * 0.015
6. Divide M by D:
   CCI = M / D
7. If CCI<-100, predict BUY
   ElseIf CCI>100, predict SELL

Where:
HIGH - current maximum price;
LOW - current minimum price;
CLOSE – current closing price;
SMA - simple moving average;
SUM - sum;
N - number of periods used for calculation.

Source:
http://www.metaquotes.net/techanalysis/indicators/commodity_channel_index/

6.9 Aroon Indicator:

Aroon indicator is constructed on calculation of period's number, those which have passed from the moment of Maximum and minimum within the time range of periods. It lets determine prices fluctuations from the market trend state to the sidewalks state and demonstrates lack of clear tendencies.

The Aroon indicator is used to define if a currency trading price is following a trend or sideways and to measure the trend's strength. If the currency's trading price is going up, the end for this period will be closer, and on the contrary. The Aroon indicator shows in

percents how much time passed between the up (highest) and down (lowest) close starting from the beginning of a period. When "Aroon up" and "Aroon down" are moving simultaneously, there is no clear trend. Then the price is moving sideways, or is going to move sideways. When the "Aroon down" is less than 50 it is a sign that the downtrend is giving up its momentum, while when the Aroon (up) is less than 50, it is a sign that the uptrend is giving up its momentum. When the "Aroon up" or "Aroon down" are more than 70, it demonstrates a strong trend in the same direction, while when the value is less than 30; it means a trend going in an opposite direction.

Prediction Algorithm:

1. Calculate Aroon(up)(AU):
   Aroon(up) = ((# of periods - # of periods since highest close)/ (#periods)) x 100
2. Calculate Aroon(down)(AD):
   Aroon(down) = ((# of periods - # of periods since highest close)/ (#periods)) x 100
3. If (AU>70 && AD<30), predict BUY
   ElseIf (AU<30 && AD>70), predict SELL

Source:
http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:aroon

6.10 Rate of Change (ROC)

The Rate of Change (ROC) indicator is a very simple yet effective momentum oscillator that measures the percent change in price from one period to the next. The ROC calculation compares the current price with the price n periods ago. In our case we have taken n = 14

Formula: ROC = ((Today's close - Close n periods ago) / (Close n periods ago)) * 100

The plot forms an oscillator that fluctuates above and below the zero line as the Rate of Change moves from positive to negative. The oscillator can be used as any other momentum oscillator by looking for higher lows, lower highs, positive and negative divergences, and crosses above and below zero for signals.

Prediction Algorithm.

1. Calculate the Today's close price first and price n period ago.
2. Put the values in the above equation.
3. If ROC>0 predict Buy else predict Sell

Source:
http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:rate_of_change

6.11 Moving Averages Convergence/Divergence (MACD)

The most popular formula for the "standard" MACD is the difference between a security's 26-day and 12-day Exponential Moving Averages (EMAs). Of the two moving averages that make up MACD, the 12-day EMA is the faster and the 26-day EMA is the slower. Closing prices are used to form the moving averages. Usually, a 9-day EMA of MACD is plotted along side to act as a trigger line. A bullish crossover occurs when MACD moves above its 9-day EMA, and a bearish crossover occurs when MACD moves below its 9-day EMA.

Prediction Algorithm:

1. Calcuate the 26 day EMa.
2. Calculate the 12 day EMA
3. Calculate the 9 day Ema
4. Calculate Subtract = (26 dya EMA – 12 day EMA)
5. If Subtract>9dayEMA predict Buy else predict Sell

Source:
http://stockcharts.com/help/doku.php?id=chart_school:technical_indicators:moving_average_conve

6.12 Accumulation/Distribution Line

The Accumulation/Distribution Line is used to assess the cumulative flow of money into and out of a security. It can be used to gauge the general flow of money. An uptrend indicates that buying pressure is prevailing, and a downtrend indicates that selling pressure is prevailing.

Formula: (((C - L) - (H - C)) / (H - L)) = CLV

Where, C= close value of the day
L = Low value of the day
H = High value of the day

Prediction Algorithm

1. Calculate CLV for today as CLV[0] and 60 days ago as CLV[60]
2. Also get closing value for today as close[0] and 60 days ago as close[60]
3. If CLV[0]>CLV[60] and close[0]<close[60] means we have positive divergence which indicates to buy the signal.
4. If CLV[0]<CLV[60] and close[0]>close[60] means we have negative divergence which indicates to sell the signal.

Source:
http://stockcharts.com/help/doku.php?id=chart_school:technical_indicators:accumulation_distrib

6.13 Price Channel.

Price channels form boundaries above and below the price line and can be used as indicators of volatility. Price channels are created by specifying a number of periods that will chart an n-period high or low around the price line. For example, a 20-day price channel will chart the level of the highest high in the last 20 days above the price line, and will chart the level of the lowest low in the last 20 days below the price line. Price channels can be used on daily, weekly, or monthly charts and can generate buy/sell signals at points of breakouts. When the price line breaks above or below the upper or lower price channel respectively, a new high or low becomes present. When the price breaks above a 20-day price channel, the price has reached a 20-day high and could potentially begin an uptrend. In this situation, the upper price channel breakout may signify that it is a good time to buy the stock. In our case we have taken a high and low of close price of the year.

Prediction Algorithm

1) Get the highest value of the Close price from database of one year.
2) Get the lowest value of the Close price from database of one year.
3) Find if there is any breakout for current price with lowest or highest close value.
4) After breaks with highest value if current price increase predict buy
5) If the current price move below after breaking with lowest close value predict sell.
6) For other condition predict hold.

Source:
http://stockcharts.com/help/doku.php?id=chart_school:technical_indicators:price_channels

6.14 Ultimate Oscillator

The "Ultimate" Oscillator combines a stock's price action during three different time frames into one bounded oscillator. Values range from 0 to 100 with 50 as the center line. Oversold territory exists below 30 and overbought territory extends from 70 to 100. Three time frames are used by the Ultimate Oscillator and can be specified by the user. Typically values of 7-periods, 14-periods and 28-periods are used. Note that **these time periods all overlap**, i.e. the 28-period time frame includes both the 14-period time frame and the 7-period time frame. This means that the action of the shortest time frame is included in the calculation **three times** and has a magnified impact on the results.

Formula: Ultimate_Oscillator = (RawUO / (4 + 2 + 1)) * 100

Where RawUO = 4 * (BPSum1 / TRSum1) + 2 * (BPSum2 / TRSum2) + (BPSum3 / TRSum3)

Prediction Algorithm

1. Calculate Today's "True Low (TL)". TL = the lower of today's low or yesterday's close
2. Calculate Today's "Buying Pressure (BP)". BP = Today's close - Today's TL
3. Calculate Today's "True Range (TR)". TR = the higher of 1.) Today's High - Today's Low; 2.) Today's High - Yesterday's Close; 3.) Yesterday's Close - Today's Low.
4. Calculate BPSum1, BPSum2, and BPSum3 by adding up all of the BPs for each of the three specified time frames.
5. Calculate TRSum1, TRSum2, and TRSum3 by adding up all of the TR's for each of the three specified time frames.
6. If (Ultimate_Oscillator>70) predict Buy or if (Ultimate_Oscilltor<30) predict Sell
7. For other condition predict Hold

Source:
http://stockcharts.com/help/doku.php?id=chart_school:technical_indicators:ultimate_oscillator

Performance evaluation:

The evaluation of the algorithms utilizes a hypothesis test. The hypotheses test can be setup to test if the algorithm did better than chance. It was assumed that there was always an equal probability of the stocks going up or down. The null hypothesis stated that the prediction would be wrong 50% or more of the time. The alternative hypotheses stated that the prediction would be correct more than 50% of the time at the 0.05 level of significance. The algorithm can be judged by how many predictions were correct. A test statistic, here the number of correct predictions, can be used to evaluate the algorithms. This test can use the binomial CDF provided by MATLAB where both the number of correct predictions and the total number of predictions can be used with the binomial CDF. The performance on each individual stock was evaluated and then the performance on all stocks combined was evaluated.

We tested the algorithm on data stored in our database and found that it was able to predict if the following day's closing price would increase or decrease better than chance (50%) with a high level of significance. Furthermore, this shows that there is some validity to technical analysis of stocks. This is not to say that this algorithm would make anyone rich, but it may be useful for trading analysis. The algorithms did very well on 60 percent of the stocks. These algorithms only provide a rough prediction about increase/decrease of prices, and need the robust support of neural networks.

7. USER INTERFACE

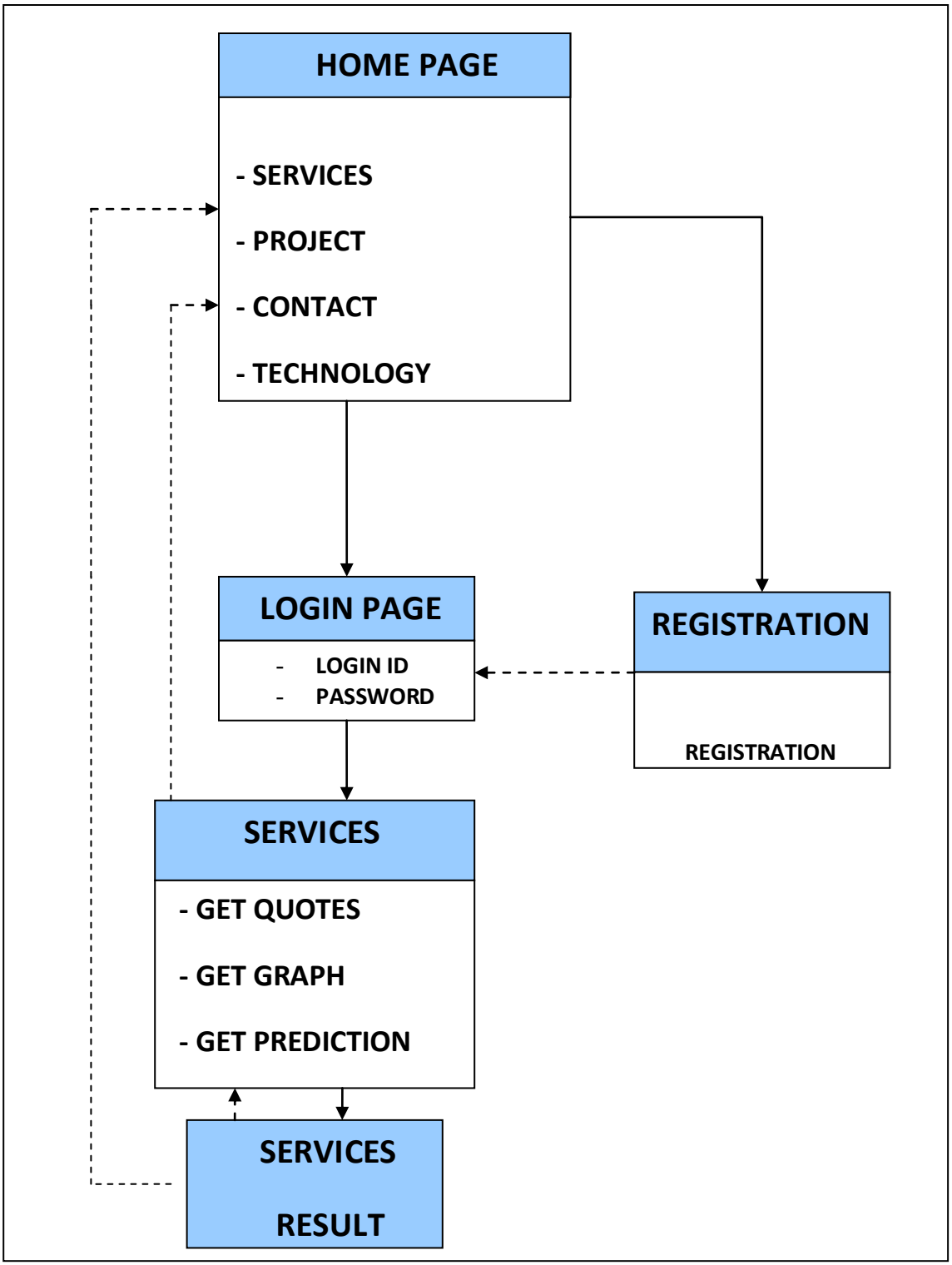


Fig 25: Class Diagram for User Interface

7.1 WEBSITE DESIGN:

We have created user friendly and easy to use client interface. It supports our powerful backend prediction model. The snapshot below gives a glimpse of the website design. Our home page provides exciting features like live business news, live indices and online help via chat.

It is authentication based application wherein members get to login and access our services like get quotes, get graphs, get prediction and get indication.

SNAPSHOTS OF WEB PAGES:

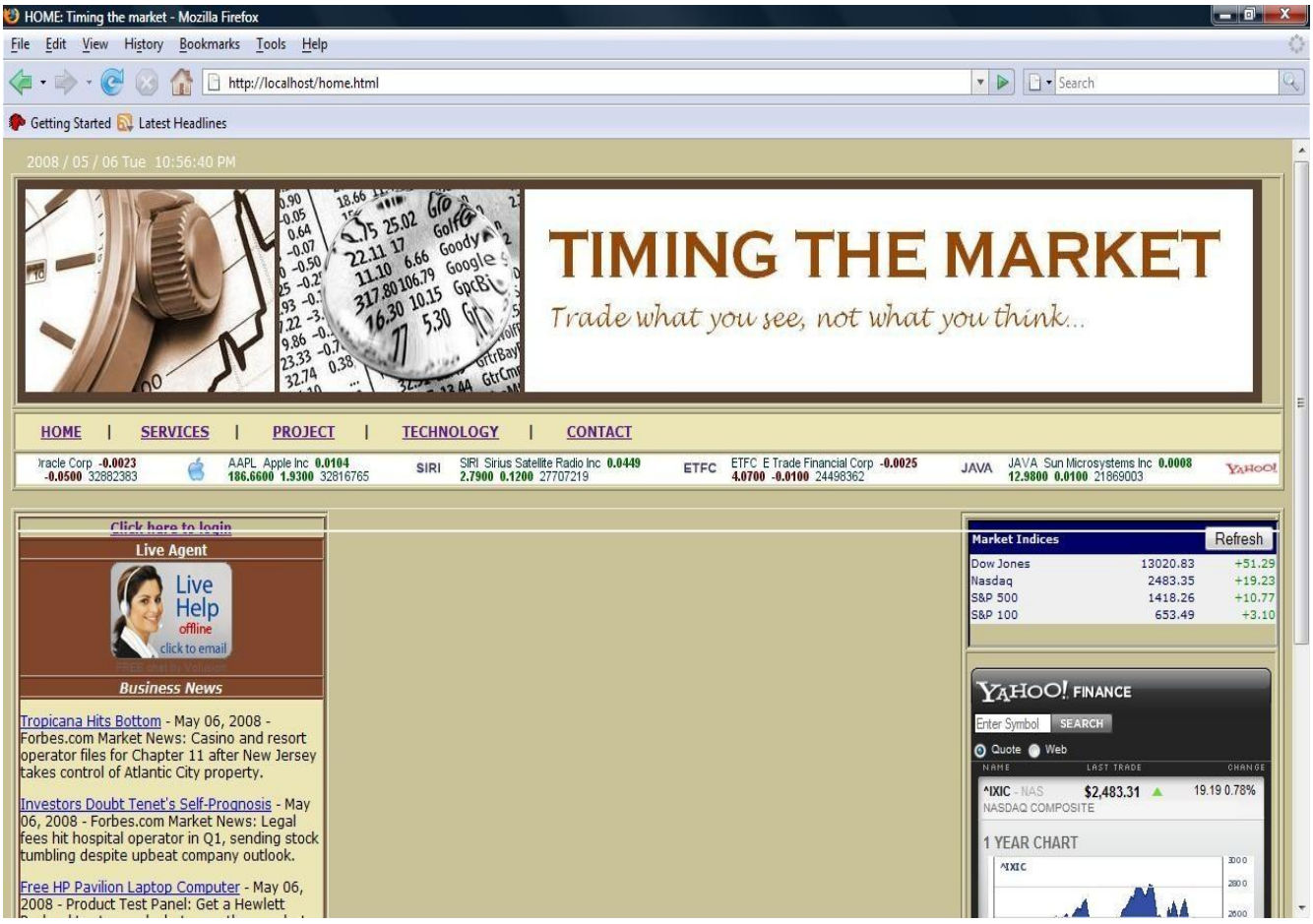


Fig 26: Home Page

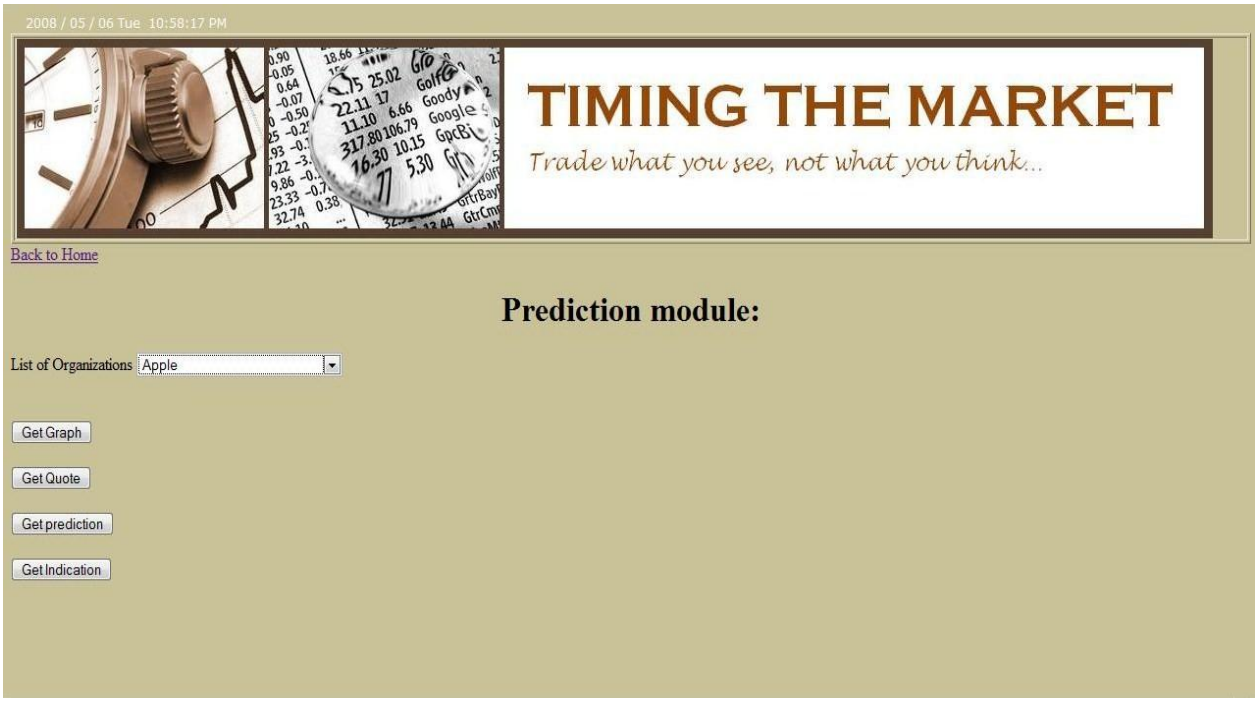


Fig 27: Services Page

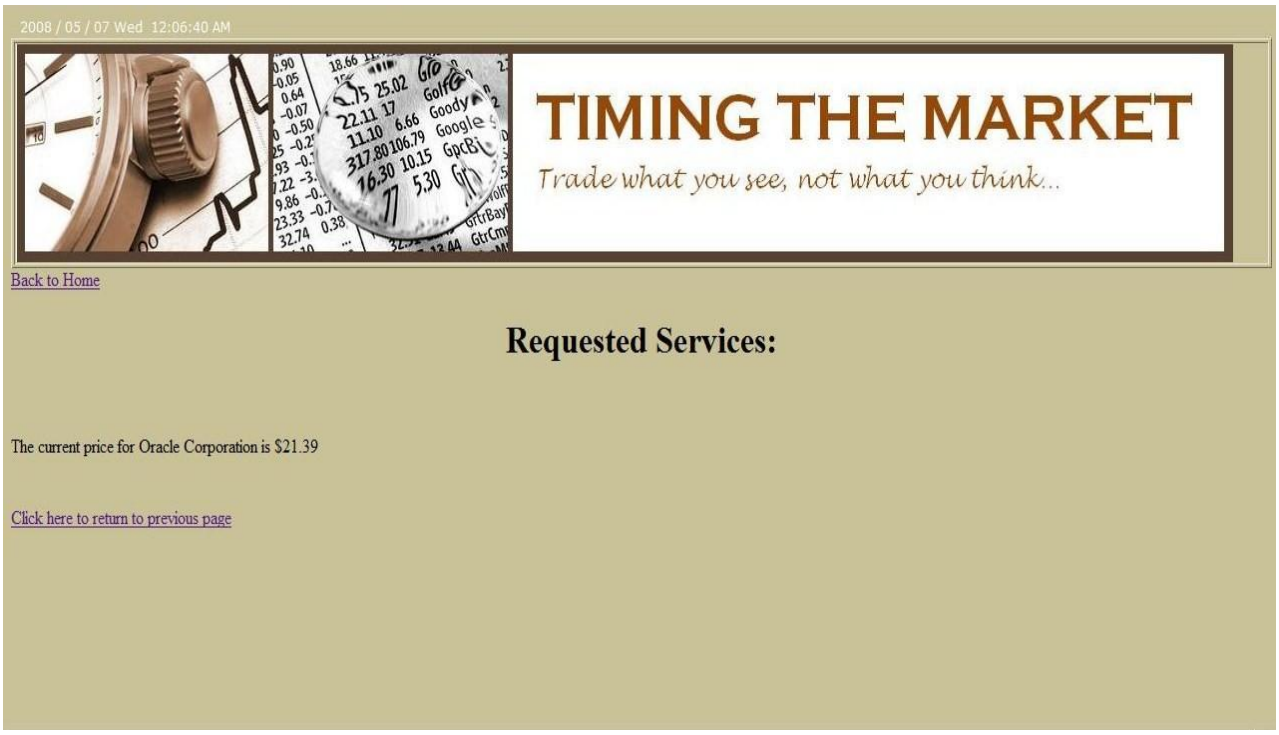
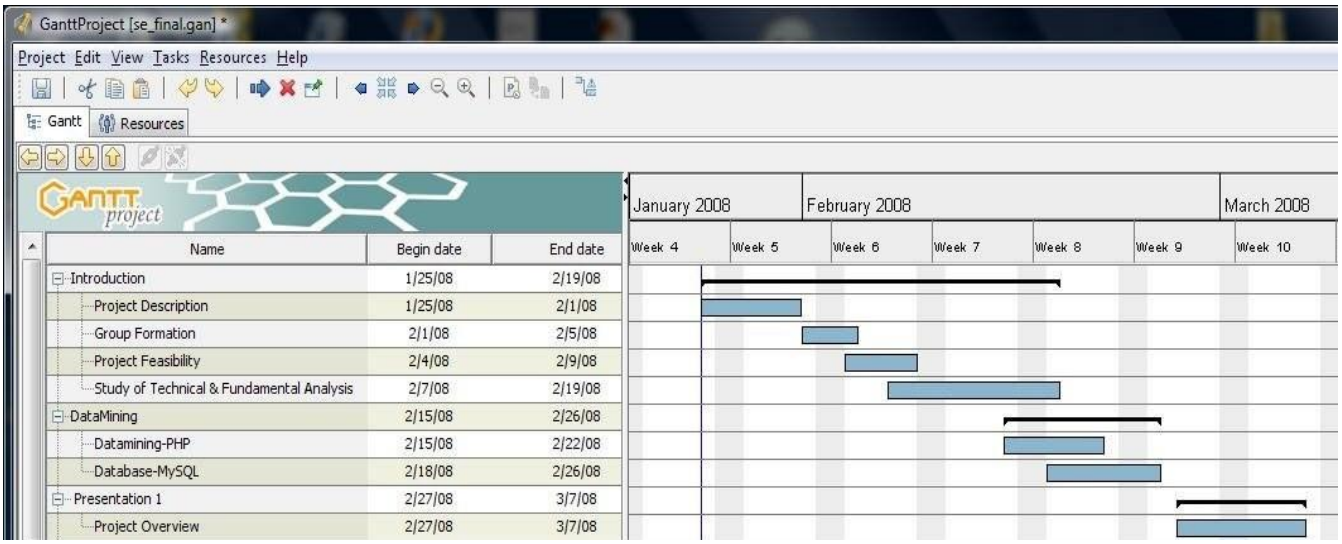
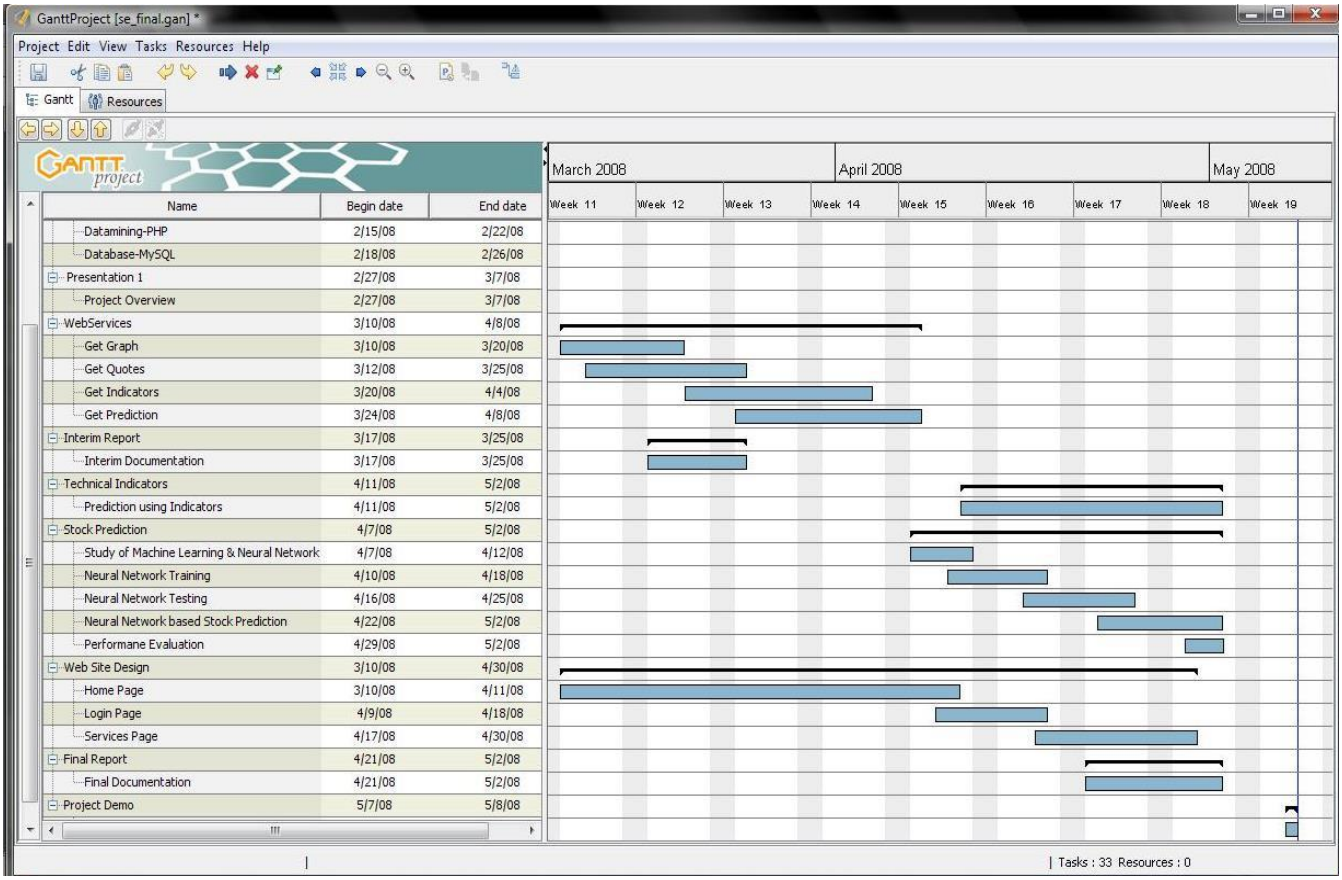


Fig 28: Results Page

HISTORY OF WORK

The Gantt chart: The document shows how the actual milestones and deadlines involved in both, planning and scheduling our project.





ACCOMPLISHMENTS

- **Satisfied target customer base:** The stock prediction model is ideal for small-time, daily and weekly investors who seek reliable recommendations for short term investments.

- **Dynamic data collection:** We have built a data feed algorithm which traverses financial web resources like Google finance and Yahoo finance; and collects data specific to the list of 25 companies for a time span of last one year, into the database. This algorithm runs every 15 minutes (Delayed) and updates the final entry consisting of opening, closing, high and low prices along with volume at the end of the day (EOD). One salient feature of the process is that for every single working day, the algorithm updates only the current day entry and does not need to dump or refill database.

- **Graphical support using Charts:** A customer can view the stock market history of any selected company in the form of a 2-D graph with the days of the previous year on the X-axis and corresponding closing price on the Y-axis. This visual display can help a customer to understand the dynamics of the stocks of an organization in a better way.

- **Friendly user interface:** We aim to educate the user on stock forecasting methodologies along with providing him with valuable prediction services. The designed website gathers a gamut of information like the latest news on stocks and companies, market indices, technology that went behind our prediction modules and the use of technical indicators. Navigating the website is kept easy and user can create his/her own account to access our services.

- **Authentication:** Every user has to register with us to gain access to services that we provide. This authentication procedure allows entry to serious investors only warding off prying softwares in the process.

- **Web services:** Most of the websites today provide web services to the customers. We are in step with them and have used the Axis software from the Apache foundation to create web services for our central prediction module and other secondary tasks.

- **Stock forecasting by machine learning:** We have employed the latest and one of the most precise prediction technology with underlying 3-layered neural networks in

our system to provide fair recommendations to end-user about buying or selling a particular stock. We successfully train the network using a wide range of real inputs from stock history and simulate the same for checking the reliability of our system. We have got high recognition rates for each company to the order of 70% to 90 %. Our machine is scheduled for running at the end of the day and prepares a set of predictions for each consecutive day. One can verify our claims by checking the predictions for previous days and comparing with prices which are already available. We also calculate the predicted price for any selected stock.

- **Secondary stock prediction modules:** We have implemented algorithms for 10 popular technical indicators in another module that works parallel to our central module. They are meant to provide more instantaneous prediction that might support the decision from neural network module.

SHORTCOMINGS

- **Need of a larger database:** Currently, our software is capable of handling data/stock history of a range of 20 selected companies for past one year. There is scope for increasing recognition rate by taking more historical data for a larger range of companies.

- **No user profile management:** Ideally, we would have maintained user profiles for each customer and provide personalized services like email alerts about the stocks cached by user in his/her profile.

- **No downloads:** Our services can be strictly accessed only online and hence, we would not be able to provide valuable services to our loyal customers in case of a server breakdown or small hacks in the program.

- **Fewer indicators:** We have a small set of 14 indicators to aid our prediction model. We aim to include more indicators in our software to concretize the decision given by our secondary prediction modules.

FUTURE WORK

- The system should track various stock movements simultaneously that might not be owned currently by customers and send alerts if it notices a favorable tilt in the graph that hints at immediate purchase.

- Developing customer profiles that should be maintained by the system and confidential customer data should be protected at all costs from prying software's.

- The system should regularly send emails or short messages to the customer informing him about the current trends and future scenarios. It could feed the user current news and conduct surveys to judge its performance.

- If time permits, the system may provide in-detail risk analysis of user portfolios to generate Sharpe Ratio to evaluate their current and future risk standing in the market and take appropriate corrective measures.

- Improving the refreshing rate of the stock database

- Trying to maintain the integrity of the database and other modules.

- Executing investors' requests as quickly as possible

REFERENCES

[1] http://www.taquote.com/public/download/techdefs.pdf

[2] Garth Garner, 'Prediction of Closing Stock Prices', Portland State University department of Electrical and Computer Engineering

[3] http://www.traders.com/Documentation/RESource_docs/Glossary/glossary_TZ

[4] http://en.wikipedia.org/wiki/Technical_Analysis_Software_(Finance)

[5] Using Neural Networks to Forecast Stock Market Prices, Ramon Lawrence

[6] MATLAB (v 7.1) Neural Network Toolbox Users Guide

[7] http://www.metaquotes.net/techanalysis/indicators/

[8] http://en.wikipedia.org/wiki/Technical_Analysis_Software_(Finance)

[9] http://www.bredemeyer.com

[10] http://www.agilemodeling.com/artifacts/useCaseDiagram.htm

[11] http://aspnet.4guysfromrolla.com/articles/100803-1.aspx

[12] http://www.15seconds.com/issue/010430.htm

[13] http://msdn.microsoft.com/en-us/library/ms972326.aspx

[14] http://msdn.microsoft.com/en-us/webservices/default.aspx

[15] http://www.php.net/

[16] http://www.w3schools.com/PHP/php_get.asp

[17] http://devzone.zend.com/article/1081-Using-cURL-and-libcurl-with-PHP

[18] http://www.php-mysql-tutorial.com/connect-to-mysql-using-php.php

[19] http://phpwebservices.blogspot.com/

[20] http://www.w3schools.com/html/default.asp

[21] http://www.w3schools.com/xml/default.asp