

1.INTRODUCTION

The rapid progress in digital data acquisition has led to the fast-growing amount of data stored in databases, data warehouses, or other kinds of data repositories. Although valuable information may be hiding behind the data, the overwhelming data volume makes it difficult for human beings to extract them without powerful tools. Easy and quick availability of news information was not possible until the beginning of the last decade. In this age of information, news is now easily accessible, as content providers and content locators such as online news services have sprouted on the World Wide Web. Continuous availability of more news articles in digital form, the latest developments in Natural Language Processing (NLP) and the availability of faster computers lead to the question of how to extract more information out of news articles. Financial analysts who invest in stock markets usually are not aware of the stock market behaviour. They are facing the problem of stock trading as they do not know which stocks to buy and which to sell in order to gain more profits. All these users know that the progress of the stock market depends a lot on relevant news and they have to deal daily with vast amounts of information. They have to analyse all the news that appears in newspapers, magazines and other textual resources. But analysis of such a large amount of financial news and articles in order to extract useful knowledge exceeds human capabilities. Text mining techniques can help them automatically extract the useful knowledge out of textual resources. We would develop a system which is able to use text mining techniques to model the reaction of the stock market to news articles and predict their reactions. By doing so, the investors are able to foresee the future behaviour of their stocks when relevant news are released and act immediately upon them. As input we use real-time news articles and intra-day stock prices of some companies in Bombay Stock Exchange. The overall purpose of study can be summarised in the following research questions:

- How to predict the reaction of stock price trends using textual financial news?
- How data and text mining techniques help to generate this predictive model? In order to investigate the impact of news on a stock trend movement, we have to make a prediction model.

1.1 BACKGROUND KNOWLEDGE

Knowledge Discovery in Text (KDT) The term KDT is used to indicate the overall process of turning unstructured textual data into high level information and knowledge, while the term Text Mining is used for the step of the KDT process that deals with the extraction of patterns from textual data. By extending the definition of KDD, the following simple definition is given:

1.1.1 Knowledge Discovery in Text (KDT)

Is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in unstructured textual data. Text Mining (TM) also known as text data mining is a step in the KDT process consisting of particular data mining and Natural Language Processing (NLP) algorithms that produces a particular enumeration of patterns over a set of unstructured textual data. There are various definitions and terminologies for text mining provided by different researchers. KDT is a multi-step process, which includes all the tasks from gathering of documents to the visualisation and evaluation of the extracted information.

1.1.2 Influence of News Articles on Stock Market Market

Stock exchange news are special messages containing mainly economic and political information. Some of them are carrying information that is important for market prediction. There are various types of financial information sources on the Web which provide electronic versions of their daily issues. All these information sources contain global and regional political and economic news, citations from influential bankers and politicians, as well as recommendations from financial analysts.

Researchers confirm the reaction to news articles. They have shown that economic news always has a positive or negative effect on the number of traded stock. They used salient political and economic news as proxy for public information. They have found that both types of news have an impact on measures of trading activity including return volatility, price volatility, number of shares traded, and trading frequency. Klibanoff [2] investigate the relationship between closed-end country funds' prices and country-specific salient news. The news that occupies at least two columns wide on The New York Times front-page is considered salient news. They have found that there is a positive relationship between trading volume and salient news. Mitchell and Mulherin [3] use the daily number of headlines reported by Dow Jones as a measure of public information. Using daily data on stock returns and trading volume, they find that market activity is affected by the arrival of news. They report that salient news has a positive impact on absolute price changes. Berry and Howe [4], use the number of news released by Reuters News Service measured in per unit of time as a proxy for public information. In contrast to Mitchell and Mulherin [3], they look into the impact of news on intraday market activity. Their results suggest that there is a significant positive relationship between news arrivals and trading volume.

2. LITERATURE SURVEY

2.1 Introduction

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of

external support. This support can be obtained from senior programmers, from book or from websites. Before building the system, the above consideration is taken into account for developing the proposed system

2.2 Existing system

The rapid progress in digital data acquisition has led to the fast-growing amount of data stored in databases, data warehouses, or other kinds of data repositories. Although valuable information may be hiding behind the data, the overwhelming data volume makes it difficult for human beings to extract them without powerful tools. Easy and quick availability of news information was not possible until the beginning of the last decade. In this age of information, news is now easily accessible, as content providers and content locators such as online news services have sprouted on the World Wide Web. Continuous availability of more news articles in digital form, the latest developments in Natural Language Processing (NLP) and the availability of faster computers lead to the question how to extract more information out of news articles. Financial analysts who invest in stock markets usually are not aware of the stock market behaviour. They are facing the problem of stock trading as they do not know which stocks to buy and which to sell in order to gain more profits. All these users know that the progress of the stock market depends a lot on relevant news and they have to deal daily with vast amount of information. They have to analyse all the news that appears on newspapers, magazines and other textual resources. But analysis of such amount of financial news and articles in order to extract useful knowledge exceeds human capabilities. Text mining techniques can help them automatically extracting the useful knowledge out of textual resources. We would develop a system which is able to use text mining techniques to model the reaction of the stock market to news articles and predict their reactions. By doing so, the investors are able to foresee the future behaviour of their stocks when relevant news are released and act immediately upon them. As input we use real-time news articles and intra-day stock prices of some companies in Bombay Stock Exchange.

The overall purpose of study can be summarised in the following research questions:

- How to predict the reaction of stock price trend using textual financial news?
- How data and text mining techniques help to generate this predictive model? In order to investigate the impact of news on a stock trend movement, we have to make a prediction model.

2.3 PROPOSED WORK

Methodology for the NLP module To exactly predict the stock price is a very complex task till the date. Here we are proposing to make a prediction based on news articles using one of the Text Mining

concepts like sentiment analysis. We would like to make the prediction system for Indian Stock market. Implementation steps to be followed to make a prediction system are:

1. Gathering of news articles.
2. Perform sentiment analysis on news articles
3. Get Polarity of the text
4. Make a prediction based on current stock price and calculated polarity of the text

To Get the News Articles To collect the news articles R.S.S feed is the main source. As R.S.S feed is used for news article collection process. Here the Times of India's R.S.S feed is used for business and market related news. It will give results by retrieving top news of Indian stock market. We have to just specify the R.S.S feed address in our code. **To Perform Sentiment Analysis and Get Polarity of the text** Sentiment analysis (also known as opinion mining) refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level - whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. For sentiment analysis and calculating polarity of text two things are used:

1. POS tagger
2. SentiWordNet
- 3.0.0 POS Tagger

A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word (and other token), such as noun, verb, adjective, etc., although generally computational applications use more fine grained POS tags like 'noun plural'. This software is a Java implementation of the log-linear part-of-speech taggers. Part-of-speech tagging (POS tagging or POST), also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition, as well as its context—i.e. relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, etc. POS-tagging algorithms fall into two distinctive groups: rule-based and stochastic. E. Brill's tagger, one of the first and widely used English POS-taggers, employs rule-based algorithms. Part-of-speech tagging is harder than just having a list of words and their parts of speech, because some words can represent more than one part of speech at different times, and because some parts of speech are complex or unspoken. This is not rare in natural languages (as opposed to many artificial languages), a large percentage of word forms are ambiguous. In part-of-speech tagging by computer, it is typical to distinguish from 50 to 150 separate parts of

speech for English. For example, NN for singular common nouns, NNS for plural common nouns, NP for singular proper nouns. Several downloads are available. The basic download contains two trained tagger models for English. The POS tagger which I have used is developed by Stanford University natural language processing group [8]; it is licensed under the GNU general public licence as it is an open source.

3.FEASIBILITY STUDY

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organisation. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running systems. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility Economic Feasibility

3.1 TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Does the proposed equipment have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed? Are there technical guarantees of accuracy, reliability, ease of access and data security?

3.2 OPERATIONAL FEASIBILITY

User-friendly Customers will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user friendly to the Client. Reliability The package will pick-up current transactions online. Regarding the old transactions, User will enter them into the system. Security The web server and database server should be protected from hacking, virus etc Portability The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on

Windows and Linux o/s. Hence portability problems will not arise. Availability This software will be available always. Maintainability The system called the wheels uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end. The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

3.3 ECONOMIC FEASIBILITY

The computerised system takes care of the present existing system's data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports. It should be built as a web based application with separate web server and database server. This is required as the activities are spread throughout the organisation and the customer wants a centralised database. Further some of the linked transactions take place in different locations. Open source software like TOMCAT, JAVA, Mysql and Linux is used to minimise the cost for the Customer

4.SYSTEM ANALYSIS:

4.1 HARDWARE REQUIREMENTS:

Hardware – Pentium

Speed - 1.1 GHz

RAM - 1GB Hard

Disk - 20 GB

Floppy Drive - 1.44 MB

Key Board - Standard Windows Keyboard

Mouse - Two or Three Button Mouse

Monitor - SVGA

4.2 SOFTWARE REQUIREMENTS:

Operating System : Windows

Technology : Java and J2EE Web

Technologies : Html, JavaScript, CSS

IDE : My Eclipse, Java Version : J2SDK1.5

Web Server : Tomcat

Tool kit : Android Phone

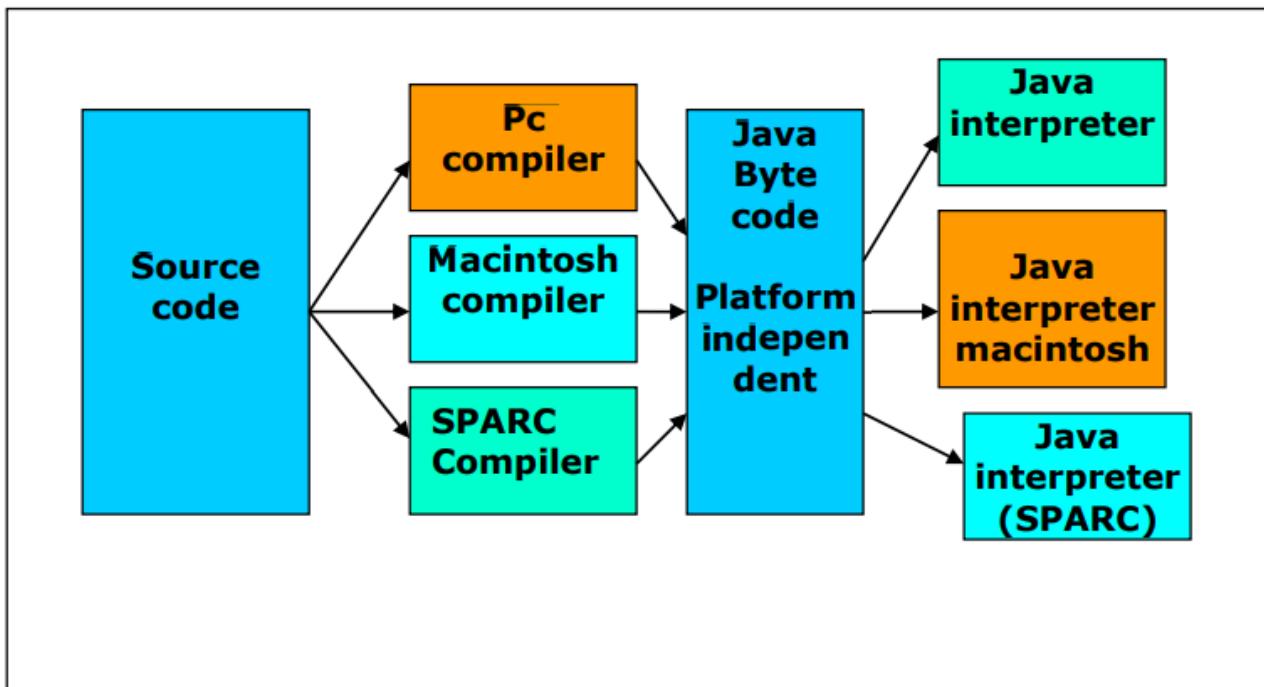
Database : My SQL

5. SYSTEM REQUIREMENTS

5.1 About Java:

Initially the language was called “oak” but it was renamed as “java” in 1995. The primary motivation of this language was the need for a platform-independent(i.e. architecture neutral)language that could be used to create software to be embedded in various consumer electronic devices. Java is a programmer’s language Java is cohesive and consistent Except for those constraints imposed by the Internet environment. Java gives the programmer, full control Finally Java is to Internet Programming where C was to System Programming. Importance of Java to the Internet has had a profound effect on the Internet. This is because; java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the server and the personal computer. They are passive information and Dynamic active programs. in the areas of Security and probability. But Java addresses these concerns and by doing so, has opened the door to an exciting new form of program called the Applet. Applications and applets. An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important. An Applet In an application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change. Java Architecture Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet. Compilation of code When you compile the code, the Java compiler creates machine code (called bytecode)for a hypothetical machine called Java Virtual Machine(JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of probability. The code is written and compiled for one machine and interpreted on all machines .This machine is called Java Virtual Machine.

Compiling and interpreting java source code.



During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through the internet and run the Applets.

5.2 Simple:

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer. Learning Java will be oriented towards features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

5.3 Object oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean, usable, pragmatic approach to objects.

5.4 Robust

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs. Was

given a high priority in the design of Java. Java is a strictly typed language; it checks your code at compile time and runtime. Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all runtime errors can and should be managed by your program.

SoftWare Requirements

- Operating System : Windows Family or higher version
- Techniques : JDK 1.7
- Databases : Mysql
- Server :Apache Tomcat

Hardware Requirements:

- Processor:: Pentium-III (or) Higher
- Ram:: 64MB (or) Higher
- Cache:: 512MB
- Hard disk:: 10GB

5.5 Servlets/JSP

5.5.1 INTRODUCTION

A Servlet Is a generic server extension. a Java class that can be loaded Dynamically to expand the functionality of a server. Servlets are commonly used with web servers. Where they can take the place of CGI scripts. A servlet is similar to proprietary server extension, except that it runs inside a Java Virtual Machine (JVM) on the server, so it is safe and portable. Servlets operate solely within the domain of the server. Unlike CGI and FastCGI, which use multiple processes to handle separate programs or separate requests, separate threads within web server processes handle all servlets. This means that servlets are all efficient and scalable.

Servlets are portable; both across operating systems and also across web servers. Java Servlets offer the best possible platform for web application development. Servlets are used as replacement for CGI scripts on a web server, they can extend any sort of server such as a mail server that allows servlets to extend its functionality perhaps by performing a virus scan on all attached documents or handling mail filtering tasks. Servlets provide a Java-based solution used to address the problems currently

associated with doing server-side programming including inextensible scripting solutions platform-specific API's and incomplete interface. Servlets are objects that conform to a specific interface that can be plugged into a Java-based server. Servlets are to the server-side what applets are to the server side what applets are to the client-side-object byte codes that can be dynamically loaded off the net. They differ from applets in that they are faceless objects (without graphics or a GUI component). They serve as platform independent, dynamically loadable, plugable helper byte code objects on the server side that can be used to dynamically extend server-side functionality. For example an HTTP servlet can be used to generate dynamic HTML content when you use servlets to do dynamic content you get the following advantages: They're faster and cleaner than CGI scripts. They use a standard API (the servlet API) They provide all the advantages of Java (run on a variety of servers without needing to be rewritten)

5.5.2 Attractiveness Servlets:

There are many features of servlets that make them easy and attractive to use these include: Easily configured using the GUI-based Admin tool] Can be Loaded and Invoked from a local disk or remotely across the network. Can be linked together or chained, so that one servlet can call another servlet, or several servlets in sequence. Can be called dynamically from within HTML, pages using server-side include-tags. Are secure-even when downloading across the network, the servlet security model and servlet and box protect your system from unfriendly behavior.

5.5.3 Advantages of the servlet API

One of the great advantages of the servlet API is protocol independence. It assumes nothing about: The protocol being used to transmit on the net How it is loaded The server environment it will be running in These quantities are important, because it allows the Servlet API to be embedded in many different kinds of servers. There are other advantages to the servlet API as well These include: It's extensible-you can inherit all your functionality from the base classes made available to you It's simple, small, and easy to use.

5.5.4 Features of Servlets:

Servlets are persistent. Servlets are loaded only by the web server and can maintain services between requests. Servlets are fast. Since servlets only need to be loaded once, they offer much better performance over their CGI counterparts. Servlets are platform independent. Servlets are extensible. Java is a robust, object-oriented programming language, which easily can be extended to suit your needs. Servlets are secure. Servlets are used with a variety of clients. Servlets are classes and interfaces from two packages, javax.servlet and javax.servlet.http. The java.servlet package contains classes to support generic, protocol-independent servlets. The classes in the javax.servlet.http package To and

HTTP specific functionality extend these classes Every servlet must implement the javax.servlet interface. Most servlets implement it by extending one of two classes javax.servlet.GenericServlet or javax.servlet.http.HttpServlet. A protocol-independent servlet should subclass Generic-Servlet. while an Http servlet should subclass HttpServlet, which is itself a subclass of Generic-servlet with added HTTP-specific functionality. Unlike a java program, a servlet does not have a main() method, Instead the server in the process of handling requests invoke certain methods of a servlet. Each time the server dispatches a request to a servlet, it invokes the servlets Service() method, A generic servlet should override its service() method to handle requests as appropriate for the servlet. The service() accepts two parameters a request object and a response object .The request object tells the servlet about the request, while the response object is used to return a response InContrast.anHttp servlet usually does not override the service() method.Instead it overrides doGet() to handle GET requests and doPost() to handle Post requests. An Http servlet can override either or both of these modules the service() method of HttpServlet handles the setup and dispatching to all the doXXX() methods. which is why it usually should not be overridden The remainders in the javax.servlet and javax.servlet.http.package are largely support classes .The ServletRequest and ServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse in javax.servlet.http provide access a HTTP requests and responses . The javax.servlet.http provide contains an HttpSession class that provides built-in session tracking functionality and Cookie class that allows quickly setup and processing HttpCookies

5.5.5 Loading Servlets:

Servlets can be loaded from their places. From a directory that is on the CLASSPATH. The CLASSPATH of the JavaWebServer includes service root/classes/, which is where the system classes reside

From the <SERVICE_ROOT/servlets/directory. This is not in the server's classpath. A class loader is used to create servlets form this directory. New servlets can be added-existing servlets can be recompiled and the server will notice these changes. From a remote location. For this a code base like http://nine.eng/classes/foo/ is required in addition to the servlet's class name. Refer to the admin Gui docs on servlet section to see how to set this up.

Loading Remote Servlets

Remote servlets can be loaded by:

Configuring the admin Tool to setup automatic loading of remote servlets.

Selecting up server side include tags in .html files

Defining a filter chain Configuration

5.5.6 Invoking Servlets:

A servlet invoker is a servlet that invokes the “server” method on a named servlet. If the servlet is not loaded in the server, then the invoker first loads the servlet (either from local disk or from the network) and then invokes the “service” method. Also like applets, local servlets in the server can be identified by just the class name. In other words, if a servlet name is not absolute, it is treated as local. A Client can Invoke Servlets in the Following Ways: The client can ask for a document that is served by the servlet. The client (browser) can invoke the servlet directly using a URL, once it has been mapped using the SERVLET ALIASES Section of the admin GUI. The servlet can be invoked through server side include tags. The servlet can be invoked by placing it in the servlets/directory. The servlet can be invoked by using it in a filter chain.

5.5.7 The Servlet Life Cycle:-

The Servlet life cycle is one of the most exciting features of Servlets. This life cycle is a powerful hybrid of the life cycles used in CGI programming and lower level NSAPI and ISAPI programming. The servlet life cycle allows servlet engines to address both the performance and resource problems of CGI and the security concerns of low level server API programming. Servlet life cycle is highly flexible. Servers have significant leeway in how they choose to support servlets. The only hard and fast rule is that a servlet engine must conform to the following life cycle contract: Create and initialise the servlets. Handle zero or more service from clients. Destroy the servlet and then garbage collects it. It's perfectly legal for a servlet to be loaded, created and initialised in its own JVM, only to be destroyed and garbage collected without handling any client request or after handling just one request. The most common and most sensible life cycle implementations for HTTP servlets are: Single java virtual machine and astatine persistence.

5.5.8 Init and Destroy:-

Just like Applets, servlets can define init() and destroy() methods. A servlet's init(ServiceConfig) method is called by the server immediately after the server constructs the servlet's instance. Depending on the server and its configuration, this can be at any of these times: When the server starts. When the servlet is first requested, just before the service() method is invoked. At the request of the server administrator. In any case, init() is guaranteed to be called before the servlet handles its first request. The init() method is typically used to perform servlet initialization, creating or loading objects that are used by the servlet in handling of its request. In order to provide a new servlet any information about itself and its environment, a server has to call a servlet's init() method and pass an object that

implement the ServletConfig interface. This ServletConfig object supplies a servlet with information about its initialization parameters. These parameters are given to the servlets and are not associated with any single request. They can specify initial values, such as where a counter should begin counting, or default values, perhaps a template to use when not specified by the request. The server calls a servlet's destroy() method when the servlet is about to be unloaded. In the destroy() method, a servlet should free any resources it has acquired that will not be garbage collected. The destroy() method also gives a servlet a chance to write out its unsaved, cached information or any persistent information that should be read during the next call to init()

5.5.9 Session Tracking:

HTTP is a stateless protocol, it provides no way for a server to recognize that a sequence of requests is all from the same client. This causes a problem for application such as shopping cart applications. Even in chat application server can't know exactly who's making a request of several clients. The solution for this is for client to introduce itself as it makes each request. Each client needs to provide a unique identifier that lets the server identify it, or it needs to give some information that the server can use to properly handle the request. There are several ways to send this introductory information with each request Such as:

5.6 USER AUTHORIZATION:

One way to perform session tracking is to leverage the information that comes with User authorization.

When a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. After the client logs in, the username is available to a servlet through getRemoteUser(). We can use the username to track the session. Once a user has logged in, the browser remembers her username and resends the name and password as the user views new pages on the site. A servlet can identify the user through her username and they're by Track her session. The biggest advantage of using user authorization to perform session tracking is that it's easy to implement. Simply tell the protect a set of pages, and use getRemoteUser() to identify each client. Another advantage is that the technique works even when the user accesses your site from or exists her browser before coming back. The biggest disadvantage of user authrization is that it requires each user to register for an account and then log in in each time the starts visiting your site. Most users will tolerate registering and lagging in as a necessary evil when they are accessing sensitive information, but its all overkill for simple session tracking. Other problem with user authorization is that a user cannot simultaneously maintain more than one session at the same site.

Hidden Form Fields:

One way to support anonymous session tracking is to use hidden form fields. As the name implies, these are fields added to an HTML form that are not displayed in the client's browser. They are sent back to the server when the form that contains them is submitted. In a sense, hidden form fields define constant variables for a form. To a servlet receiving a submitted form, there is no difference between a hidden field and a visible field. As more and more information is associated with a client's session, it can become burdensome to pass it all using hidden form fields. In these situations, it's possible to pass on just a unique session ID that identifies a particular client's session. That session ID can be associated with complete information about its session that is stored on the server. The advantage of hidden form fields is their ubiquity and support for anonymity. Hidden fields are supported in all the popular browsers; they demand no special server requirements, and they can be used with clients that haven't registered or logged in. The major disadvantage with this technique, however, is that it works only for a sequence of dynamically generated forms. The technique breaks down immediately with static documents, emailed documents, bookmarked documents, and browser shutdowns.

URL Rewriting:

URL rewriting is another way to support anonymous session tracking. With URL rewriting, every local URL the user might click on is dynamically modified or rewritten, to include extra information. The extra information can be in the form of extra path information, added parameters, or some custom, server-specific URL change. Due to the limited space available in rewriting a URL, the extra information is usually limited to a unique session. Each rewriting technique has its own advantages and disadvantages. Using extra path information works on all servers, and it works as a target for forms that use both the Get and Post methods. It does not work well if the servlet has to use the extra path information as true path information. The advantages and disadvantages of URL rewriting closely match those of hidden form fields. The major difference is that URL rewriting works for all dynamically created documents, such as the Help servlet, not just forms. With the right server support, custom URL rewriting can even work for static documents.

Persistent Cookies:

A fourth technique to perform session tracking involves persistent cookies. A cookie is a bit of information sent by a web server to a browser that can later be read back from that browser. When a browser receives a cookie, it saves the cookie and thereafter sends the cookie back to the server each time it accesses a page on that server, subject to certain rules. Because a cookie's value can uniquely identify a client, cookies are often used for session tracking. Persistent cookies offer an elegant, efficient, and easy way to implement session tracking. Cookies provide an automatic introduction for

each request as we could hope for. For each request, a cookie can automatically provide a client's session ID or perhaps a list of clients performance. The ability to customise cookies gives them extra power and versatility. The biggest problem with cookies is that browsers don't always accept cookies sometimes this is because the browser doesn't support cookies. More often its because The browser doesn't support cookies. More often its because the user has specifically configured the browser to refuse cookies. The power of servers: The power of servlets is nothing but the advantages of servlets over other approaches, which include portability, power, efficiency, endurance, safety, elegance, integration, extensibility and flexibility.

Portability:

As servlets are written in java and conform to a well defined and widely accepted API.they are highly portable across operating systems and across server implementation We can develop a servlet on a windows NT machine running the java web server and later deploy it effortlessly on a high-end Unix server running apache. With servlets we can really "write once, serve every where" Servlet portability is not the stumbling block it so often is with applets, for two reasons First,Servlet portability is not mandatory i.e. servlets has to work only on server machines that we are using for development and deployment Second, servlets avoid the most error-prone and inconstancy implemented portion of the java languages.

Power:

Servlets can harness the full power of the core java. API's: such as Networking and Url access, multithreading, image manipulation, data compression, data base connectivity, internationalisation, remote method invocation(RMI) CORBA connectivity, and object serialisation, among others.

Efficiency And Endurance:

Servlet invocation is highly efficient, Once a servlet is loaded it generally remains in the server's memory as a single object instance, There after the server invokes the servlet to handle a request using a simple, light weighted method invocation .Unlike the CGI, there's no process to spawn or interpreter to invoke, so the servlet can begin handling the request almost immediately, Multiple, concurrent requests are handled the request almost immediately. Multiple, concurrent requests are handled by separate threads, so servlets are highly scalable. Servlets in general are enduring objects. Because a servlets stays in the server's memory as a single object instance. it automatically maintains its state and can hold onto external resources, such as database connections.

Safety:

Servlets support safe programming practices on a number of levels. As they are written in java, servlets inherit the strong type safety of the java language. In addition the servlet API is implemented to be type safe. Java's automatic garbage collection and lack of pointers mean that servlets are generally safe from memory management problems like dangling pointers invalid pointer references and memory leaks.

Elegance:

The elegance of the servlet code is striking. Servlet code is clean, object oriented modular and amazingly simple one reason for this simplicity is the served API itself. Which includes methods and classes to handle many of the routine chores of servlet development. Even advanced to operations like cookie handling and session tracking tracking are abstracted int convenient classes.

Integration:

Servlets are tightly integrated with the server. This integration allows a servlet to cooperate with the server in two ways . for e.g.: a servlet can use the server to translate file paths, perform logging, check authorization, perform MIME type mapping and in some cases even add users to the server's user database.

Extensibility and Flexibility:

The servlet API is designed to be easily extensible. As it stands today the API includes classes that are optimised for HTTP servlets. But later it can be extended and optimised for another type of servlets. It is also possible that its support for HTTP servlets could be further enhanced. Servlets are also quite flexible, Sun also introduced java server pages. which offer a way to write snippets of servlet code directly with in a static HTML page using syntax similar to Microsoft's Active server pages(ASP)

5.7 JDBC

What is JDBC?

Any relational database. One can write a single program using the JDBC API, and the JDBC is a Java API for executing SQL Statements(As a point of interest JDBC is trademarked name and is not an acronym; nevertheless, Jdbc is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java Programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API Using JDBC, it is easy to send SQL statements to virtually program will be able to send SQL statements to the appropriate database. The Combination of Java and JDBC lets a programmer writes it once and run it anywhere

What Does JDBC Do?

Simply put,JDBC makes it possible to do three things

- o Establish a connection with a database
- o Send SQL statements o Process the results
- o JDBC Driver Types
 - o The JDBC drivers that we are aware of this time fit into one of four categories
 - o JDBC-ODBC Bridge plus ODBC driver
 - o Native-API party-java driver
 - o JDBC-Net pure java driver
 - o Native-protocol pure Java driver An individual database system is accessed via a specific JDBC driver that implements the java.sql.Driver interface. Drivers exist for nearly all-popular RDBMS systems, through few are available for free. Sun bundles a free JDBC ODBC bridge driver with the JDK to allow access to a standard ODBC,data sources, such as a Microsoft Access database, Sun advises against using the bridge driver for anything other than development and very limited development. JDBC drivers are available for most database platforms, from a number of vendors and in a number of different flavours. There are four driver categorised.

Type 01-JDBC-ODBC Bridge Driver :

Type 01 drivers use a bridge technology to connect a java client to an ODBC database service. Sun's JDBC-ODBC bridge is the most common type 01 driver. These drivers implemented using native code.

Type 02-Native-API party-java

Driver Type 02 drivers wrap a thin layer of java around database-specific native code libraries for Oracle databases, the native code libraries might be based on the OCI(Oracle call Interface) libraries, which were originally designed for c/c++ programmers, Because type-02 drivers are implemented using native code. in some cases they have better performance than their all-java counter parts. They add an element of risk, however, because a defect in a driver's native code section can crash the entire server

Type 03-Net-Protocol All-Java Driver

Type 03 drivers communicate via a generic network protocol to a piece of custom middleware.

The middleware component might use any type of driver to provide the actual database access. These drivers are all java, which makes them useful for applet deployment and safe for servlet deployment

Type-04-native-protocol All-java Driver Type 04 drivers are the most direct of the lot. Written entirely

in java, Type 04 drivers understand database-specific networking protocols and can access the database directly without any additional software

JDBC-ODBC Bridge:

If possible use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge(that is, the Bridge native library, the ODBC driver manager library, library, the ODBC driver library, and the database client library)

WHAT IS The JDBC-ODBC Bridge ?

The JDBC-ODBC Bridge is a Jdbc driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is joint development of Intersolv and Java Soft

5.8 Oracle

Oracle is a relational database management system, which organises data in the form of tables. Oracle is one of many database servers based on RDBMS model, which manages a series of data that attends three specific things-data structures, data integrity and data manipulation. With oracle cooperative server technology we can realise the benefits of open, relational systems for all the applications. Oracle makes efficient use of all systems resources, on all hardware architecture; to deliver unmatched performance, price performance and scalability. Any DBMS to be called as RDBMS has to satisfy Dr.E.F.Codd's rules.

Features of Oracle:

Portable The Oracle RDBMS is available on wide range of platforms ranging from PCs to super computers and as a multi user loadable module for Novel NetWare, if you develop application on system you can run the same application on other systems without any modifications. Compatible Oracle commands can be used for communicating with IBM DB2 mainframe RDBMS that is different from Oracle, which is Oracle compatible with DB2. Oracle RDBMS is a high performance fault tolerant DBMS, which is specially designed for online transaction processing and for handling large database applications. Multithreaded Server Architecture Oracle adaptable multithreaded server architecture delivers scalable high performance for very large number of users on all hardware architecture including symmetric multiprocessors (sumps) and loosely coupled multiprocessors. Performance is achieved by eliminating CPU, I/O, memory and operating system bottlenecks and by

optimising the Oracle DBMS server code to eliminate all internal bottlenecks. Oracle has become the most popular RDBMS in the market because of its ease of use • Client/server architecture. • Data independence. • Ensuring data integrity and data security. • Managing data concurrency. • Parallel processing support for speed up data entry and online transaction processing used for applications. • DB procedures, functions and packages. Dr.E.F.Codd's Rules These rules are used for valuating a product to be called as relational database management systems. Out of 12 rules, a RDBMS product should satisfy at least 8 rules + rule called rule 0 that must be satisfied. RULE 0: Foundation Rule For any system to be advertised as, or claimed to be relational DBMS should manage database with in it self, with out using an external language.

RULE 1: Information Rule All information in relational database is represented at logical level in only one way as values in tables.

RULE 2: Guaranteed Access Each and every data in a relational database is guaranteed to be logically accessibility by using to a combination of table name, primary key value and column name.

RULE 3: Systematic Treatment of Null Values Null values are supported for representing missing information and inapplicable information. They must be handled in systematic way, independent of data types.

RULE 4: Dynamic Online Catalog based Relation Model The database description is represented at the logical level in the same way as ordinary data so that authorised users can apply the same relational language to its interrogation as they do to the regular data.

RULE 5: Comprehensive Data Sub Language A relational system may support several languages and various models of terminal use. However there must be one language whose statement can express all of the following: Data Definitions, View Definitions, Data Manipulations, Integrity, Constraints, Authorization and transaction boundaries.

RULE 6: View Updating Any view that is theoretical can be updatable if changes can be made to the tables that effect the desired changes in the view.

RULE 7: High level Update, Insert and Delete The capability of handling a base relational or derived relational as a single operand applies not only retrieval of data also to its insertion, updating, and deletion.

RULE 8: Physical Data Independence Application program and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access method.

RULE 9: Logical Data Independence Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access methods.

RULE 10: Integrity Independence Integrity constraints specific to particular database must be definable in the relational data stored in the catalogue, not in application program.

RULE 11: Distributed Independence Whether or not a system supports database distribution, it must have a data sublanguage that can support distributed databases without changing the application program.

RULE 12: Non Subversion If a relational system has low level language, that low language cannot use to subversion or by pass the integrity rules and constraints expressed in the higher level relational language. Oracle supports the following Codd's Rules

Rule 1: Information Rule (Representation of information)-YES.

Rule 2: Guaranteed Access-YES.

Rule 3: Systematic treatment of Null values-YES.

Rule 4: Dynamic on-line catalogue-based Relational Model-YES.

Rule 5: Comprehensive data sub language-YES.

Rule 6: View Updating-PARTIAL.

Rule 7: High-level Update, Insert and Delete-YES.

Rule 8: Physical data Independence-PARTIAL.

Rule 9: Logical data Independence-PARTIAL.

Rule 10: Integrity Independence-PARTIAL.

Rule 11: Distributed Independence-YES.

Rule 12: Non-subversion-YES.

5.9 HTML :

Hypertext Markup Language (HTML), the language of the world wide web(WWW), allows users to produce web pages that include text, graphics and pointers to other web pages (Hyperlinks). HTML is not a programming language but it is an application of ISO Standard 8879, SGML(Standard Generalised Markup Language), but Specialized to hypertext and adapted to the Web. The idea behind Hypertext is to point from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of items enclosed within the elements that should be displayed. Hyperlinks are underlined or emphasised words that lead to other documents or some portions of the same document. HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop. HTML provides tags(special codes) to make the document look attractive. HTML provides are not case-sensitive. Using graphics, fonts, different sizes, colour, etc.. can enhance the presentation of the document. Anything that is not a tag is part of the document itself. Basic HTML Tags: Specific Comments. Creates Hypertext links. Creates hypertext links. Formats text in large-font contains all tags and text in the HTML-document

Creates Text

.....
Definition of a term.

creates table indicates table data in a table. designates a table row creates a heading in a table

ADVANTAGES:-

A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information. HTML is platform independent HTML tags are not case-sensitive.

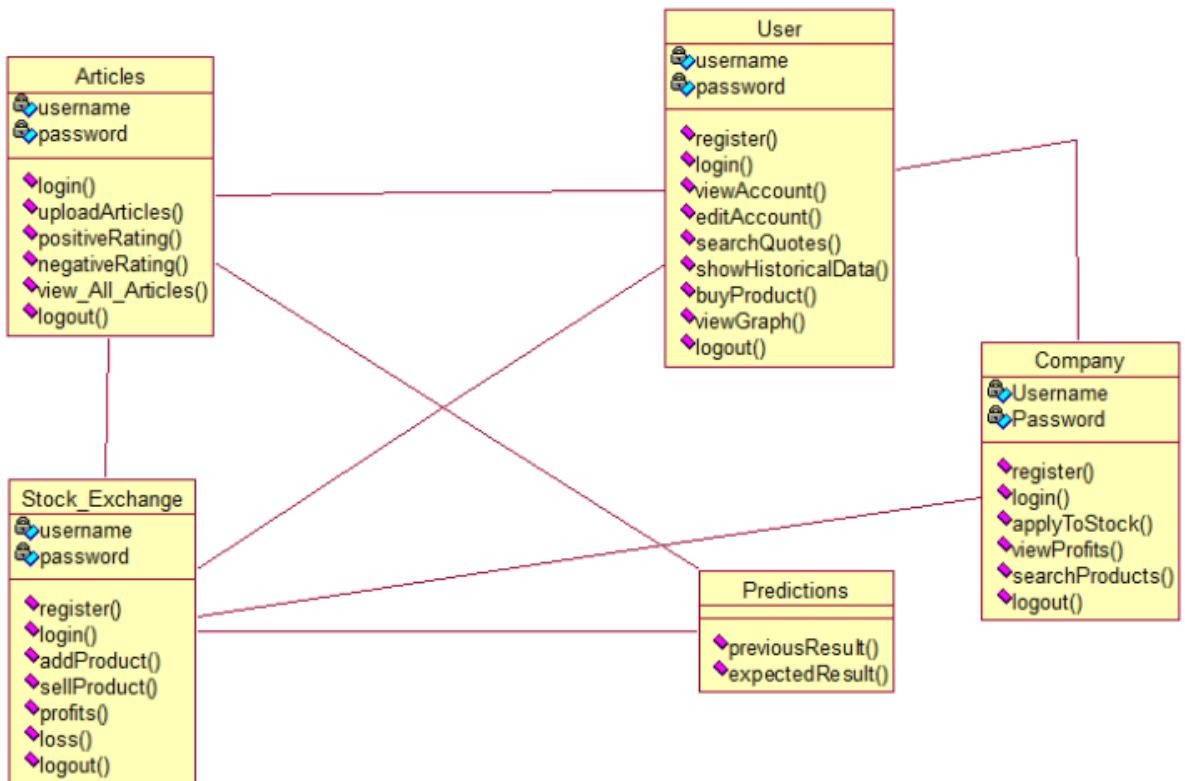
5.10 JAVA SCRIPT

The Java Script Language JavaScript is a compact , object-based scripting language for developing client and server internet applications. Netscape Navigator 2.0 interprets JavaScript statements embedded directly in an HTML page. and Livewire enables you to create server-based applications similar to common gateway interface(cgi) programs. In a client application for Navigator, JavaScript statements embedded in an HTML Page can recognize and respond to user events such as mouse clicks form Input, and page navigation. For example, you can write a JavaScript function to verify that users enter valid information into a form requesting a telephone number or zip code . Without any network transmission, an Html page with embedded Java Script can interpret the entered text and alert the user with a message dialog if the input is invalid or you can use JavaScript to perform an action (such as play an audio file, execute an applet, or communicate with a plug-in) in response to the user opening or exiting a page.

6.SYSTEM DESIGN

Stock market prediction UML Diagrams:

6.1 Class diagram:



6.2 STUDY OF THE SYSTEM

To provide flexibility to the users, the interfaces have been developed that are accessible through a browser. The GUI'S at the top level have been categorised as

1. Administrative user interface
2. The operational or generic user interface The ‘administrative user interface’ concentrates on the consistent information that is practically part of the organisational activities and which needs proper authentication for the data collection. These interfaces help the administrators with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities. The ‘operational or generic user interface’ helps the end users of the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information in a customised manner as per the included flexibilities

6.3 INPUT & OUTPUT REPRESENTATION

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

6.3.1 INPUT STAGES:

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

6.3.2 INPUT TYPES:

It is necessary to determine the various types of inputs. Inputs can be categorised as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are the computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

6.3.3 INPUT MEDIA:

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As Input data is to be directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

6.3.4 OUTPUT DESIGN:

In general are:

- External Outputs whose destination is outside the organisation.
- Internal Outputs whose destination is within organisation and they are the User's main interface with the computer. Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly with the system.

6.3.5 OUTPUT DEFINITION

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output It is not always desirable to print or display data as it is held on a computer.

It should be decided as to which form of the output is the most suitable.

For Example

- Will decimal points need to be inserted
- Should leading zeros be suppressed.

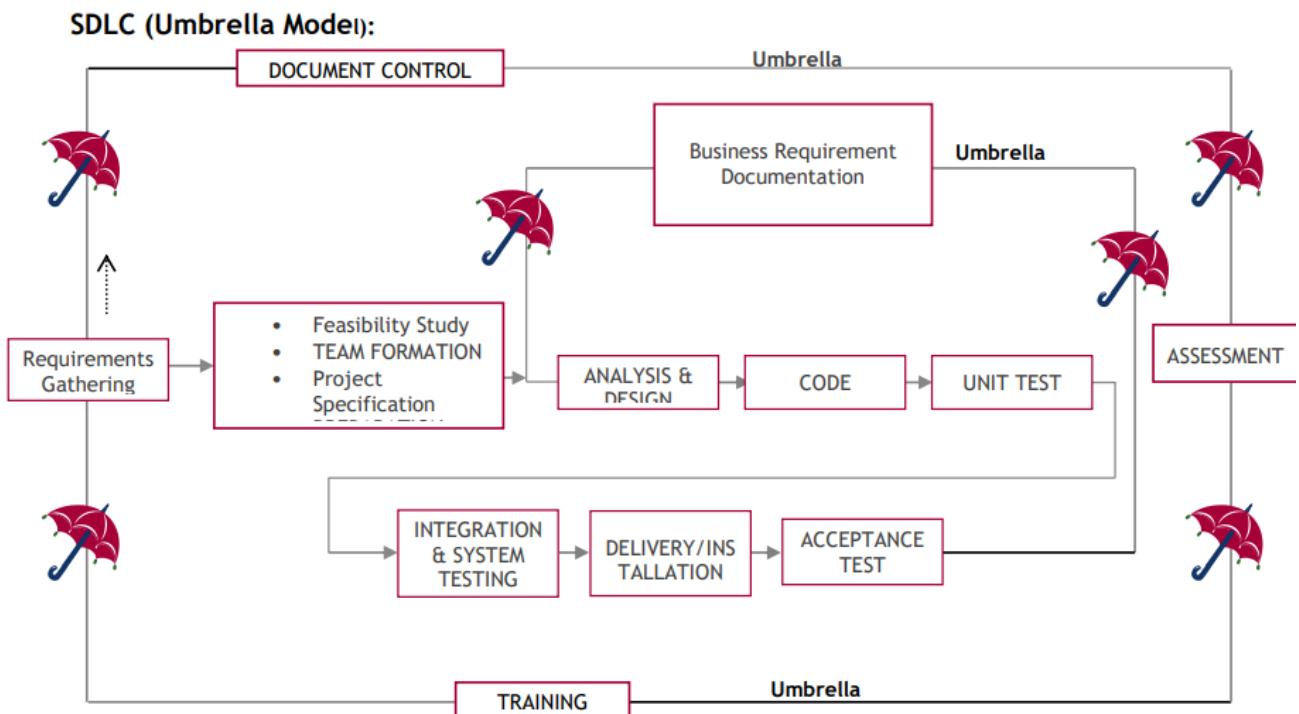
6.3.6 OUTPUT MEDIA:

In the next stage it is to be decided which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

- The suitability for the device to the particular application.
- The need for a hard copy.
- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are: The outputs were needed to be generated as a hard copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

2.3 PROCESS MODEL USED WITH JUSTIFICATION



SDLC is nothing but Software Development Life Cycle. It is a standard which is used by the software industry to develop good software.

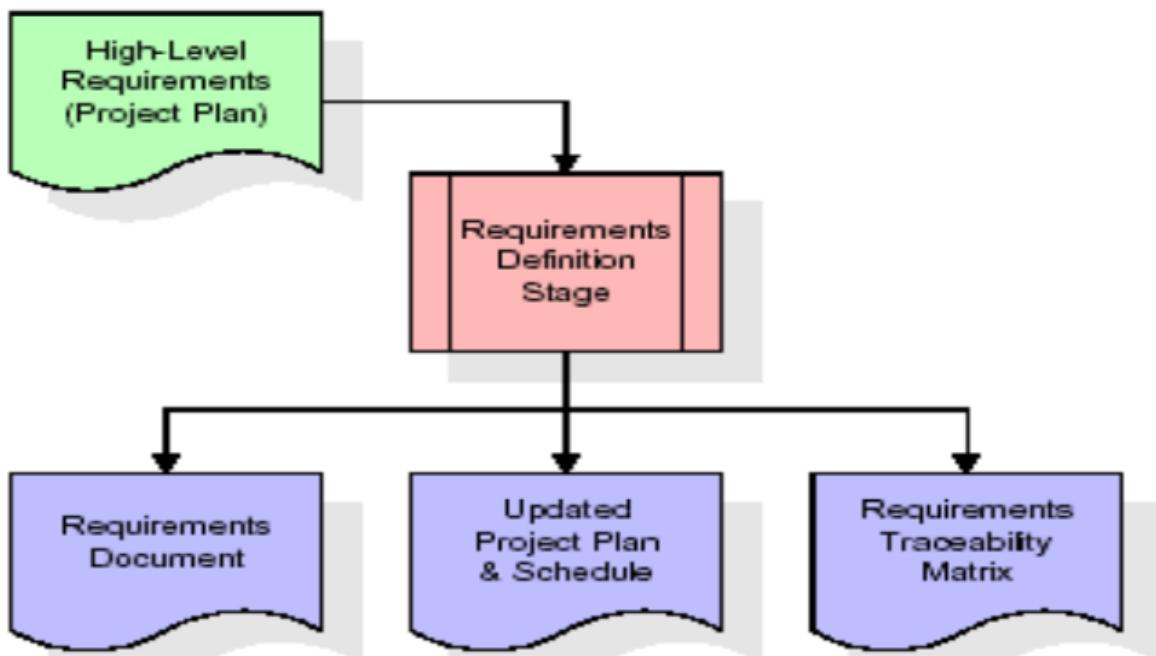
Stages in SDLC:

- Requirement Gathering
- Analysis
- Designing
- Coding
- Testing
- Maintenance Requirements

→ Gathering stage:

The requirements gathering process takes as its input the goals identified in the high level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define Umbrella Activity Umbrella Activity Umbrella Activity

- Feasibility Study
- TEAM FORMATION
- Project Specification PREPARATION Business Requirement Documentation ANALYSIS & DESIGN CODE UNIT TEST DOCUMENT CONTROL ASSESSMENT TRAINING INTEGRATION & SYSTEM TESTING DELIVERY/INSTALLATION ACCEPTANCE TEST Requirements Gathering operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

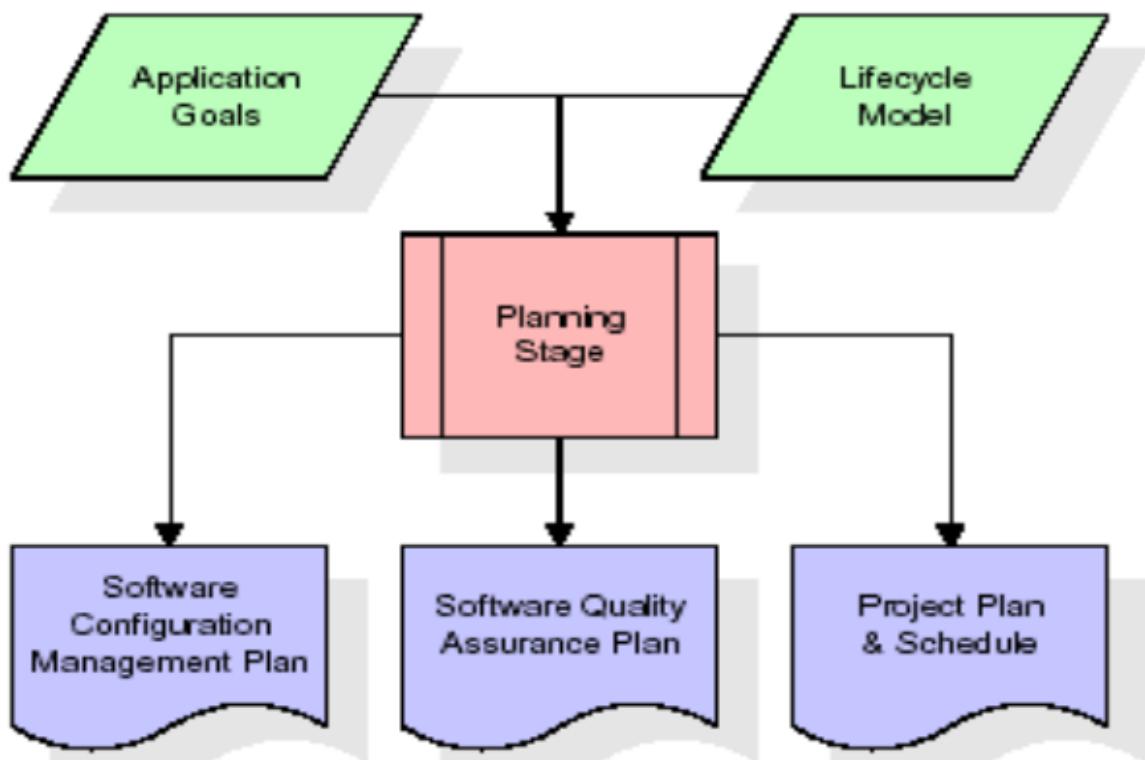


These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are not included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages. In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability. The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan. Feasibility study is all about identification of problems in a project. No. of staff required to handle a project is represented as Team Formation, in this case only modules with individual tasks will be assigned to employees who are working for that project. Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator

Analysis Stage:

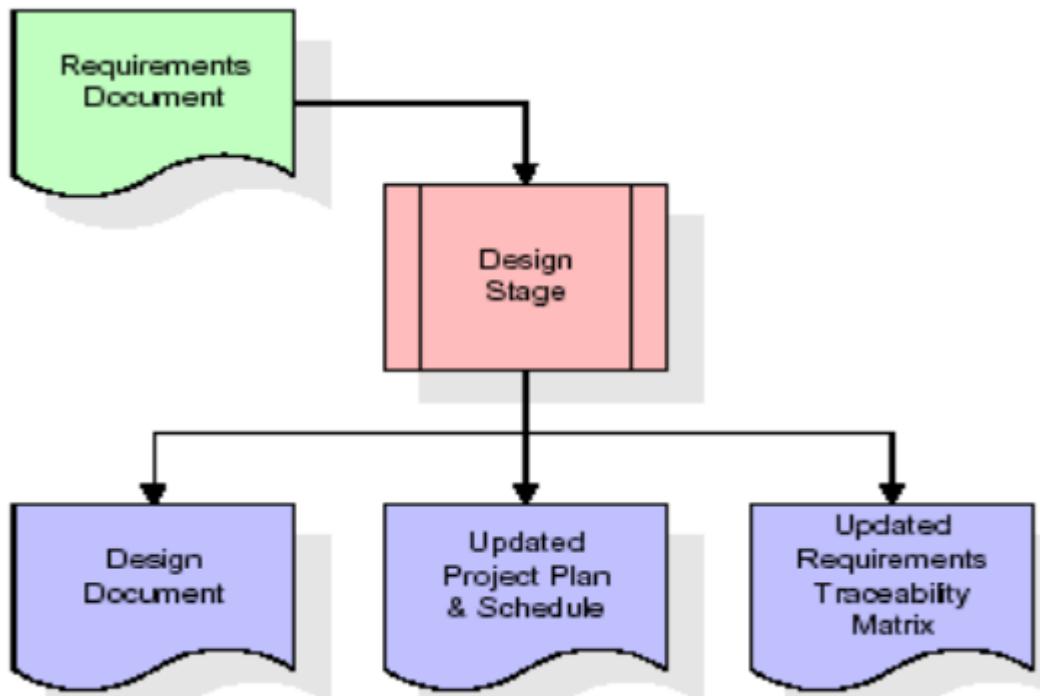
The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

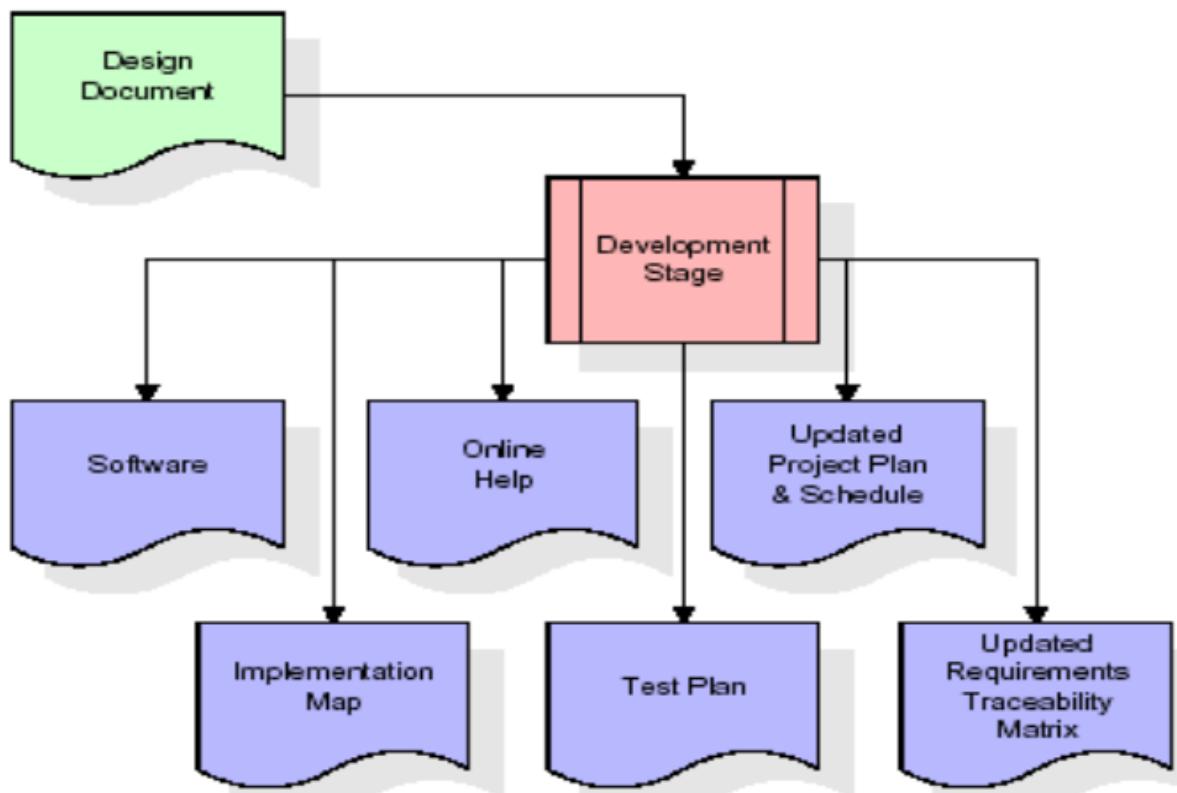
6.4 Designing Stage:

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalised and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan. Development (Coding) Stage: The

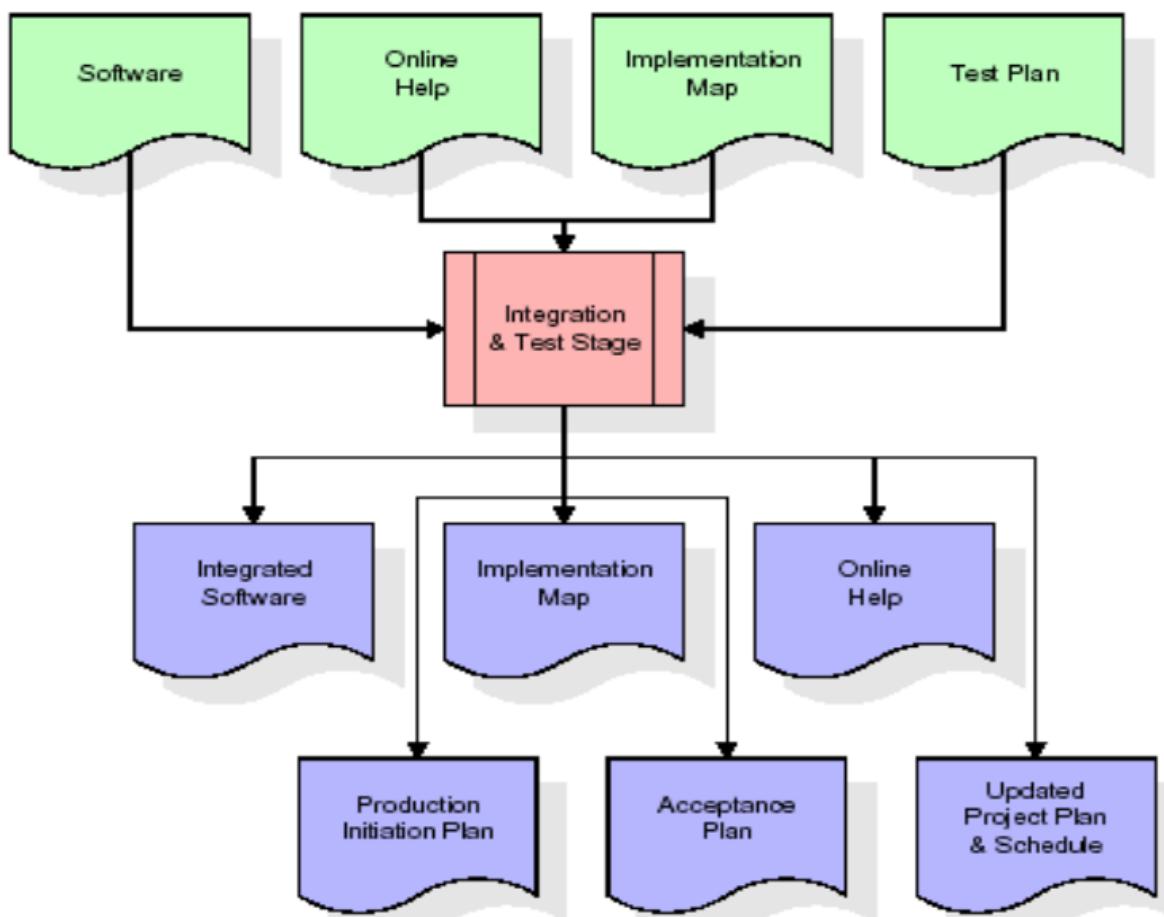
development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artefacts will be produced. Software artefacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialised procedures and functions. Appropriate test cases will be developed for each set of functionally related software artefacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artefact is linked to a specific design element, and that each developed artefact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

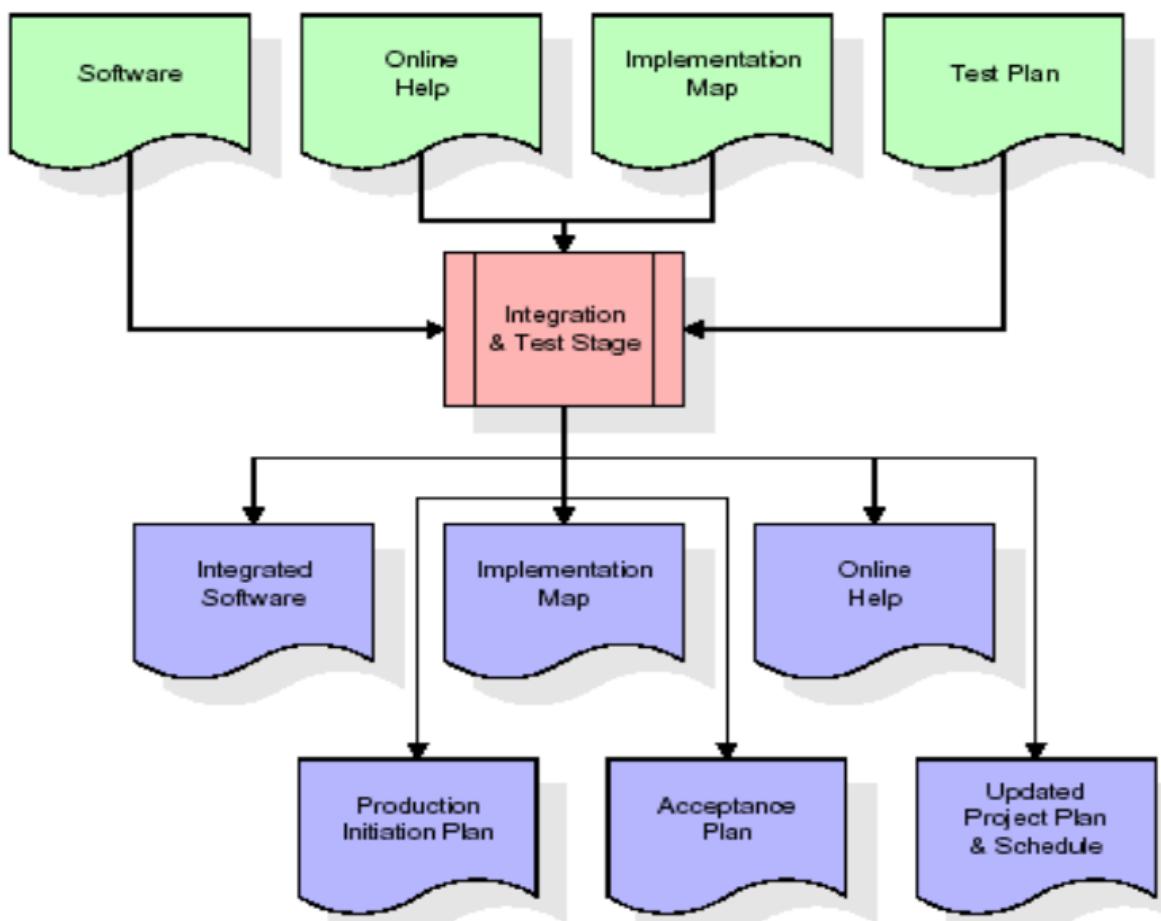
6.5 Integration & Test Stage:

During the integration and test stage, the software artefacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalised for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated Avanthy's Scientific Technological and Research Academy

project plan. Installation & Acceptance Test: During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer. After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



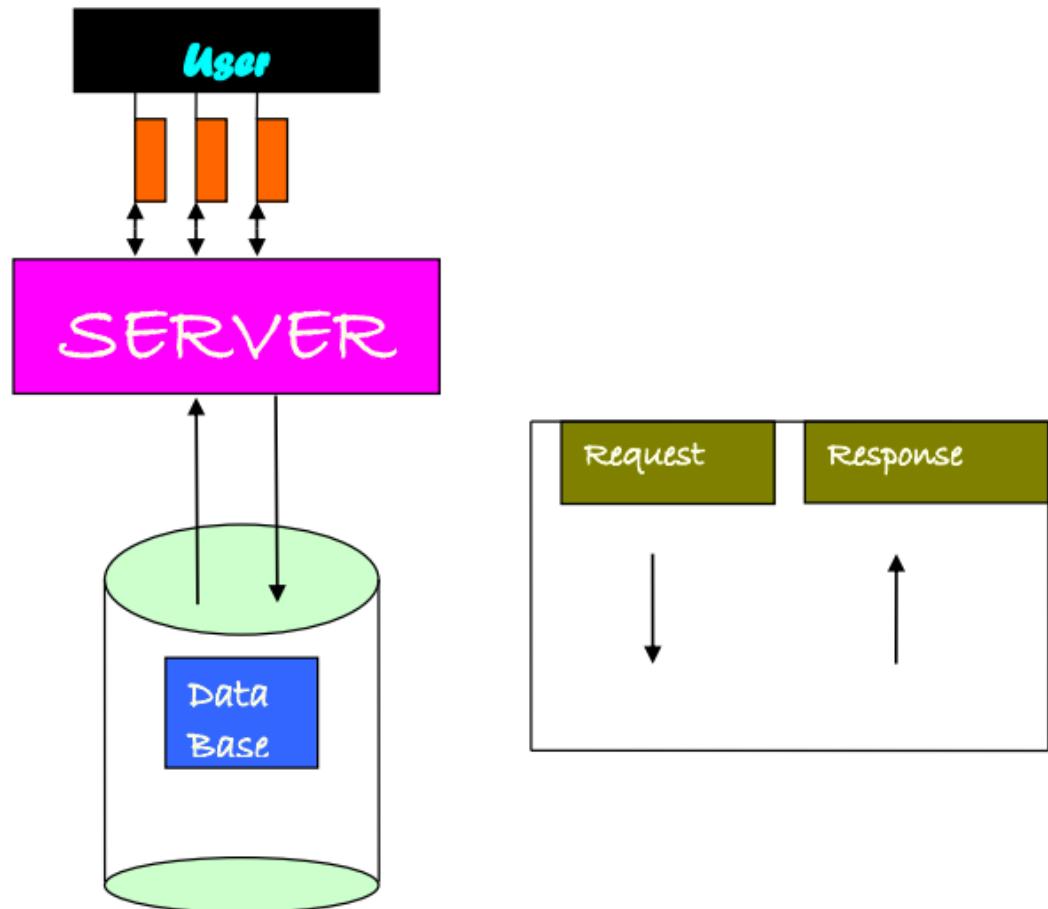
The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

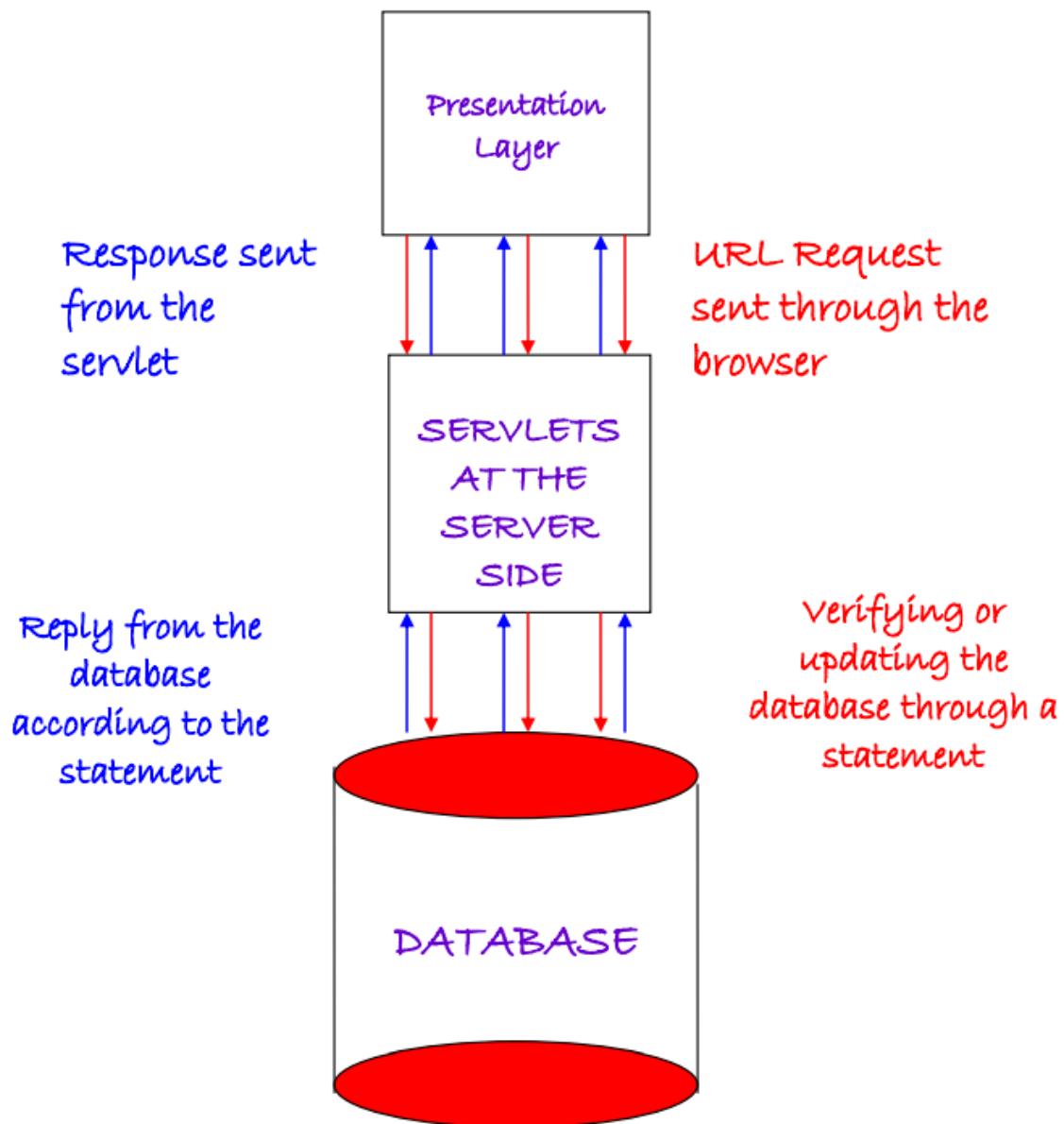
Maintenance:

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation, and later employees will be assigned work and they will undergo training in that particular assigned category. For this life cycle there is no end, it will be continued like an umbrella (no ending point to umbrella sticks).

6.6 SYSTEM ARCHITECTURE Architecture flow:

Below architecture diagram represents mainly the flow of requests from users to the database through servers. In this scenario the overall system is designed in three tires separately using three layers called presentation layer, business logic layer and data link layer. This project was developed using 3-tier architecture.



URL Pattern:

URL pattern represents how the requests are flowing through one layer to another layer and how the responses are getting by other layers to presentation layer through server in architecture diagram.

7. IMPLEMENTATION

7.1 SOURCE CODE

```

<!--
*** GENERATED FROM project.xml - DO NOT EDIT ***
***      EDIT ../build.xml INSTEAD      ***
-->

For the purpose of easier reading the script
is divided into following sections:
- initialization
- compilation
- dist
- execution
- debugging
- javadoc
- test compilation
- test execution
- test debugging
- cleanup

-->
<project
          xmlns:webproject1="http://www.netbeans.org/ns/web-project/1"
          xmlns:webproject2="http://www.netbeans.org/ns/web-project/2"
          xmlns:webproject3="http://www.netbeans.org/ns/web-project/3"      basedir=.."      default="default"
          name="Stock_Market_Prediction-impl">
    <import file="ant-deploy.xml"/>
    <fail message="Please build using Ant 1.7.1 or higher.">
        <condition>
            <not>
                <antversion atleast="1.7.1"/>
            </not>
        </condition>
    </fail>
    <target depends="dist,javadoc" description="Build whole project." name="default"/>
<!--
    INITIALIZATION SECTION
-->
<target name="-pre-init">
    <!-- Empty placeholder for easier customization. -->
    <!-- You can override this target in the ../build.xml file. -->
</target>
<target depends="-pre-init" name="-init-private">
    <property file="nbproject/private/private.properties"/>
</target>
<target depends="-pre-init,-init-private" name="-init-user">
    <property file="${user.properties.file}"/>
    <!-- The two properties below are usually overridden -->
    <!-- by the active platform. Just a fallback. -->
    <property name="default.javac.source" value="1.4"/>

```

```

<property name="default.javac.target" value="1.4"/>
</target>
<target depends="-pre-init,-init-private,-init-user" name="-init-project">
  <property file="nbproject/project.properties"/>
</target>
<target depends="-pre-init,-init-private,-init-user,-init-project,-init-macrodef-property"
if="dist.ear.dir" name="-do-ear-init"/>
<target depends="-pre-init,-init-private,-init-user,-init-project,-init-macrodef-property"
name="-do-init">
  <condition property="have.tests">
    <or>
      <available file="${test.src.dir}"/>
    </or>
  </condition>
  <condition property="have.sources">
    <or>
      <available file="${src.dir}"/>
    </or>
  </condition>
  <condition property="netbeans.home+have.tests">
    <and>
      <isset property="netbeans.home"/>
      <isset property="have.tests"/>
    </and>
  </condition>
  <condition property="no.javadoc.preview">
    <iffalse value="${javadoc.preview}"/>
  </condition>
  <property name="javac.compilerargs" value="" />
  <condition property="no.deps">
    <and>
      <istrue value="${no.dependencies}"/>
    </and>
  </condition>
  <condition property="no.dist.ear.dir">
    <not>
      <isset property="dist.ear.dir"/>
    </not>
  </condition>
  <property name="build.web.excludes" value="${build.classes.excludes}"/>
  <condition property="do.compile.jspss">
    <istrue value="${compile.jspss}"/>
  </condition>
  <condition property="do.debug.server">
    <or>
      <not>
        <isset property="debug.server"/>
      </not>
      <istrue value="${debug.server}"/>
    <and>
      <not>
        <istrue value="${debug.server}"/>
      </not>
    </and>
  </condition>

```

```

        </not>
        <not>
            <itrue value="\$\{debug.client\}" />
        </not>
        </and>
    </or>
</condition>
<condition property="do.debug.client">
    <itrue value="\$\{debug.client\}" />
</condition>
<condition property="do.display.browser">
    <itrue value="\$\{display.browser\}" />
</condition>
<condition property="do.display.browser.debug.old">
    <and>
        <isset property="do.display.browser"/>
        <not>
            <isset property="do.debug.client"/>
        </not>
        <not>
            <isset property="browser.context"/>
        </not>
    </and>
</condition>
<condition property="do.display.browser.debug">
    <and>
        <isset property="do.display.browser"/>
        <not>
            <isset property="do.debug.client"/>
        </not>
        <isset property="browser.context"/>
    </and>
</condition>
<available file="\$\{conf.dir\}/MANIFEST.MF" property="has.custom.manifest"/>
<available file="\$\{persistence.xml.dir\}/persistence.xml" property="has.persistence.xml"/>
<condition property="do.war.package.with.custom.manifest">
    <isset property="has.custom.manifest"/>
</condition>
<condition property="do.war.package.without.custom.manifest">
    <not>
        <isset property="has.custom.manifest"/>
    </not>
</condition>
<condition property="do.tmp.war.package.with.custom.manifest">
    <and>
        <isset property="has.custom.manifest"/>
        <or>
            <isfalse value="\$\{directory.deployment.supported\}" />
            <isset property="dist.ear.dir"/>
        </or>
    </and>
</condition>

```

```

<condition property="do.tmp.war.package.without.custom.manifest">
  <and>
    <not>
      <isset property="has.custom.manifest"/>
    </not>
    <or>
      <iffalse value="\${directory.deployment.supported}"/>
      <isset property="dist.ear.dir"/>
    </or>
  </and>
</condition>
<condition property="do.tmp.war.package">
  <or>
    <iffalse value="\${directory.deployment.supported}"/>
    <isset property="dist.ear.dir"/>
  </or>
</condition>
<property name="build.meta.inf.dir" value="\${build.web.dir}/META-INF"/>
<condition else="" property="application.args.param" value="\${application.args}">
  <and>
    <isset property="application.args"/>
    <not>
      <equals arg1="\${application.args}" arg2="" trim="true"/>
    </not>
  </and>
</condition>
<property name="source.encoding" value="\${file.encoding}"/>
<condition property="javadoc.encoding.used" value="\${javadoc.encoding}">
  <and>
    <isset property="javadoc.encoding"/>
    <not>
      <equals arg1="\${javadoc.encoding}" arg2="" />
    </not>
  </and>
</condition>
<property name="javadoc.encoding.used" value="\${source.encoding}"/>
<property name="includes" value="**"/>
<property name="excludes" value="" />
<property name="runmain.jvmargs" value="" />
<path id="endorsed.classpath.path" path="\${endorsed.classpath}"/>
<condition else="" property="endorsed.classpath.cmd.line.arg"
value="-Xbootclasspath/p:'\${toString:endorsed.classpath.path}'">
  <and>
    <isset property="endorsed.classpath"/>
    <length length="0" string="\${endorsed.classpath}" when="greater"/>
  </and>
</condition>
<condition else="false" property="jdkBug6558476">
  <and>
    <matches pattern="1\.[56]" string="\${java.specification.version}"/>
    <not>
      <os family="unix"/>
    </not>
  </and>
</condition>

```

```

        </not>
    </and>
</condition>
<property name="javac.fork" value="${jdkBug6558476}"/>
<condition property="junit.available">
    <or>
        <available classname="org.junit.Test" classpath="${run.test.classpath}"/>
        <available classname="junit.framework.Test" classpath="${run.test.classpath}"/>
    </or>
</condition>
<condition property="testng.available">
    <available classname="org.testng.annotations.Test" classpath="${run.test.classpath}"/>
</condition>
<condition property="junit+testng.available">
    <and>
        <istrue value="${junit.available}"/>
        <istrue value="${testng.available}"/>
    </and>
</condition>
<condition else="testng" property="testng.mode" value="mixed">
    <istrue value="${junit+testng.available}"/>
</condition>
<condition else="" property="testng.debug.mode" value="-mixed">
    <istrue value="${junit+testng.available}"/>
</condition>
</target>
<target depends="init" name="-init-cos" unless="deploy.on.save">
    <condition property="deploy.on.save" value="true">
        <or>
            <istrue value="${j2ee.deploy.on.save}"/>
            <istrue value="${j2ee.compile.on.save}"/>
        </or>
    </condition>
</target>
<target name="-post-init">
    <!-- Empty placeholder for easier customization. -->
    <!-- You can override this target in the ..build.xml file. -->
</target>
<target depends="-pre-init,-init-private,-init-user,-init-project,-do-init" name="-init-check">
    <fail unless="src.dir">Must set src.dir</fail>
    <fail unless="test.src.dir">Must set test.src.dir</fail>
    <fail unless="build.dir">Must set build.dir</fail>
    <fail unless="build.web.dir">Must set build.web.dir</fail>
    <fail unless="build.generated.dir">Must set build.generated.dir</fail>
    <fail unless="dist.dir">Must set dist.dir</fail>
    <fail unless="build.classes.dir">Must set build.classes.dir</fail>
    <fail unless="dist.javadoc.dir">Must set dist.javadoc.dir</fail>
    <fail unless="build.test.classes.dir">Must set build.test.classes.dir</fail>
    <fail unless="build.test.results.dir">Must set build.test.results.dir</fail>
    <fail unless="build.classes.excludes">Must set build.classes.excludes</fail>
    <fail unless="dist.war">Must set dist.war</fail>
    <condition property="missing.j2ee.server.home">

```

```

<and>
  <matches pattern="j2ee.server.home" string="${j2ee.platform.classpath}"/>
  <not>
    <isset property="j2ee.server.home"/>
  </not>
</and>
</condition>
<fail if="missing.j2ee.server.home"> The Java EE server classpath is not correctly set up - server home directory is missing. Either open the project in the IDE and assign the server or setup the server classpath manually. For example like this: ant -Dj2ee.server.home=<app_server_installation_directory></fail>
<fail unless="j2ee.platform.classpath"> The Java EE server classpath is not correctly set up. Your active server type is ${j2ee.server.type}. Either open the project in the IDE and assign the server or setup the server classpath manually. For example like this: ant -Duser.properties.file=<path_to_property_file> (where you put the property "j2ee.platform.classpath" in a .properties file) or ant -Dj2ee.platform.classpath=<server_classpath> (where no properties file is used) </fail>
</target>
<target name="-init-macrodef-property">
  <macrodef name="property" uri="http://www.netbeans.org/ns/web-project/1">
    <attribute name="name"/>
    <attribute name="value"/>
    <sequential>
      <property name="@{name}" value="${@{value}}"/>
    </sequential>
  </macrodef>
</target>
<target depends="-init-ap-cmdline-properties" if="ap.supported.internal"
name="-init-macrodef-javac-with-processors">
  <macrodef name="javac" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${src.dir}" name="sredir"/>
    <attribute default="${build.classes.dir}" name="destdir"/>
    <attribute default="${javac.classpath}:${j2ee.platform.classpath}" name="classpath"/>
    <attribute default="${javac.processorpath}" name="processorpath"/>
    <attribute default="${build.generated.sources.dir}/ap-source-output"
name="apgeneratedsredir"/>
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="${javac.debug}" name="debug"/>
    <attribute default="${empty.dir}" name="gensrcdir"/>
    <element name="customize" optional="true"/>
    <sequential>
      <property location="${build.dir}/empty" name="empty.dir"/>
      <mkdir dir="${empty.dir}"/>
      <mkdir dir="@{apgeneratedsredir}"/>
      <javac debug="@{debug}" deprecation="${javac.deprecation}" destdir="@{destdir}"
encoding="${source.encoding}" excludes="@{excludes}" fork="${javac.fork}"
includeantruntime="false" includes="@{includes}" source="${javac.source}"
srcdir="@{sredir}" target="${javac.target}"/>
      <src>
        <dirset dir="@{gensrcdir}" erroronmissingdir="false">
          <include name="*"/>

```

```

        </dirset>
    </src>
    <classpath>
        <path path="@{classpath}"/>
    </classpath>
    <compilerarg line="${endorsed.classpath.cmd.line.arg}"/>
    <compilerarg line="${javac.compilerargs}"/>
    <compilerarg value="-processorpath"/>
    <compilerarg path="@{processorpath}:${empty.dir}"/>
    <compilerarg line="${ap.processors.internal}"/>
    <compilerarg value="-s"/>
    <compilerarg path="@{apgeneratedsrcdir}"/>
    <compilerarg line="${ap.proc.none.internal}"/>
    <customize/>
</javac>
</sequential>
</macrodef>
</target>
<target depends="-init-ap-cmdline-properties" name="-init-macrodef-javac-without-processors"
unless="ap.supported.internal">
    <macrodef name="javac" uri="http://www.netbeans.org/ns/web-project/2">
        <attribute default="${src.dir}" name="srmdir"/>
        <attribute default="${build.classes.dir}" name="destdir"/>
        <attribute default="${javac.classpath}:${j2ee.platform.classpath}" name="classpath"/>
        <attribute default="${javac.processorpath}" name="processorpath"/>
        <attribute default="${build.generated.sources.dir}/ap-source-output"
name="apgeneratedsrcdir"/>
        <attribute default="${includes}" name="includes"/>
        <attribute default="${excludes}" name="excludes"/>
        <attribute default="${javac.debug}" name="debug"/>
        <attribute default="${empty.dir}" name="gensrcdir"/>
        <element name="customize" optional="true"/>
        <sequential>
            <property location="${build.dir}/empty" name="empty.dir"/>
            <mkdir dir="${empty.dir}"/>
            <javac debug="@{debug}" deprecation="${javac.deprecation}" destdir="@{destdir}"
encoding="${source.encoding}" excludes="@{excludes}" includeantruntime="false"
includes="@{includes}" source="${javac.source}" srmdir="@{srmdir}"
target="${javac.target}"/>
            <src>
                <dirset dir="@{gensrcdir}" erroronmissingdir="false">
                    <include name="*"/>
                </dirset>
            </src>
            <classpath>
                <path path="@{classpath}"/>
            </classpath>
            <compilerarg line="${endorsed.classpath.cmd.line.arg}"/>
            <compilerarg line="${javac.compilerargs}"/>
            <customize/>
        </javac>
    </sequential>

```

```

</macrodef>
</target>
<target      depends="-init-macrodef-javac-with-processors,-init-macrodef-javac-without-processors"
name="-init-macrodef-javac">
<macrodef name="depend" uri="http://www.netbeans.org/ns/web-project/2">
<attribute default="${src.dir}" name="scredir"/>
<attribute default="${build.classes.dir}" name="destdir"/>
<attribute default="${javac.classpath}:${j2ee.platform.classpath}" name="classpath"/>
<sequential>
<depend    cache="${build.dir}/depccache"    destdir="@{destdir}"    excludes="${excludes}"
includes="${includes}" scredir="@{scredir}">
<classpath>
<path path="@{classpath}"/>
</classpath>
</depend>
</sequential>
</macrodef>
<macrodef name="force-recompile" uri="http://www.netbeans.org/ns/web-project/2">
<attribute default="${build.classes.dir}" name="destdir"/>
<sequential>
<fail unless="javac.includes">Must set javac.includes</fail>
<pathconvert pathsep="${line.separator}" property="javac.includes.binary">
<path>
<filelist dir="@{destdir}" files="${javac.includes}"/>
</path>
<globmapper from="*.java" to="*.class"/>
</pathconvert>
<tempfile deleteonexit="true" property="javac.includesfile.binary"/>
<echo file="${javac.includesfile.binary}" message="${javac.includes.binary}"/>
<delete>
<files includesfile="${javac.includesfile.binary}"/>
</delete>
<delete file="${javac.includesfile.binary}"/>
</sequential>
</macrodef>
</target>
<target if="${junit.available}" name="-init-macrodef-junit-init">
<condition else="false" property="nb.junit.batch" value="true">
<and>
<itrue value="${junit.available}"/>
<not>
<isset property="test.method"/>
</not>
</and>
</condition>
<condition else="false" property="nb.junit.single" value="true">
<and>
<itrue value="${junit.available}"/>
<isset property="test.method"/>
</and>
</condition>
</target>

```

```

<target name="-init-test-properties">
  <property name="test.binaryincludes" value="<nothing>"/>
  <property name="test.binarytestincludes" value="" />
  <property name="test.binaryexcludes" value="" />
</target>
<target if="${nb.junit.single}" name="-init-macrodef-junit-single" unless="${nb.junit.batch}">
  <macrodef name="junit" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="**" name="testincludes"/>
    <attribute default="" name="testmethods"/>
    <element name="customize" optional="true"/>
    <sequential>
      <junit      dir="${basedir}"      errorproperty="tests.failed"      failureproperty="tests.failed"
        fork="true" showoutput="true" tempdir="${java.io.tmpdir}">
        <test          methods="@{testmethods}"                  name="@{testincludes}"
          todir="${build.test.results.dir}" />
        <syspropertyset>
          <propertyref prefix="test-sys-prop." />
          <mapper from="test-sys-prop.*" to="*" type="glob"/>
        </syspropertyset>
        <formatter type="brief" usefile="false"/>
        <formatter type="xml"/>
        <jvmarg value="-ea"/>
        <customize/>
      </junit>
    </sequential>
  </macrodef>
</target>
<target depends="-init-test-properties" if="${nb.junit.batch}" name="-init-macrodef-junit-batch"
unless="${nb.junit.single}">
  <macrodef name="junit" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="**" name="testincludes"/>
    <attribute default="" name="testmethods"/>
    <element name="customize" optional="true"/>
    <sequential>
      <property name="run.jvmargs.ide" value="" />
      <junit      dir="${basedir}"      errorproperty="tests.failed"      failureproperty="tests.failed"
        fork="true" showoutput="true" tempdir="${build.dir}">
        <batchtest todir="${build.test.results.dir}">
          <fileset          dir="${test.src.dir}"                  excludes="@{excludes},${excludes}"
            includes="@{includes}">
            <filename name="@{testincludes}" />
          </fileset>
          <fileset          dir="${build.test.classes.dir}">
            excludes="@{excludes},${excludes},${test.binaryexcludes}"
            includes="${test.binaryincludes}">
            <filename name="${test.binarytestincludes}" />
          </fileset>
        </batchtest>
      </junit>
    </sequential>
  </macrodef>
</target>

```

```

<syspropertyset>
    <propertyref prefix="test-sys-prop."/>
    <mapper from="test-sys-prop.*" to="*" type="glob"/>
</syspropertyset>
<formatter type="brief" usefile="false"/>
<formatter type="xml"/>
<jvmarg value="-ea"/>
<jvmarg line="${run.jvmargs.ide}"/>
<customize/>
</junit>
</sequential>
</macrodef>
</target>
<target depends="-init-macrodef-junit-init,-init-macrodef-junit-single, -init-macrodef-junit-batch"
if="${junit.available}" name="-init-macrodef-junit"/>
<target if="${testng.available}" name="-init-macrodef-testng">
    <macrodef name="testng" uri="http://www.netbeans.org/ns/web-project/2">
        <attribute default="${includes}" name="includes"/>
        <attribute default="${excludes}" name="excludes"/>
        <attribute default="**" name="testincludes"/>
        <attribute default="" name="testmethods"/>
        <element name="customize" optional="true"/>
        <sequential>
            <condition else="" property="testng.methods.arg" value="@{testincludes}.@{testmethods}">
                <isset property="test.method"/>
            </condition>
            <union id="test.set">
                <fileset      dir="${test.src.dir}"      excludes="@{excludes},**/*.xml,${excludes}">
                    <includes>@{includes}>
                        <filename name="@{testincludes}"/>
                </fileset>
            </union>
            <taskdef      classname="org.testng.TestNGAntTask"      classpath="${run.test.classpath}"
name="testng"/>
                <testng      classfilesetref="test.set"      failureProperty="tests.failed"
listeners="org.testng.reporters.VerboseReporter"      methods="${testng.methods.arg}"
mode="${testng.mode}"      outputdir="${build.test.results.dir}"
suitename="Stock_Market_Prediction" testname="TestNG tests" workingDir="${basedir}">
                    <xmlfileset dir="${build.test.classes.dir}" includes="@{testincludes}"/>
                <propertyset>
                    <propertyref prefix="test-sys-prop."/>
                    <mapper from="test-sys-prop.*" to="*" type="glob"/>
                </propertyset>
                <customize/>
            </testng>
        </sequential>
    </macrodef>
</target>
<target name="-init-macrodef-test-impl">
    <macrodef name="test-impl" uri="http://www.netbeans.org/ns/web-project/2">
        <attribute default="${includes}" name="includes"/>
        <attribute default="${excludes}" name="excludes"/>

```

```

<attribute default="**" name="testincludes"/>
<attribute default="" name="testmethods"/>
<element implicit="true" name="customize" optional="true"/>
<sequential>
  <echo>No tests executed.</echo>
</sequential>
</macrodef>
</target>
<target depends="-init-macrodef-junit" if="${junit.available}" name="-init-macrodef-junit-impl">
  <macrodef name="test-impl" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="**" name="testincludes"/>
    <attribute default="" name="testmethods"/>
    <element implicit="true" name="customize" optional="true"/>
    <sequential>
      <webproject2:junit           excludes="@{excludes}"           includes="@{includes}">
        testincludes="@{testincludes}" testmethods="@{testmethods}">
        <customize/>
      </webproject2:junit>
    </sequential>
  </macrodef>
</target>
<target depends="-init-macrodef-testng" if="${testng.available}" name="-init-macrodef-testng-impl">
  <macrodef name="test-impl" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="**" name="testincludes"/>
    <attribute default="" name="testmethods"/>
    <element implicit="true" name="customize" optional="true"/>
    <sequential>
      <webproject2:testng           excludes="@{excludes}"           includes="@{includes}">
        testincludes="@{testincludes}" testmethods="@{testmethods}">
        <customize/>
      </webproject2:testng>
    </sequential>
  </macrodef>
</target>
<target depends="-init-macrodef-test-impl,-init-macrodef-junit-impl,-init-macrodef-testng-impl" name="-init-macrodef-test">
  <macrodef name="test" uri="http://www.netbeans.org/ns/web-project/2">
    <attribute default="${includes}" name="includes"/>
    <attribute default="${excludes}" name="excludes"/>
    <attribute default="**" name="testincludes"/>
    <attribute default="" name="testmethods"/>
    <sequential>
      <webproject2:test-impl           excludes="@{excludes}"           includes="@{includes}">
        testincludes="@{testincludes}" testmethods="@{testmethods}">
        <customize>
          <classpath>

```

```

<target depends="initdeps-jar,-pre-pre-compile,-pre-compile,-do-compile,-post-compile"
description="Compile project." name="compile"/>
<target name="-pre-compile-single">
  <!-- Empty placeholder for easier customization. -->
  <!-- You can override this target in the ../build.xml file. -->
</target>
<target depends="initdeps-jar,-pre-pre-compile" name="-do-compile-single">
  <fail unless="javac.includes">Must select some files in the IDE or set javac.includes</fail>
  <webproject2:javac excludes="" gensrcdir="${build.generated.sources.dir}"
includes="${javac.includes}">
  <copy todir="${build.classes.dir}">
    <fileset dir="${src.dir}" excludes="${build.classes.excludes},${excludes}"
includes="${includes}">
  </copy>
</target>
<target name="-post-compile-single">
  <!-- Empty placeholder for easier customization. -->
  <!-- You can override this target in the ../build.xml file. -->
</target>

<target
depends="init,compile-test-single,-pre-test-run-single,-do-test-run-single-method,-post-test-run-single-
method" description="Run single unit test." name="test-single-method"/>
<!--

```

TEST DEBUGGING SECTION

-->

```

<target depends="init,compile-test-single,-pre-test-run-single" if="have.tests"
name="-debug-start-debuggee-test">
  <fail unless="test.class">Must select one file in the IDE or set test.class</fail>
  <webproject2:test-debug excludes="" includes="${javac.includes}" testClass="${test.class}"
testincludes="${javac.includes}">
</target>
<target depends="init,compile-test-single,-pre-test-run-single" if="have.tests"
name="-debug-start-debuggee-test-method">
  <fail unless="test.class">Must select one file in the IDE or set test.class</fail>
  <fail unless="test.method">Must select some method in the IDE or set test.method</fail>
  <webproject2:test-debug excludes="" includes="${javac.includes}" testClass="${test.class}"
testMethod="${test.method}" testincludes="${test.class}" testmethods="${test.method}">
</target>
<target depends="init,compile-test" if="netbeans.home+have.tests"
name="-debug-start-debugger-test">
  <webproject1:nbjpdastart classpath="${debug.test.classpath}" name="${test.class}">
</target>
<target depends="init,compile-test,-debug-start-debugger-test,-debug-start-debuggee-test"
name="debug-test"/>
<target
depends="init,compile-test-single,-debug-start-debugger-test,-debug-start-debuggee-test-method"
name="debug-test-method">
<target depends="init,-pre-debug-fix,compile-test-single" if="netbeans.home"
name="-do-debug-fix-test">
  <webproject1:nbjpdareload dir="${build.test.classes.dir}">

```

```

</target>
<target depends="init,-pre-debug-fix,-do-debug-fix-test" if="netbeans.home" name="debug-fix-test"/>
<!--

    CLEANUP SECTION
    -->

<target depends="init" name="deps-clean" unless="no.deps"/>
<target depends="init" name="do-clean">
    <condition property="build.dir.to.clean" value="${build.web.dir}">
        <isset property="dist.ear.dir"/>
    </condition>
    <property name="build.dir.to.clean" value="${build.web.dir}">
    <delete includeEmptyDirs="true" quiet="true">
        <fileset dir="${build.dir.to.clean}/WEB-INF/lib"/>
    </delete>
    <delete dir="${build.dir}">
        <available file="${build.dir.to.clean}/WEB-INF/lib" property="status.clean-failed" type="dir"/>
        <delete dir="${dist.dir}">
    </target>
    <target depends="do-clean" if="status.clean-failed" name="check-clean">
        <echo message="Warning: unable to delete some files in ${build.web.dir}/WEB-INF/lib - they are
probably locked by the J2EE server. "/>
        <echo level="info" message="To delete all files undeploy the module from Server Registry in
Runtime tab and then use Clean again."/>
    </target>
    <target depends="init" if="netbeans.home" name="undeploy-clean">
        <nbundeploy failOnError="false" startServer="false"/>
    </target>
    <target name="-post-clean">
        <!-- Empty placeholder for easier customization. -->
        <!-- You can override this target in the ./build.xml file. -->
    </target>
    <target depends="init,undeploy-clean,deps-clean,do-clean,check-clean,-post-clean"
description="Clean build products." name="clean">
        <target depends="clean" description="Clean build products." name="clean-ear"/>
</project>

```

7.2 DATABASE

```

/*
SQLyog Enterprise - MySQL GUI v6.56
MySQL - 5.0.67-community-nt : Database - stock_prediction
*****
*/
/*!40101 SET NAMES utf8 */;

/*!40101 SET SQL_MODE='';
```

```

/*!40014      SET      @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101          SET          @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO'`;

CREATE DATABASE /*!32312 IF NOT EXISTS*/`stock_predection` /*!40100 DEFAULT
CHARACTER SET latin1 */;

USE `stock_predection`;

/*Table structure for table `articlereg` */

DROP TABLE IF EXISTS `articlereg`;

CREATE TABLE `articlereg` (
  `articlename` varchar(1000) default NULL,
  `password` varchar(1000) default NULL,
  `articleid` varchar(1000) default NULL,
  `articletype` varchar(1000) default NULL,
  `address` varchar(1000) default NULL,
  `employees` varchar(1000) default NULL,
  `branches` varchar(1000) default NULL,
  `status` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*Data for the table `articlereg` */

insert into
`articlereg`(`articlename`,`password`,`articleid`,`articletype`,`address`,`employees`,`branches`,`status`)
values
('timesofindia','123','1591804690854330','public','hyderabad','100','10','waiting'),('the
hindu','123','318075888696996','public','asdfghjkl','100','10','waiting'),('decconschronicle','123',
'1591804690854330','public','hyderabad','100','10','waiting');

/*Table structure for table `comment` */

```

```
DROP TABLE IF EXISTS `comment`;

CREATE TABLE `comment` (
    `articlename` varchar(1000) default NULL,
    `companynname` varchar(1000) default NULL,
    `companyid` varchar(1000) default NULL,
    `opinion` varchar(1000) default NULL,
    `comment` varchar(1000) default NULL,
    `date` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
/*Data for the table `comment` */
```

```
insert into `comment`(`articlename`,`companynname`,`companyid`,`opinion`,`comment`,`date`)
values
        ('timesofindia','techmahindra','1','negative','bad','2018-03-26
14:36:18'),('timesofindia','techmahindra','1','negative','very           bad','2018-03-26
14:36:29'),('timesofindia','techmahindra','1','neutral','may      be     gud','2018-03-26
14:36:42'),('timesofindia','techmahindra','1','neutral','maybe   bad','2018-03-26
14:36:53'),('timesofindia','techmahindra','1','positive','nice','2018-03-26
14:39:59'),('timesofindia','techmahindra','1','positive','very     nice','2018-03-26
14:40:11'),('timesofindia','techmahindra','1','negative','bad','2018-03-26 14:42:34');
```

```
/*Table structure for table `comregister` */
```

```
DROP TABLE IF EXISTS `comregister`;
```

```
CREATE TABLE `comregister` (
    `companynname` varchar(1000) NOT NULL,
    `companyid` varchar(1000) NOT NULL,
    `product` varchar(1000) default NULL,
    `companytype` varchar(1000) NOT NULL,
    `companyaddress` varchar(1000) NOT NULL,
    `employees` varchar(1000) NOT NULL,
    `branches` varchar(1000) NOT NULL,
    `turnover` double NOT NULL,
    `username` varchar(1000) NOT NULL,
    `password` varchar(1000) NOT NULL,
```

```

`status` varchar(1000) NOT NULL,
`shares` double NOT NULL,
`sharevalue` double NOT NULL,
`availableshares` double default NULL,
`id` int(11) NOT NULL auto_increment,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;

```

/*Data for the table `comregister` */

```

insert into
`comregister`(`companyname`, `companyid`, `product`, `companytype`, `companyaddress`, `employees`, `branches`, `turnover`, `username`, `password`, `status`, `shares`, `sharevalue`, `availableshares`, `id`)
values
('techmahindra','123','softwares','it','gachibowli','100','10',2000000,'venkat','venkat','activated',1000,2000,1000,1);

```

/*Table structure for table `publish` */

```
DROP TABLE IF EXISTS `publish`;
```

```

CREATE TABLE `publish` (
`publishedby` varchar(1000) default NULL,
`companyname` varchar(1000) default NULL,
`description` varchar(1000) default NULL,
`status` varchar(1000) default NULL,
`quality` varchar(1000) default NULL,
`positive` varchar(1000) default NULL,
`negative` varchar(1000) default NULL,
`neutral` varchar(1000) default NULL,
`id` int(11) NOT NULL auto_increment,
`date` varchar(1000) default NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;

```

/*Data for the table `publish` */

```

insert into
`publish`(`publishedby`,`companynname`,`description`,`status`,`quality`,`positive`,`negative`,`natural`,`id`,`date`) values ('timesofindia','techmahindra','this is the one of the software company in india','positive','80','3','4','3',1,'2018-03-26 13:47:53'),('the hindu','techmahindra','this is the software company which is the best in india','positive','90','0','0','0',2,'2018-03-26 15:24:55'),('deccanchronicle','techmahindra','best software company','positive','70','0','0','0',3,'2018-03-26 15:25:55');

```

/*Table structure for table `purchaseshare` */

DROP TABLE IF EXISTS `purchaseshare`;

```

CREATE TABLE `purchaseshare` (
`buyer` varchar(1000) default NULL,
`companynname` varchar(1000) default NULL,
`companyid` varchar(1000) default NULL,
`totalshares` double default NULL,
`purchasedshare` double default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

/*Data for the table `purchaseshare` */

/*Table structure for table `stock` */

DROP TABLE IF EXISTS `stock`;

```

CREATE TABLE `stock` (
`username` varchar(200) default NULL,
`password` varchar(200) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

/*Data for the table `stock` */

```
insert into `stock`(`username`,`password`) values ('Stock','Stock');
```

/*Table structure for table `userreg` */

```
DROP TABLE IF EXISTS `userreg`;
```

```
CREATE TABLE `userreg` (
    `fullname` varchar(1000) default NULL,
    `emailid` varchar(1000) default NULL,
    `aadhar` varchar(1000) default NULL,
    `panno` varchar(1000) default NULL,
    `mobile` varchar(1000) default NULL,
    `address` varchar(1000) default NULL,
    `username` varchar(1000) default NULL,
    `password` varchar(1000) default NULL,
    `status` varchar(1000) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
/*Data for the table `userreg` */
```

```
insert into
`userreg`(`fullname`,`emailid`,`aadhar`,`panno`,`mobile`,`address`,`username`,`password`,`status`)
values
('venkat','venkatjavaprojects@gmail.com','123456789','beypg1212B','1234567890','hyd','venkat','venkat','activated');
```

```
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
```

```
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
```

8. TESTING

8.1 INTRODUCTION TO TESTING

Introduction to Testing: Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

8.2 TESTING IN STRATEGIES

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

8.2.1 Unit Testing:

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements.

Each module can be tested using the following two Strategies:

8.2.2 Black Box Testing:

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been used to find errors in the following categories:

0.Incorrect or missing functions

1. Interface errors

2.Errors in data structure or external database access

3.Performance errors Initialization and termination errors.

In this testing only the output is checked for correctness

. The logical flow of the data is not checked.

8.2.3 White Box testing :

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases. It has been used to generate the test cases in the following cases:

- ✓ Guarantee that all independent paths have been Executed.
- ✓ Execute all logical decisions on their true and false Sides
- ✓ Execute all loops at their boundaries and within their operational bounds
- ✓ Execute internal data structures to ensure their validity.

8.3 Integrating Testing :

Integration testing ensures that software and subsystems work together a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together.

8.4 System Testing:

Involves in-house testing of the entire system before delivery to the user. Its aim is to satisfy the user the system meets all requirements of the client's specifications.

8.5 Acceptance Testing :

It is a pre-delivery testing in which entire system is tested at client's site on real world data to find errors.

8.6 Test Approach :

Testing can be done in two ways:

Bottom up approach

Top down approach

8.6.1 Bottom up Approach:

Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded with in the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

8.6.2 Top down approach:

This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

9. Validation:

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed

9.1 INTRODUCTION

System Security: Setting Up Authentication for Web Applications Introduction: To configure authentication for a Web Application, use the element of the web.xml deployment descriptor. In this element you define the security realm containing the user credentials, the method of authentication, and the location of resources for authentication.

9.2 SECURITY IN SOFTWARE

To set up authentication for Web Applications:

1. Open the web.xml deployment descriptor in a text editor or use the Administration Console. Specify the authentication method using the element. The available options are:

BASIC Basic authentication uses the Web Browser to display a username/password dialog box. This username and password is authenticated against the realm.

Form-based authentication requires that you return an HTML form containing the username and password. The fields returned from the form elements must be: j_username and j_password, and the action attribute must be j_security_check. Here is an example of the HTML coding for using FORM authentication:

The resource used to generate the HTML form may be an HTML page, a JSP, or a servlet. You define this resource with the element. The HTTP session object is created when the login page is served. Therefore, the session.isNew () method returns FALSE when called from pages served after successful authentication.

10. BIBLIOGRAPHY

References for the Project Development Were Taken From the following Books and Websites .

JAVA Technologies

JAVA Complete Reference

Java Script Programming by Yehuda Shiran

Mastering JAVA Security

JAVA2 Networking by Pistoria

JAVA Security by Scotl oaks

Head First EJB Sierra Bates

J2EE Professional by Shadab siddiqui

JAVA server pages by Larne Pekowsky

JAVA Server pages by Nick Todd

HTML

HTML Black Book by Holzner

JDBC Java Database Programming with

JDBC by Patel moss.

Software Engineering by Roger Pressman

11. CONCLUSION

In Data Mining to predict stock market here we have created NLP based module & statistical parameter based module which results the sentence polarity & behaviour compared to past year data.