Java Browser with Integrated Chat Server

By

Kunal Jain 071309

Manav Bhasin 071344

Under the Supervision of

Dr. Ravi Rastogi

Submitted in partial fulfillment

Of the degree of

Bachelor of Technology

Department of CSE & / IT

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT

SOLAN, HIMACHAL PRADESH

INDIA

2011

CONTENTS

CERTIFICATE

This is to certify that the work entitled **Java Browser with Integrated Chat Server** submitted by Kunal Jain and Manav Bhasin in partial fulfillment for award of degree of Bachelor of Technology (Computer Science & Engineering) of Jaypee University of Information Technology has been carried out under my supervision.

This work has not been submitted partially or wholly to any other university or institution for award of this or any other degree programme.

Signature of Supervisor ..... Ravi Rastogi

Name of Supervisor ..... DR. RAVI RASTOGI

Department ..... CSE and IT

Date ..... 20th May, 2011

ACKNOWLEDGMENT

We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout the duration of our project.

This report acknowledges the intense driving and technical competence of all the individuals who have contributed to it. It would have been almost impossible to produce fruitful results during the working of the project without the support of those people. We extend our thanks and gratitude to our Project Guide Dr. Ravi Rastogi who has helped us at every step. He spent his valuable time from his busy schedule to train us and provided his active and sincere support for our daily activities.

I would like to express my heartfelt thanks to Brig. (Mr.) S.P.Ghrera, H.O.D., Computer Science Department, Jaypee University of Information Technology, for his astute guidance, his constant encouragement and support throughout.

This report has been compiled by the sincere and active support from our guide who provided us proper guidance and direction regarding various problems. We have tried our best to summarize this report.

SIGNATURE ---------- ----------

NAME Kunal Jain (071309) Manav Bhasin (071344)

DATE 20th May '2011 20th May '2011

SUMMARY

If current trends continue, it is likely that the web browser will become the only widely used user interface. Web applications will become the predominant software. Should this happen, user interface design, implementation and evaluation skills can become more focused and effective. Building on the existing technology, we have adopted a modest approach to recreate most of the features, provided in the existing browser applications in JAVA. We have built an application that exploits the tremendous potential of JFC (Java foundation classes) and AWT (abstract window toolkit).However, in order to gain an edge over the existing browses we have embedded a Chat application and a Download Manager, which again uses the above said technology. In this the Chat Facility is Intranet.

Albeit, we have successfully reached our goals, there remains a huge scope in this project. A browser based purely on JAVA is an innovative and rather unexplored concept, which should generate much interest in the near future.

Keywords: JAVA, AWT, JFC, Browser, Download Manager, Chat

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 World Wide Web (WWW)

The World Wide Web (or the "Web") is a system of interlinked, hypertext documents that runs over the Internet. With a Web Browser, a user views Web Pages that may contain text, images, and other multimedia and navigates between them using hyperlinks The Web was created around 1990 by the Englishman Tim Bemers-Lee and the Belgian Robert Cailliau working at CERN in Geneva, Switzerland. Since then, Berners-Lee has played an active role in guiding the development of Web standards (such as the markup languages in which Web pages are composed), and in recent years has advocated his vision of a Semantic Web.

The World Wide Web is the combination of four basic ideas:

1. Hypertext: a format of information which allows, in a computer environment, one to move from one part of a document to another or from one document to another through internal connections among these documents (called "hyperlinks");
2. Resource Identifiers: unique identifiers used to locate a particular resource (computer file, document or other resource) on the network -this is commonly known as a URL or URI, although the two have subtle technical differences;
3. The Client-server model of computing: a system in which client software or a client computer makes requests of server software or a server computer that provides the client with resources or services, such as data or files; and
4. Markup language: characters or codes embedded in text which indicates structure, semantic meaning, or advice on presentation.

On the World Wide Web, a client program called a user agent retrieves information resources, such as Web pages and other computer files, from Web servers using their URLs. If the user agent is a kind of Web browser, it displays the resources on a user's computer. The user can then follow hyperlinks in each web page to other World Wide Web resources, whose location is embedded in the hyperlinks. It is also possible, for example by filling in and

submitting web forms, to post information back to a Web server for it to save or process in some way. Web pages are often arranged in collections of related material called "Web sites." The act of following hyperlinks from one Web site to another is referred to as *"browsing"* or sometimes as "surfing "the Web.

How the Web works

Viewing a Web page or other resource on the World Wide Web normally begins either by typing the URL of the page into a Web browser, or by following a hypertext link to that page or resource. The first step, behind the scenes, is for the server-name part of the URL to be resolved into an IP address by the global, distributed Internet database known as the Domain name system or DNS. The browser then establishes a TCP connection with the server at that IP address.

The next step is for an HTTP request to be sent to the Web server, requesting the resource. In the case of a typical Web page, the HTML text is first requested and parsed by the browser, which then makes additional requests for graphic^' and any other files that form a part of the page in quick succession. When considering web site popularity statistics, these additional file requests give rise to the difference between one single 'page view' and an associated number of server 'hits'.

The Web browser then renders the page as described by the HTML, CSS and other file's received, incorporating the images and other resources as necessary. This produces the on-screen page that the viewer sees. Most Web pages will themselves contain hyperlinks to other related pages and perhaps to downloads, source documents, definitions and other Web resources.

Caching

If the user returns to a page fairly soon, it is likely that the data will not be retrieved from the source Web server, as above, again. By default, browsers cache all web resources on the local hard drive. An HTTP request will be sent by the browser that asks for the data *only if it has been updated since the last download.* If it has not, the cached version will be reused in the rendering step.

This is particularly valuable in reducing the amount of Web traffic on the Internet. The decision

about expiration is made independently for each resource (image, style sheet, Java Script file etc., as well as for the HTML itself). Thus even on sites with highly dynamic content, many of the basic resources are only supplied once per session or less. It is worth it for any Web site designer to collect all the CSS and JavaScript into a few site-wide files so that they can be downloaded into users' caches and reduce page download times and demands on the server.

There are other components of the Internet that can cache Web content. The most common in practice are often built into corporate and academic firewalls where they cache web resources requested by one user for the benefit of all. Some search engines such as Google or Yahoo! also store cached content from Web sites.

Apart from the facilities built into Web servers that can ascertain when physical files have been updated, it is possible for designers of dynamically generated web pages to control the HTTP headers sent back to requesting users, so that pages are not cached when they should not be — for example Internet banking and news pages.

This helps with understanding the difference between the HTTP 'GET' and 'POST' verbs - data requested with a GET may be cached, if other conditions are met, whereas data obtained after posting information to the server usually will not.

1.2 WHAT IS A WEB BROWSER?

A web browser is a software application that enables a user to display and interact with text, images, and other information typically located on a web page at a website on the World Wide Web or a local area network. Web browsers allow a user to quickly and easily access information provided on many web pages at many websites by traversing these links. They can also be used to access information provided by web servers in private networks or content in file systems. ,

Protocols and standards

Web browsers communicate with web servers primarily using HTTP (hypertext transfer protocol) to fetch web pages. HTTP allows web browsers to submit information to web servers as well as fetch web pages from them. The most commonly used HTTP is HTTP/I.I, which is fully defined in RFC 26J6. HTTP/I.I has its own required standards that Internet Explore? does

not fully support, but most other current-generation web browsers do.

Pages are located by means of a URL (uniform resource locator), which is treated as an address, beginning with *http:* for HTTP access. Many browsers also support a variety of other URL types and their corresponding protocols, such *as ftp:* for FTP (file transfer protocol), *rtsp:* for RTSP (real-time streaming protocol), and *https:* for HTTPS (an SSL encrypted version of HTTP).

The file format for a web page is usually HTML (hyper-text markup language) and is identified in the HTTP protocol using a MIME *content type.* Most browser natively support a variety of formats in addition to HTML, such as the JPEG, PNG and GIF image formats, and can be extended to supcript more through the use of plugins. The combination of HTTP *content type* and URL protocol specification allows web page designers to embed images, animations, video, sound, and streaming media into a web page, or to make them accessible through the web page.

1.3 CLIENT – SEVER MODEL

Client/server is a computational architecture that involves client processes requesting service from server processes. In any given exchange, the client initiates the request and the server responds accordingly. A server cannot initiate dialog with clients. Since the client and server are software entities they can be located on any appropriate hardware. A client process, for instance, could be resident on a network server hardware, and request data from a server process running on server hardware or even on a PC.

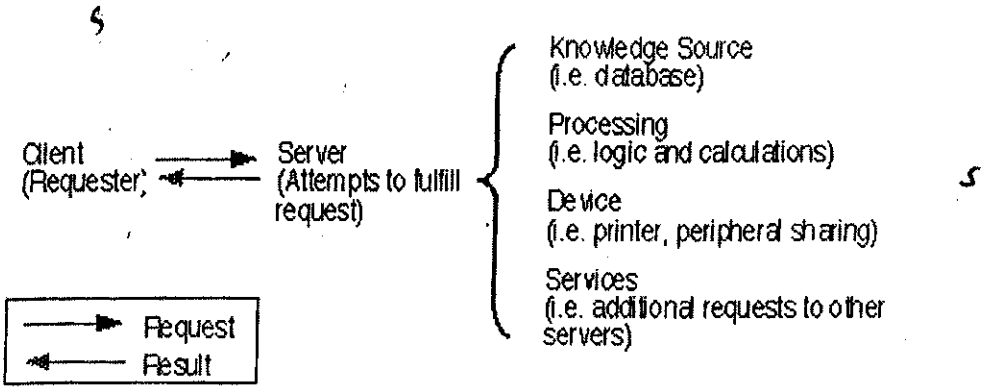


Fig 1.1

The steps involved in establishing a socket on the client side are as follows:

1. Create a socket with the socket ( ) system call
2. Connect the socket to the address of the server using the connect() system call
3. Send and receive data. There are a number of ways to do this, but the simplest is to use the read ( ) and write ( ) system calls.

The steps involved in establishing a socket on the server side are as follows:

1. Create a socket with the socket() system call
2. Bind the socket to an address using the bind( ) system call. For a server socket on the Internet, an address consists of a port number on the host machine.
3. Listen for connections with the listen( ) system call
4. Accept a connection with the accept( ) system call. This call typically blocks until a client connects with the server.
5. Send and receive data

Architechture Types:

The vast majority of end user applications consist of three components: presentation, processing, and data. The client/server architectures can be defined by how these components are split up among software entities and distributed on a network.

1 Two-tier Architecture

In this implementation, the three components of an application (presentation, processing, and data) are divided among two software entities (tiers): client application code and database server. A robust client application development language and a versatile mechanism for transmitting client requests to the server are essential for a two tier implementation. Presentation is handled exclusively by the client, processing is split between client and server, and data is stored on and accessed via the server.

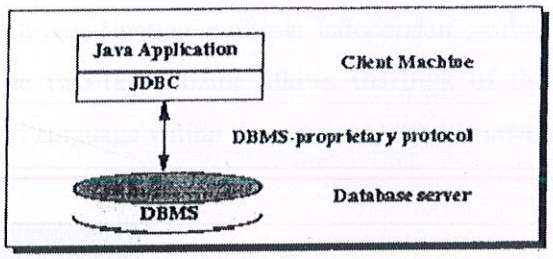


Fig 1.2

2. Three Tier Architecture

The three tier architecture attempts to overcome some of the limitations of the two-tier scheme by separating presentation, processing, and data into *separate,* distinct software entities (tiers). When calculations or data access is required by the presentation client, a call is made to a middle tier functionality server. This tier can perform calculations or can make requests as a client to additional servers. The middle tier servers are typically coded in a highly-portable, non-proprietary language such as C, Middle-tier functionality servers may be multi-threaded and can be accessed by multiple clients, even those from separate applications.

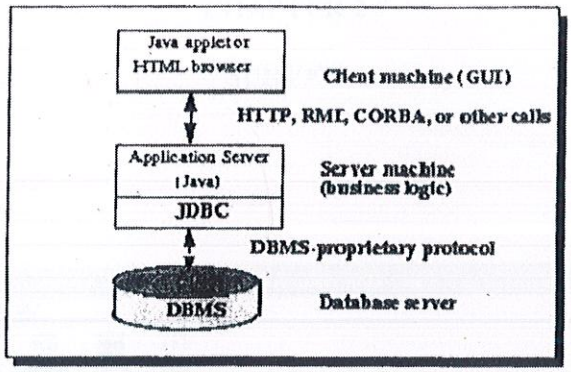


Fig 1.3

Comparison between two –tier & Three-Tier Architecture:

Comparison between Two-Tier & Three-Tier Architecture:

The three tier application takes much longer to develop - this is primarily due to the complexity involved in coding the bulk of the application logic in a lower-level 3GL such as C and the difficulties associated with coordinating multiple independent software modules on disparate platforms. In contrast, the two-tier scheme allows the bulk of the application logic to be developed in a higher-level language within the same tool used to create the user interface.

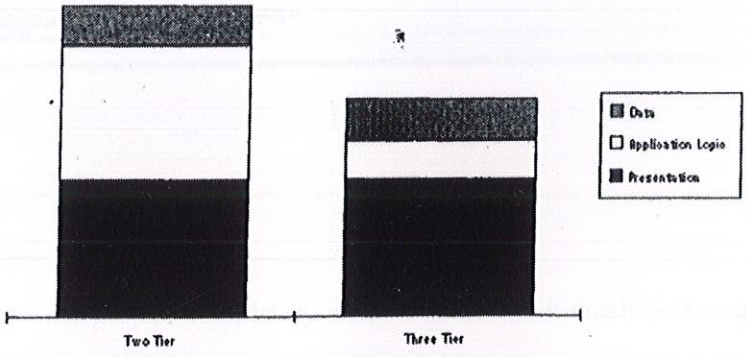


Figure 1.4

In view of the reason stated above our application uses two-tier architecture.

CHAPTER 2

SURVEY

2.1 DILLO

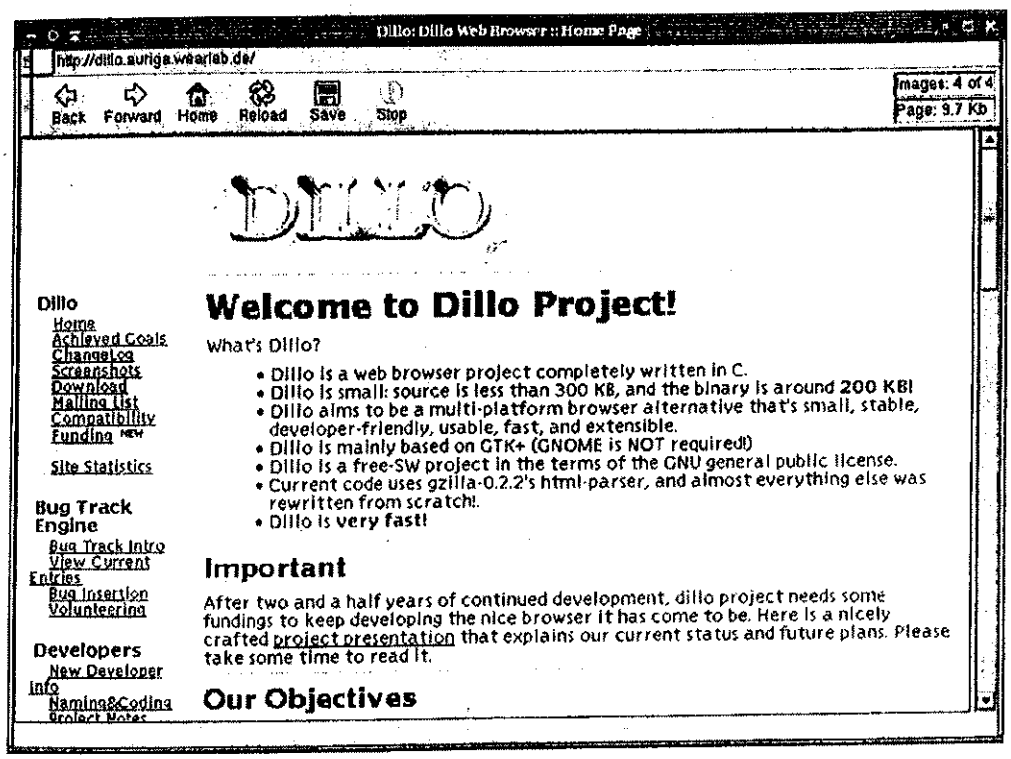


Figure 2.1

Dillo is web Browser project completely written in C. It's a small source and multi-platform browser

2.2 MULTIVALENT

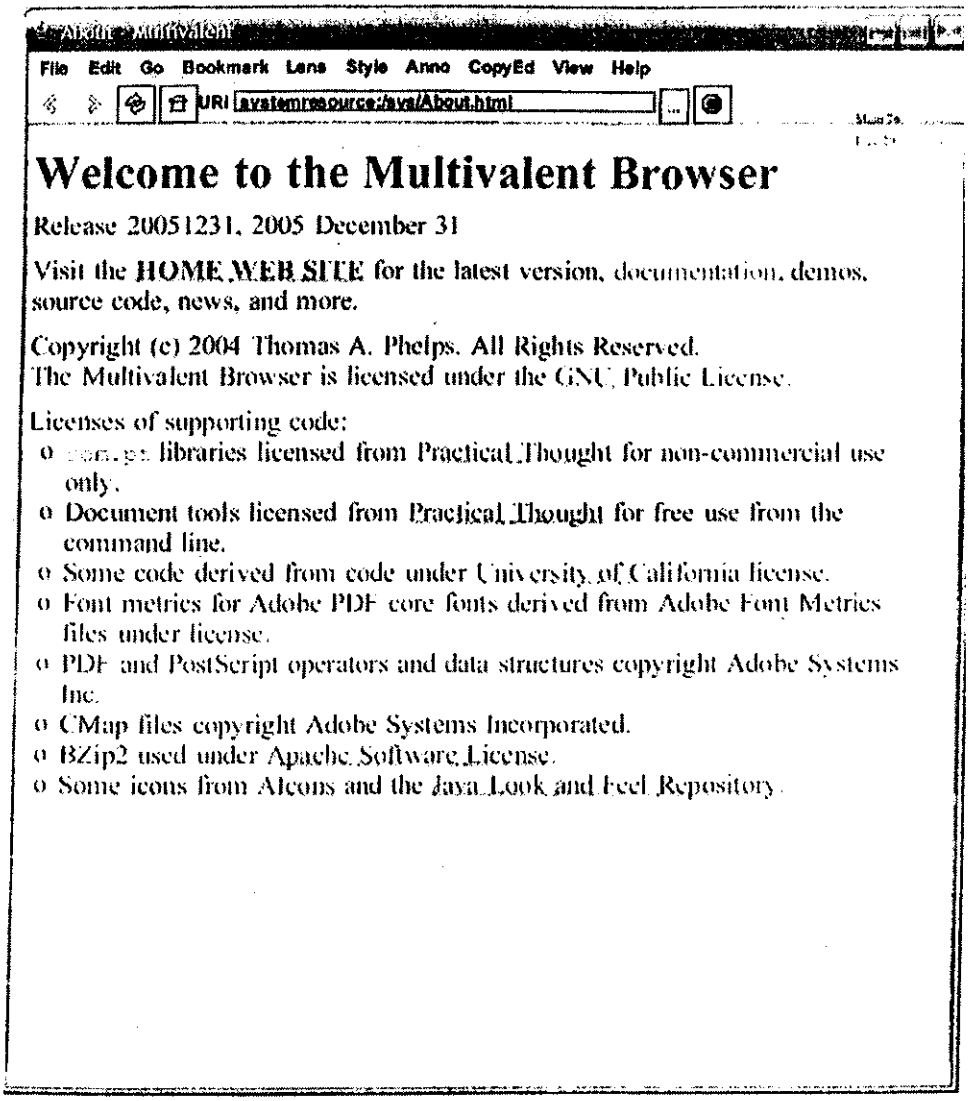


Fig 2.2

Company: UC Berkeley's Digital Library Project

License: Open source (GPL)

URL: multivalent.sourceforge.net

Type: 100 percent Java

Multivalent is an interesting research web browser. Meant primarily for browsing

documentation, its HTML features are a bit behind. It rendered Amazon pretty well, but showed only the unstyled version of the Zen Garden. It loaded Hamlet reasonably fast, but nothing spectacular. Strangely, I couldn't get it to load Slashdot. I kept getting GZip errors, but that may stem from some strange headers on Slashdot's front page. Multivalent supports complete visual and behavioral customization. Plus, since it's open source, you can always start banging on the code. It does have some interesting features, such as lenses for magnifying the screen, full text searching, on-screen annotation, PDF support, on-the-fly decompression, and a speed-reading mode.

Modem Compliance: Virtually none
Legacy Web: Poor
JavaScript: None
Hackability: Good
Speed: Good

2.3 Calpane:

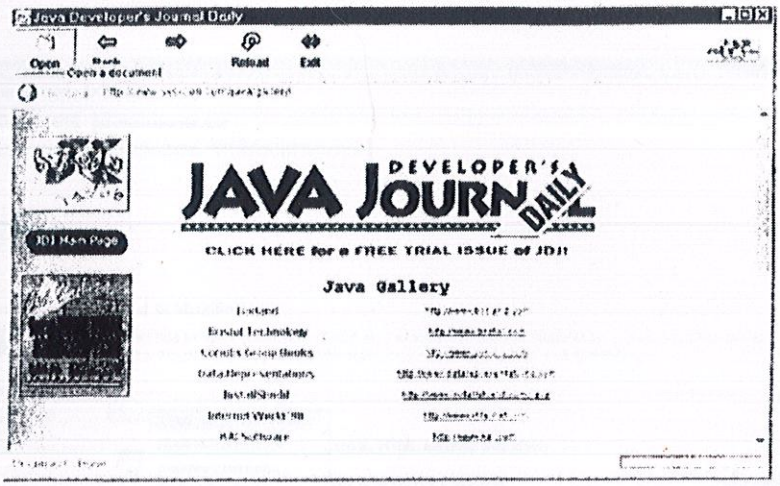


Fig 2.3

Company: Andrew Moulden

License: Free for non-commercial and some commercial apps.

URL: www.netcomuk.co.uk/~offshore/index.html

Type: 100 percent Java

CalPane is an older browser without JavaScript support, but it can render legacy HTML fairly well. As you can see in the screenshots below, both Amazon and Slashdot render pretty well, though the lack of anti-aliasing is especially apparent on Slashdot. It doesn't support CSS at all, though it does degrade properly. This also highlights the principle of using CSS properly so that sites are still usable without it. As far as speed goes, it is pretty snappy on pages it supports. There is support for event callbacks, and you can override certain features such as how images are loaded, but there isn't too much hackability. In the long run, the lack of CSS and XHTML means that more and more sites will fail in CalPane. The greatest problem with CalPane is that its site doesn't appear to have been updated since 2002.1 bent my own rule and included it in this roundup because the renderer is perfectly usable in its current state.

Modem Compliance: None

Legacy Web: Decent

JavaScript: None

Hackability: Decent

Speed: Good

2.4 JRex

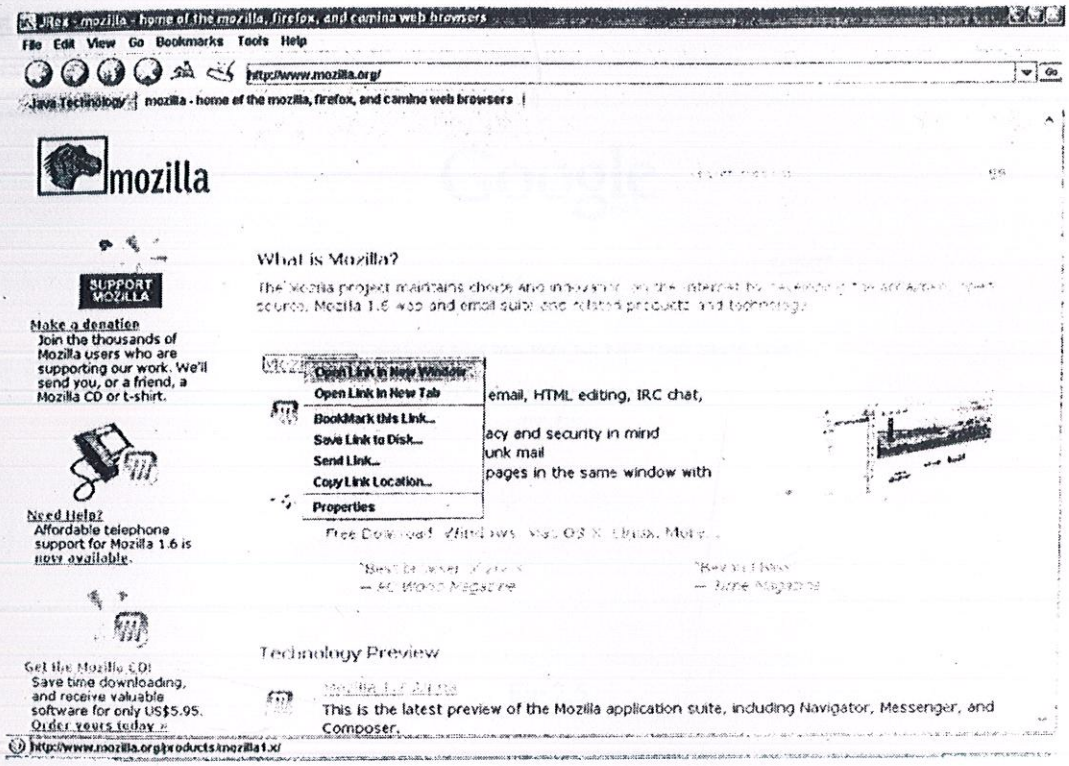


Fig 2.4

Company : Matt McBride

License: Open source (MPL)

URL: jrex.mcbridematt.dhs.org/

Type: 100 percent Java

JRex is a resurrection of the Javagator, Netscape's Navigator-in-Java project started before they open sourced Mozilla in 1998. Speed is poor, and the rendering for general web sites is almost unusable. Since it's based on so much legacy code, it will probably never support modem features such as CSS2. Still, it can be useful for certain things, especially where a small-footprint browser is required (such as a chat application).

Modern Compliance: Poor

Legacy Web: Poor

JavaScript: None

Hackability: Some

Speed: Slow

2.5 Simple Browser

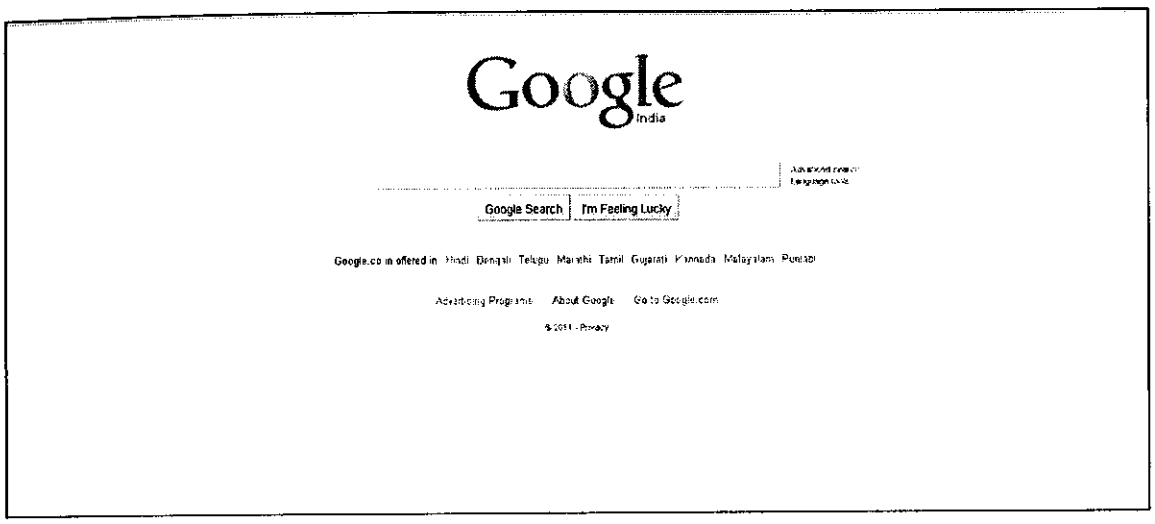


Fig 2.5

CHAPTER 3
FEASIBILITY ANALYSIS

3.1 Software requirements

Requires JDK 1.6

Operating system used: Window 98/2000/XP

Java Editor Kit used: JCreator

Back End : Oracle 9i

3.2 Hardware requirement

RAM : 64 MB

Hard Disk: 40 GB

Processor : Pentium

CHAPTER 4

TOOLS & TECHNOLOGY

4.1 JAVA & ADVANCED JAVA

4.1.1 JAVA

A program written in Java programming language is first translated into Java's intermediate language[1] (this is called "compiling", and the software used for compiling is called a "compiler"). The program is then executed on a Java virtual machine which interprets the intermediate language on some target computer.

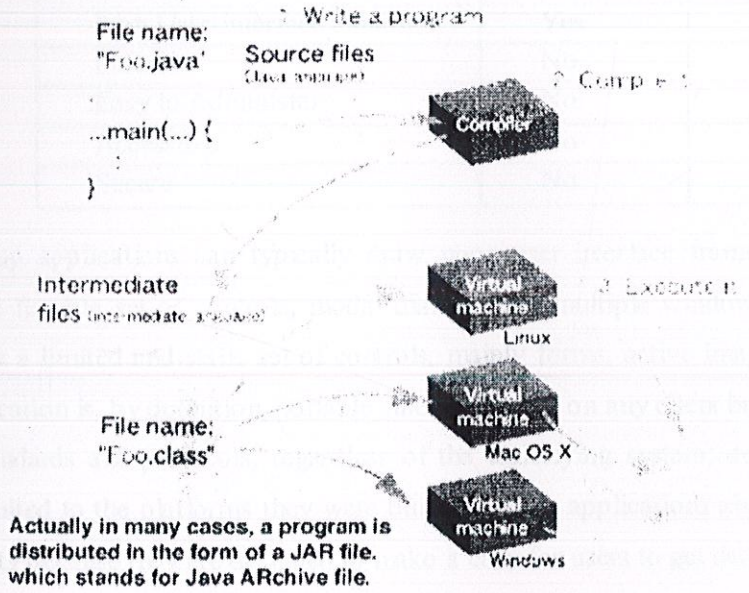


Fig 4.1

By preparing a Java virtual machine for each platform, one program translated into the intermediate language can be run on many different computers.

There are a number of advantages to be derived from a browser that is written in Java as opposed to a language compiled into native code. Namely:

- Security - In principle, a Java program is less susceptible to certain types of vulnerabilities such as a buffer overflow attack

- Expandability - A Java-based application can be implemented to be expandable via powerful cross-platform plugins.
- Portability - This is the obvious advantage of a pure Java application.

To understand the difference that Java makes in client-server architectures, consider two of the more common types of client-server applications: the traditional desktop application and the Web application. Rated on a set of desirable characteristics, each class of application has complementary strengths and weaknesses.

Characteristic	Desktop	Web
Interactive	Yes	No
Flexible Controls	Yes	No
Rich User-interface Paradigm	Yes	No
Portable	No	Yes
Easy to Administer	No	Yes
Accessible	No	Yes
Secure	No	Yes

Desktop applications can typically draw upon user-interface frameworks that provide a varied and flexible set of controls, modal dialogs, and multiple windows. On the other hand, HTML has a limited and static set of controls, mainly forms, active images, and hyperlinks. A Web application is, by definition, portable since it can run on any client browser that implements certain standards and protocols, regardless of the underlying system; desktop applications are usually limited to the platforms they were built for. Web applications also have high marks for accessibility because they are designed to make it easy for users to get data on networks. Finally, because sensitive data and business logic is confined to the server in Web applications, they tend to be more secure. Java scores high on each of these characteristics because it can have a strong presence on each side of the client-server divide. The principal advantage of Java is that it runs almost anywhere. The client need only have a compatible Java virtual machine (VM), something that most operating systems and browsers now include as a standard feature. A Java application can run on the server or can be downloaded to the client as an applet. AWT and JFC packages provide a rich source of flexible, interactive controls for developers.

4.1.2 GUI COMPONENTS

Overview of Swing Components

"Swing" refers to the new library of GUI controls (buttons, sliders, checkboxes, etc.) that replaces the somewhat weak and inflexible AWT controls. This tutorial is aimed at getting Java programmers who already know the AWT going as quickly as possible in Swing. Swing, which is an extension library to the AWT, includes new and improved components that enhance the look and functionality of GUIs. Swing can be used to build Standalone swing GUI Apps as well as Servlets and Applets. It employs model/view design architecture. Swing being 100% Java code makes it more portable and more flexible than AWT.

Swing Model/view design: The "view part" of the MV design is implemented with a component object and the UI object. The "model part" of the MV design is implemented by a model object and a change listener object.

Swing is built on top of AWT. Swing is written entirely in Java, using AWT's lightweight component support. In particular, unlike AWT, the architecture of Swing components makes it easy to customize both their appearance and behavior. Components from AWT and Swing can be mixed, allowing you to add Swing support to existing AWT-based programs. For example, swing components such as JSlider, JButton and JCheckbox could be used in the same program with standard AWT labels, text fields and scrollbars. You could subclass the existing Swing UI, model, or change listener classes without having to reinvent the entire implementation. Swing also has the ability to replace these objects on-the-fly

Java Swing Class hierarchy

The class JComponent, descended directly from container, is the root class for most of wing's user interface components.

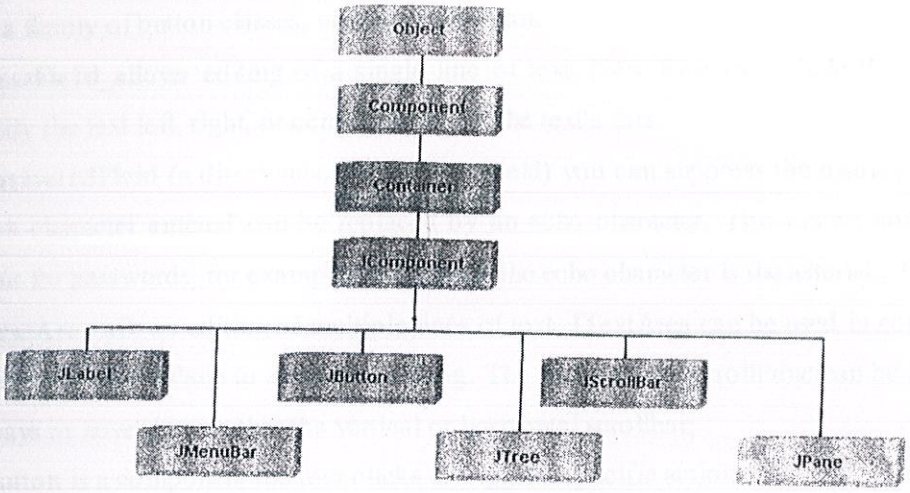


Fig 4.2

Common GUI components are as given below:

1. **JPanel** is Swing's version of the AWT class Panel and uses the same default layout, FlowLayout. JPanel is descended directly from JComponent.
2. **JFrame** is Swing's version of Frame and is descended directly from that class. The components added to the frame are referred to as its contents; these are managed by the *contentPane.* To add a component to a JFrame, we must use its contentPane instead.
3. **JInternalFrame** is confined to a visible area of a container it is placed in. It can be *iconified, maximized* and *layered.*
4. **JWindow** is Swing's version of Window and is descended directly from that class. Like Window, it uses BorderLayout by default.
5. **JDialog** is Swing's version of Dialog and is descended directly from that class. Like Dialog, it uses BorderLayout by default. Like JFrame and JWindow, JDialog contains a rootPane hierarchy including a contentPane, and it allows layered and glass panes. All dialogs are modal, which means the current thread is blocked until user interaction with it has been completed. JDialog class is intended as the basis for creating custom dialogs; however, some of the most common dialogs are provided through static methods in the class JoptionPane.
6. **JLabel,** descended from JComponent, is used to create text labels.
7. The abstract class AbstractButton extends class JComponent and provides a foundation

for a family of button classes, including **JButton.**

8. **JTextFieId_allows** editing of a single line of text. New features include the ability to justify the text left, right, or center, and to set the text's font.
9. **JPasswordField** (a direct subclass of JTextField) you can suppress the display of input. Each character entered can be replaced by an echo character. This allows confidential input for passwords, for example. By default, the echo character is the asterisk, *.
10. **JTextArea** allows editing of multiple lines of text. JTextArea can be used in conjunction with class JScrollPane to achieve scrolling. The underlying JScrollPane can be forced to always or never have either the vertical or horizontal scrollbar;
11. **JButton** is a component the user clicks to trigger a specific action.
12. **JRadioButton** is similar to JCheckbox, except for the default icon for each class. A set of radio buttons can be associated as a group in which only one button at a time can be selected.
13. **JCheckBox** is not a member of a checkbox group. A checkbox can be selected and deselected, and it also displays its current state.
14. **JComboBox** is like a drop down box. You can click a drop-down arrow and select an option from a list. For example, when the component has focus, pressing a key that corresponds to the first character in some entry's name selects that entry. A vertical scrollbar is used for longer lists.
15. **JList** provides a scrollable set of items from which one or more may be selected. JList can be populated from an Array or Vector. JList does not support scrolling directly; instead, the list must be associated with a scrollpane. The view port used by the scroll pane can also have a user-defined border. JList actions are handled using ListSelectionListener.
16. **JTabbedPane** contains a tab that can have a tool tip and a mnemonic, and it can display both text and an image.
17. **JToolbar** contains a number of components whose type is usually some kind of button which can also include separators to group related components within the toolbar.
18. **FlowLayout** when used arranges swing components from left to right until there's no more space available. Then it begins a new row below it and moves from left to right again. Each component in a FlowLayout gets as much space as it needs and no more.

19. **BorderLayout** places swing components in the North, South, East, West and center of a container. You can add horizontal and vertical gaps between the areas.
20. **GridLayout** is a layout manager that lays out a container's components in a rectangular grid. The container is divided into equal-sized rectangles, and one component is placed in each rectangle.
21. **GridBagLayout** is a layout manager that lays out a container's components in a grid of cells with each component occupying one or more cells, called its *display area.* The display area aligns components vertically and horizontally, without requiring that the components be of the same size.
22. **JMenubar** can contain several JMenu's. Each of the JMenu's can contain a series ofJMenuItem's that you can select. Swing provides support for pull-down and popup menus.
23. **Scrollable JPopupMenu** is a scrollable popup menu that can be used whenever we have so many items in a popup menu that exceeds the screen visible height.
24. JTree allows visualization, traversal and manipulation of hierarchical information (ie. parent-children) much easier. Trees consist of nodes. Common examples of trees are directories, organizational charts, and family trees.
25. **JFileChooser** class is part of the javax.swing.filechooser package. It allows browsing through a file system and selecting appropriate files. The JFileChooser (string Var) constructor can be used to set the starting browsepoint (currentDirectory is common). An external FileFilter extension or a *FileNameExtensionFilter* (depending on Java version) can be added using the *addChoosableFileFilter ()* method to limit the scope of the selection.
26. **Split Pane** allows you to place two components side by side in a single pane. The divider can be adjusted by the user. It can also split the pane horizontally.

4.1.3 JDBC

Introduction

The JDBC (Java Database Connectivity) API defines interfaces and classes for writing database applications in Java by making database connections. Using JDBC you can send SQL, PL/SQL statements to almost any relational database. JDBC is a Java API for executing SQL statements and supports basic SQL functionality. It provides RDBMS access by allowing you to embed SQL inside Java code. Because Java can run on a thin client, applets embedded in Web pages can contain downloadable JDBC code to enable remote database access. You will leam how to create a table, insert values into it, query the table, retrieve results, and update the table with the help of a JDBC Program example.

JDBC Architecture

It is divided into 2 parts:

- JDBC API (java.sql & javax.sql packages)
- JDBC Driver Types

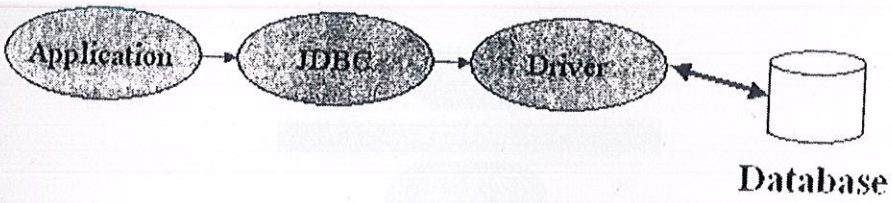


Fig. 4.3

JDBC API:

The JDBC API is available in the java.sql and javax.sql packages. Following are important JDBC classes, interfaces and exceptions in the java.sql package:

- **DriverManager** - Loads JDBC drivers in memory. Can also be used to open connections to a data source.
- **Connection** - Represents a connection with a data source. Is also used for creating Statement, PreparedStatement and CallableStatement objects.

- **Statement** - Represents a static SQL statement. Can be used to retrieve Result Set object/s.
- **PreparedStatement** - Higher performance alternative to Statement object represents a precompiled SQL statement.
- **CallableStatement** - Represents a stored procedure. Can be used execute stored procedures in a RDBMS which supports them.
- **ResultSet** - Represents a database result set generated by using a SELECT SQL statement.

- **SQLException** - An exception class which encapsulates database base access errors.

JDBC Driver Types :

There are 4 types of JDBC drivers. Commonest and most efficient of which are type 4 drivers. Here is the description of each of them:

- **JDBC Type 1 Driver** - They are JDBC-ODBC Bridge drivers. They delegate the work of data access to ODBC API. SUN provides a JDBC/ODBC driver implementation.

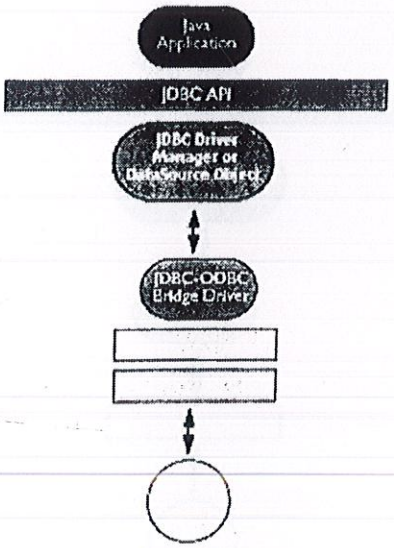


Fig 4.4

- **JDBC Type 2 Driver : -** They mainly use native API for data access and provided Java wrapper classes to be able to be invoked using JDBC drivers.

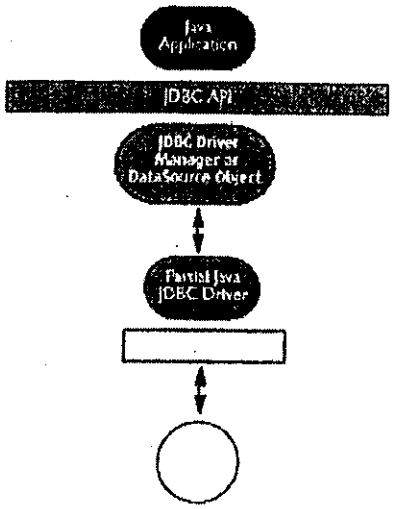


Fig. 4.5

- **JDBC Type 3 Driver -** They are written in 100% Java and use vendor independent Net-protocol to access a vendor independent remote listener, This listener in turn maps the vendor independent calls to vender dependent ones. This extra step adds complexity and decreases the data access efficiency

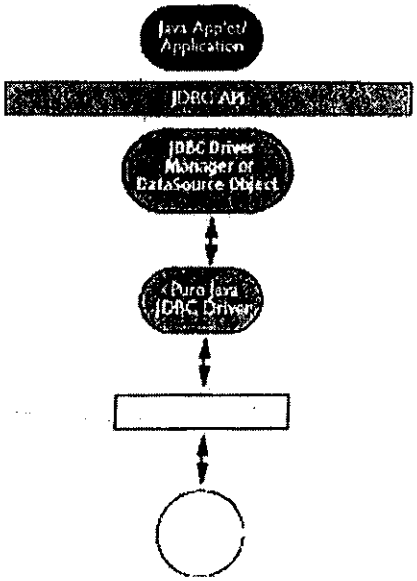


Fig 4.6

JDBC Type 4 Driver - They are also written in 100% Java and are the most efficient among all driver types.

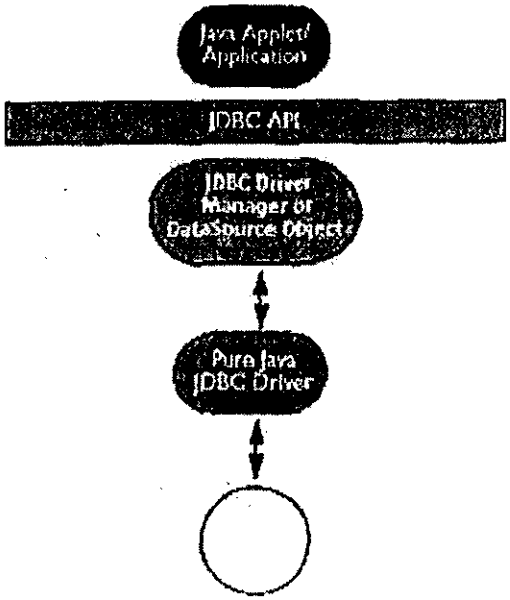


Fig 4.7

4.2 Oracle 10G

An **Oracle database** consists of a collection of data managed by an Oracle database management system. Popular generic usage also uses the term to refer to the Oracle DBMS management software, but not necessarily to a specific database under its control.

One can refer to the Oracle database management system unambiguously as **Oracle DBMS** or (since it manages databases which have relational characteristics) as **Oracle RDBMS.**

An Oracle database comprises at least one instance, along with data storage. An instance comprises a set of operating system processes and memory structures that interact with the storage. Typical processes include PMON (the process monitor) and SMON (the system monitor).

Users of Oracle databases refer to the server-side memory-structure as the SGA (System Global Area). The SGA typically holds cache information such as data-buffers, SQL commands and user information. In addition to storage, the database consists of online redo logs (which hold transactional history). Processes can in turn archive the online redo logs into archive logs (offline redo logs), which provide the basis (if necessary) for data recovery and for some forms of data replication.

The Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files. Tablespaces can contain various types of segments, for example. Data Segments, Index Segments etc. Segments in turn comprise one or more extents. Extents comprise groups of contiguous data blocks. Data blocks form the basic units of data storage. At the physical level, data-files comprise one or more data blocks, where the blocksize can vary between data-files.

Oracle database management keeps track of its data storage with the help of information stored in the SYSTEM tablespace. The SYSTEM tablespace contains the data dictionary — and often (by default) indexes and clusters. (A data dictionary consists of a special collection of tables that contains information about all user-objects in the database). Since version 10G, the Oracle RDBMS also supports "locally managed" tablespaces which can store space management information in bitmaps in their own headers rather than in the SYSTEM tablespace (as happens

with the default "dictionary-managed" tablespaces).

Edition Used In JBrowser:

Enterprise Edition (EE) includes more features than the 'Standard Edition', especially in the areas of performance and security. Oracle Corporation licenses this product on the basis of users or of processors, typically for servers running 4 or more CPUs. EE has no memory limits, and can utilize clustering using Oracle RAC software.

Version Used:

Client-Side: Oracle 10G

Server Side: Oracle 10G

CHAPTER 5

JBrowser

5.1 STATEMENT OF PURPOSE

JBrowser opens the page required by the user, specified in URL text field. It allows browsing and navigation through pages retrieved at any point of time. It allows users to stop browsing a particular page if not required. JBrowser provides downloading and intranet chatting facilities. It interacts with user and the world wide web.

5.2 CONTEXT DIAGRAM

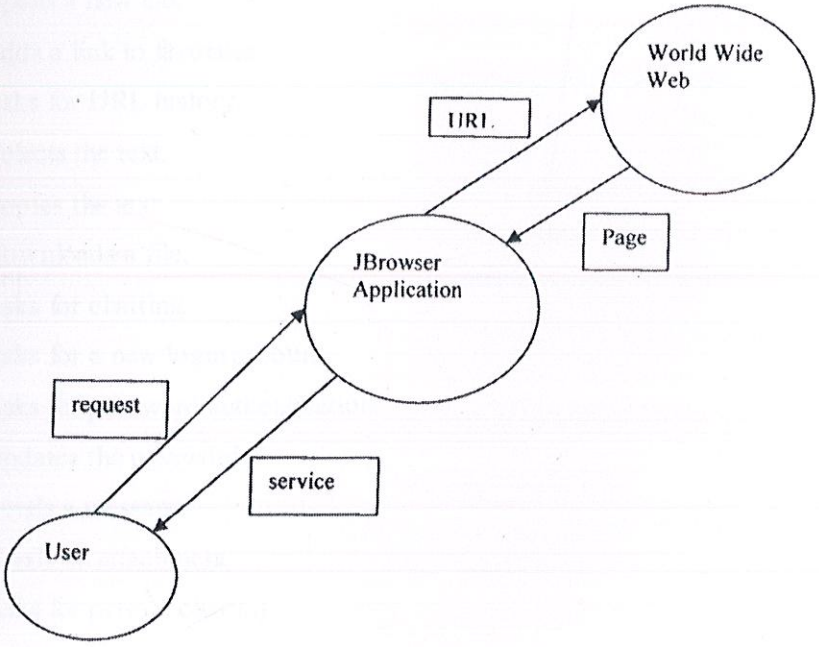


Fig 5.1

5.3 EVENT LIST

5.3.1 Normal Events

1. User asks for opening a page specified in URL text field.
2. User asks for backward navigation.
3. User asks for forward navigation.
4. User sets proxy host and port number.
5. User stops the loading of the web page.
6. User asks for refreshing the web page.
7. User asks for different working environment.
8. User saves a web page.
9. User saves a web page with a different name.
10. User opens a new tab.
11. User adds a link to favorites. "
12. User asks for URL history.
13. User selects the text.
14. User copies the text.
15. User downloads a file.
16. User asks for chatting.
17. User asks for a new login account.
18. User asks for password authentication.
19. User updates the password.
20. User sends a message.
21. User sends an attachment.
22. User asks for private chatting.
23. User ends the chat session.

5.3.2 Remind Events

1. User is informed that the URL specified is not correct.
2. User is informed that the sign up format is incorrect.
3. User is informed that the password/login id is incorrect.

5.4 CARTESIAN HIERARCHY

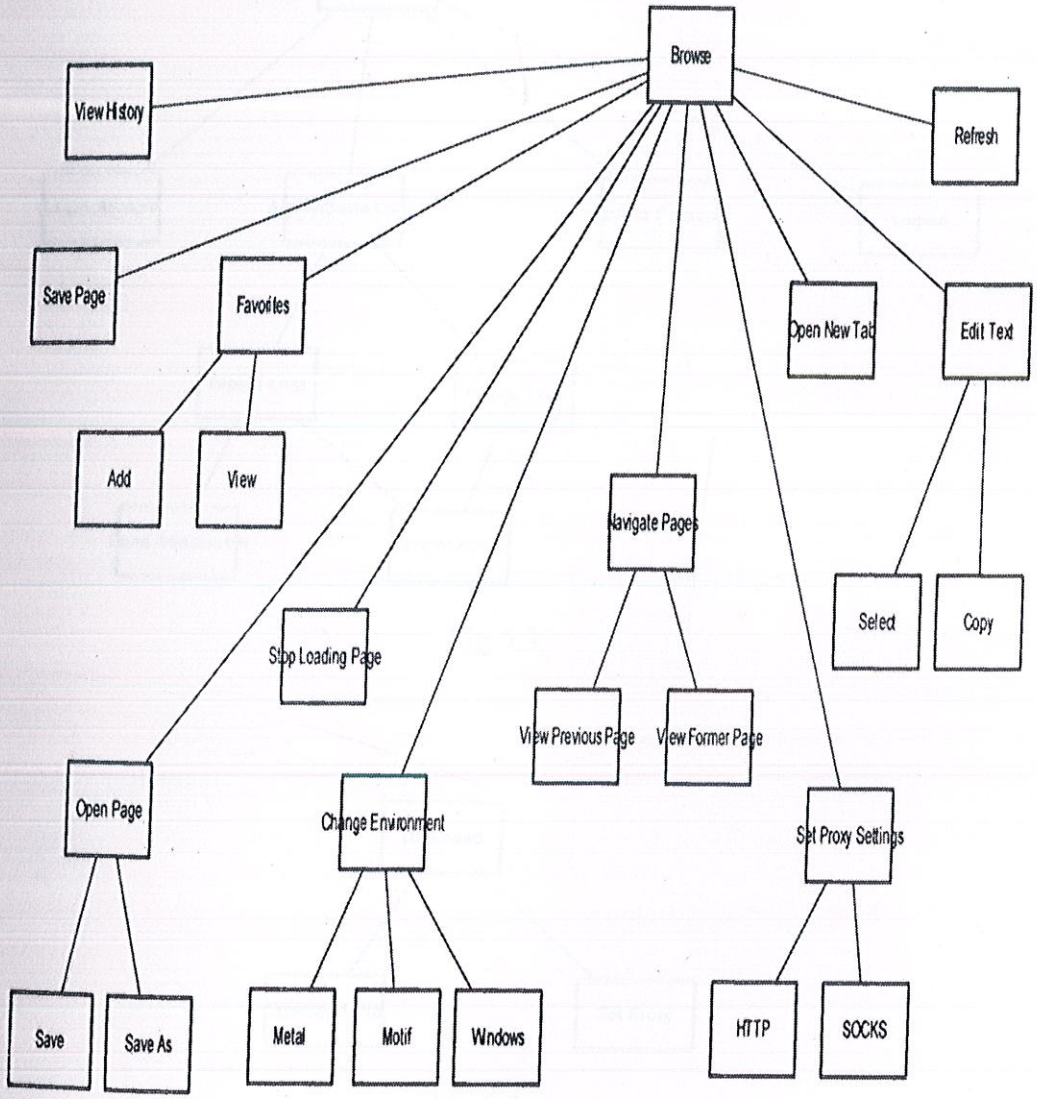


Fig 5.2

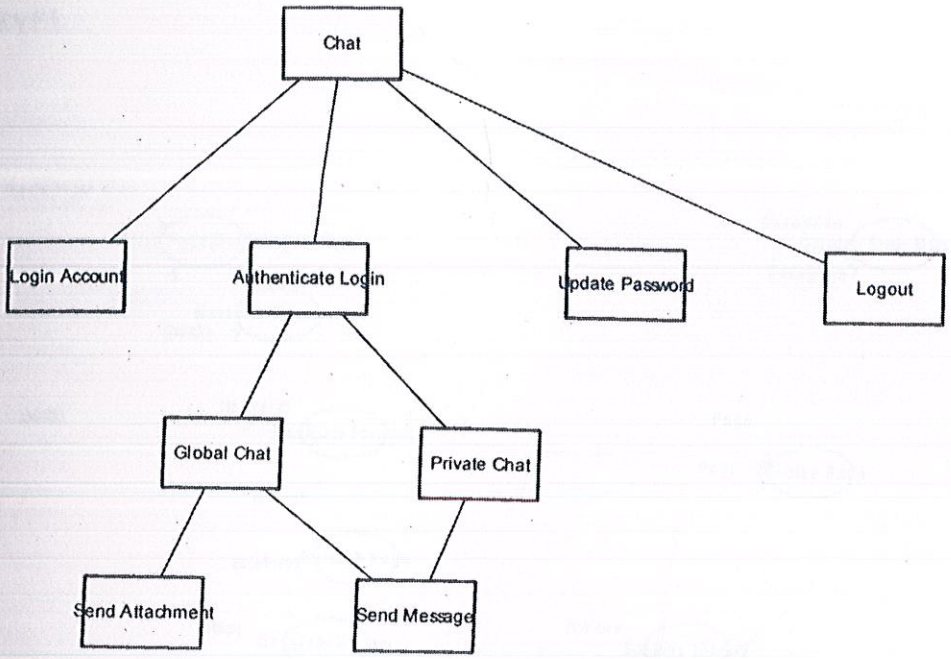

Chat
Login Account
Authenticate Login
Update Password
Logout
Global Chat
Private Chat
Send Attachment
Send Message

Fig 5.3

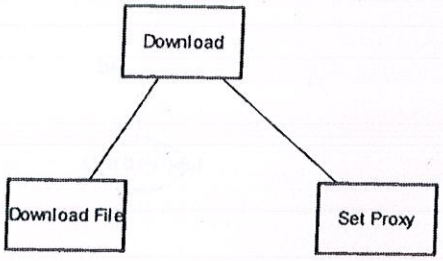

Download
Download File
Set Proxy

Fig 5.4

5.5 DATAFLOW DIAGRAMS

1st LEVEL

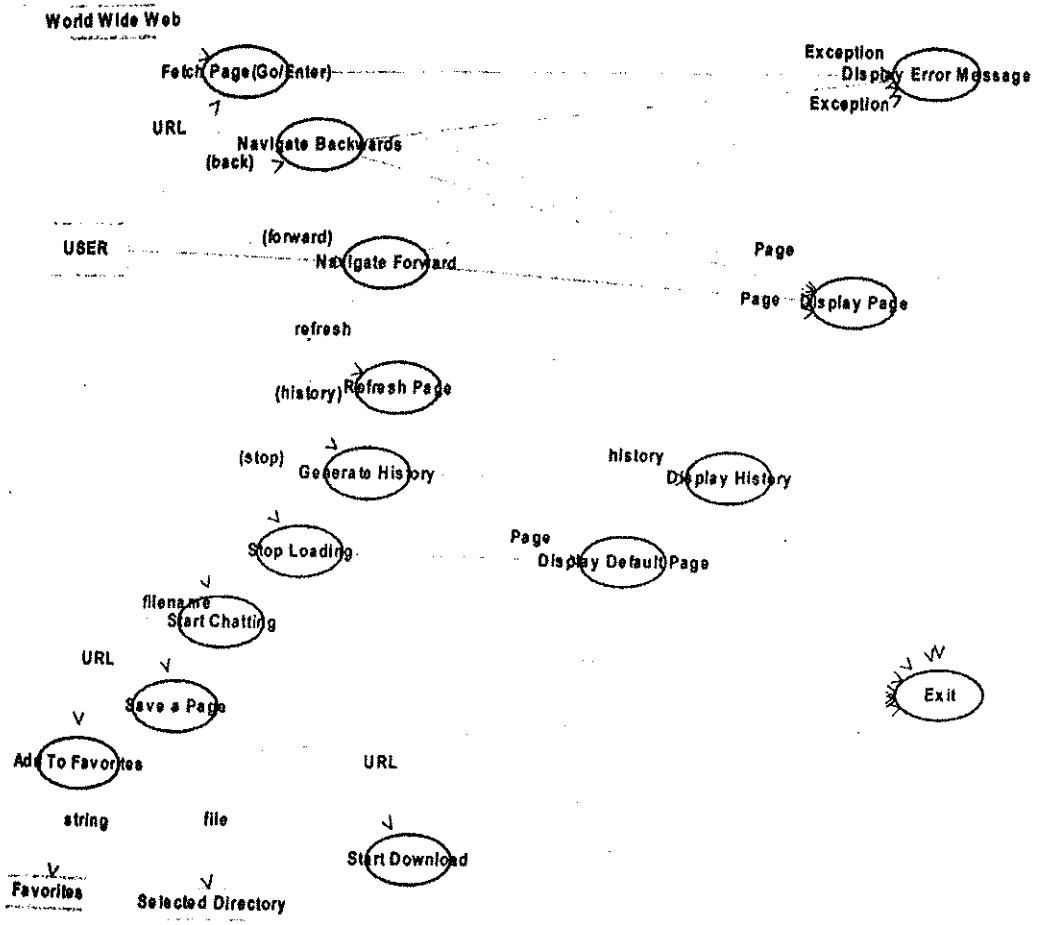


Fig 5.5

2nd LEVEL

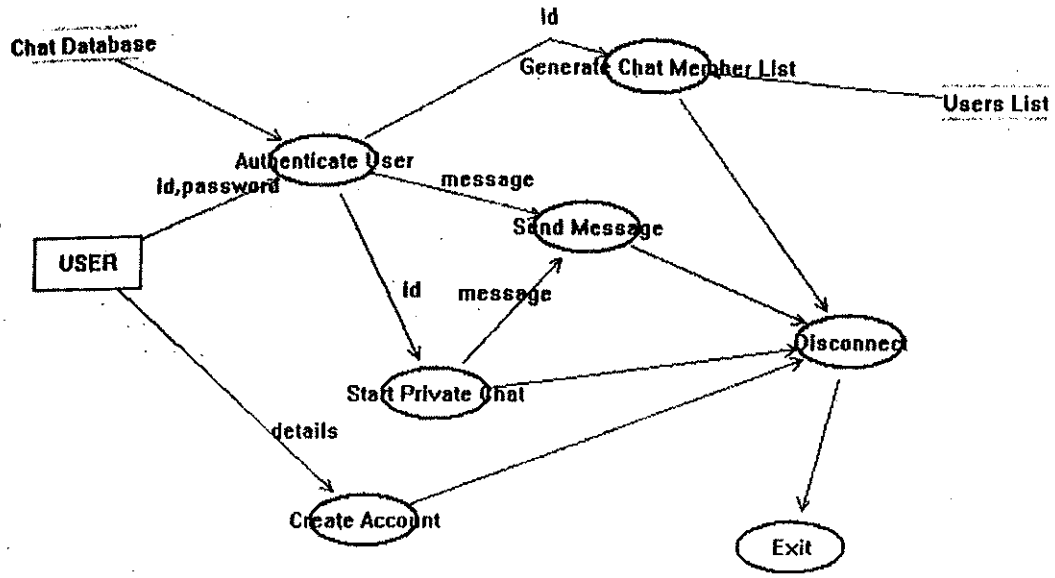


Fig 5.6

5.6 MODULAR DESIGN

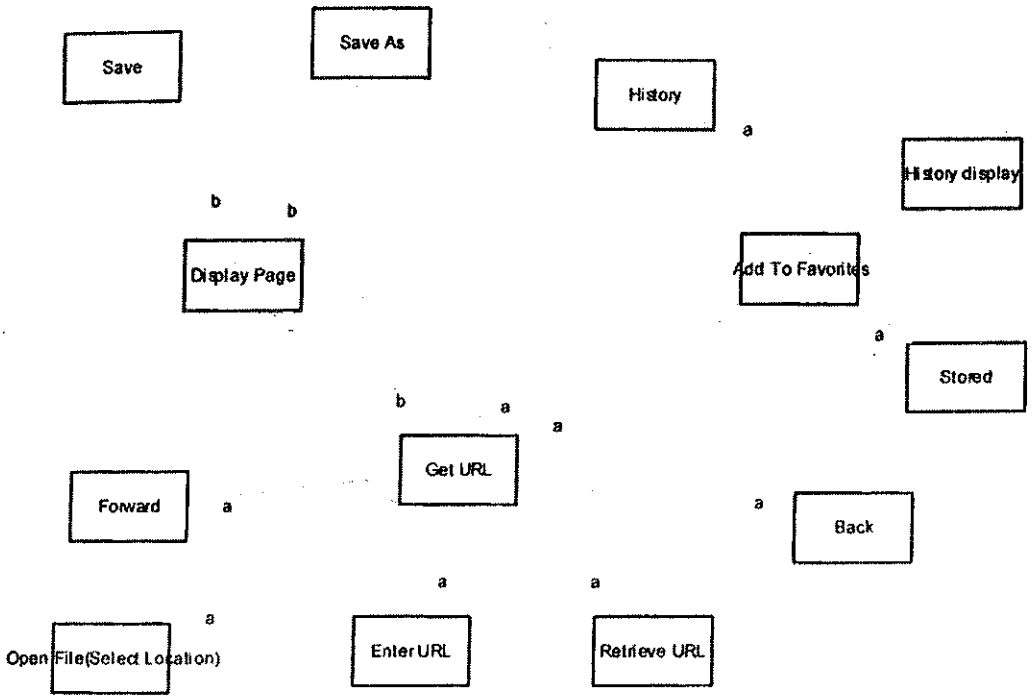


Fig 5.7

JBrowser

5.7 TESTING

5.7.1 Testing Criterion

Unit Tests will be preformed to test the proper functionality of each module. Each module will be tested to check all the conditions and options and the output.e.g. in the Reminder module, proper data is entered by the user as required an deny misinformation is not provided as far as possible ,also checks will be kept so as to assure that all the entries required are made by the user.

System testing will be performed to check whether the application performs according to expectations.

The Black Box Testing plays the major role because once the different modules are interconnected as we must test them for their correct working and secondly during the coding of different two modules.

The following tests will be performed:

- Black box test will measure the system for correctness of output once the system is integrated and is given some input.
- The browser opens the file effectively.

All the menu items of Java browser should work properly.

5.7.2 TEST CASES JBrowser

1. Valid Test Cases

1. http:\\www.astralreflections.com
2. C:\Document and Settings\native.txt

2. Invalid Test Cases I: Syntactical Errors

1. http://www.astralreflections.com

2. http:\\astralreflections.com
3. http:\\www.astralreflections
4. http:\\wwwastralreflections.com
5. http:\\www.astralreflectionscom
6. http:\\wwwastralreflectionscom
7. http:\www.astralreflections.com
8. http:www.astralreflections.com
9. httpNNwww.astralreflections.com

II.: Typographical Errors

1. http:\\www.yahooo.
2. http:\\ww.yahoo.com
3. http.AVwww.yahoo.com

Download Manager

The Download Manager can download files with any extension although it requires the user to manually enter the URL for the file that he needs to download.

Test Cases

1. http ://static.scribd.com/docs/736egzypod07v_files/
2. http://medial.santabanta.com/fulll/nature/motivational/mot62e.jpg
3. http://i.indiafm.com/posters/ash/ash53 .jpg
4. http://appldnld.apple.com.edgesuite.net/content.info.apple.com/iTu nes7/Win/061-3351.20070501 .nU8yb/iTunesSetup.exe
5. http://medial.santabanta.com/full/indian%20%20celebrities(f)/aishwarya%20rai/ais228v.jpg

5.8 COMPARISON

Comparison with other University Projects :

FEATURES	JBrowser	MULTIVALENT	Simple Browser
Inbuilt Chat	√	X	X
Proxy Support	√	X	X
JavaScript	X	X	X
Tabbed Pane	√	√	√
Font Settings	√	√	X
Lens Effects	X	√	X
Annotations	X	√	X
PDF support	X	√	X

Comparison with Comercial Web Browsers (Pure Java):

FEATURES	JBrowser	Jazilla	CalPane
Inbuilt Chat	√	X	X
Proxy Support	√	√	√
JavaScript	X	X	X
Tabbed Pane	√	√	X
Font Settings	√	X	X
Lens Effects	X	X	X
Annotations	X	X	X
PDF support	X	√	√

Comparison with C Browser:

FEATURES	JBrowser	Dillo
Inbuilt Chat	√	X
Proxy Support	√	√
JavaScript	X	√
Tabbed Pane	√	√
Font Settings	√	X
Lens Effects	X	X
Annotations	X	X
PDF support	X	√

5.9 INSTALLATION GUIDE

Installation Steps:

1. Assume that JDK1.6 is already installed on the host computer.
2. Add the Java files and Packages to \jdk1 .6\bin.
3. Open command prompt.
4. Set the current directory to \jdk1.6\bin.
5. Compile all the files. Use javac <fuename> Java.
6. Execute ReadNewFile.Use Java <filename>

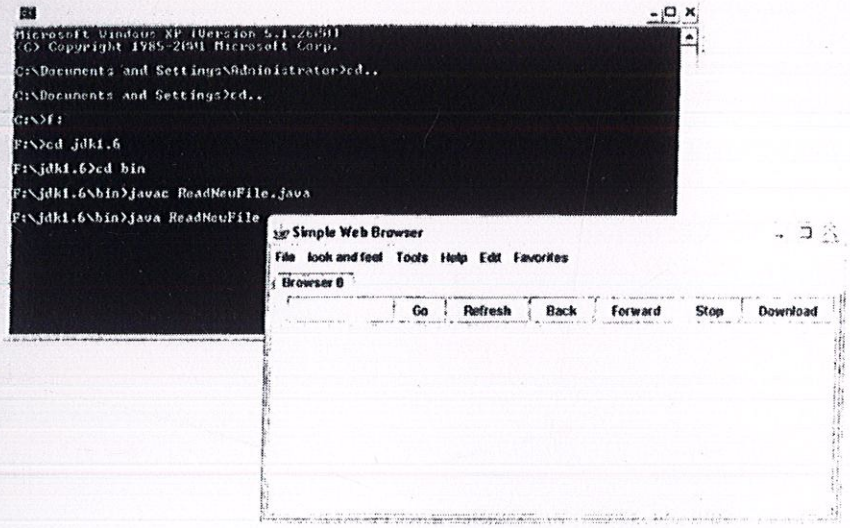


Fig 5.8

5.9 LIMITATIONS

1. Swing provides minimal support for JavaScript; hence, pages with extensive JavaScript Features are not rendered.
2. Secure pages are not rendered.

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

Based on the results and discussions presented in the report, the following major conclusions are drawn from the present study:

1. Jbrowser works fairly well in loading HTML pages.
2. Chat application works only on the intranet.
3. Download Manager downloads files of all formats with explicit URL Mention from the user.

6.2 SCOPE FOR FURTHER WORK

JAVA HTML Renderers are not widely used or developed. Moreover they are not open source. With more advancements in this field, JBrowser's support for JavaScript can be enhanced.It is yet required to enable the chat application to work on the internet on the lines of GTalk, Yahoo Messenger, AOL etc.The Download Manger needs to take an implicit URL feed for the files that the user wants to download.

Improvements in these areas can further enhance the working environment of JBrowser

BIBLIOGRAPHY

- JAVA How To Program By DEITEL & DEITEL
- Complete reference in JAVA By Herbert scildt
- www.wikipedia.org
- www.java2s.com
- www.berkeley.edu
- www.linuxdevices.com
- www.dillo.com
- www.javasketchbook.com
- www.java.sun.com
- www.jrex.mozdev.org
- www.multivalent.sourceforge.net
- www.jrex.mcbridematt.dhs.org
- www.netcomuk.co.uk/~offshore/index.html
- www.java.net

APPENDIX A

Read New File

```
import CHAT.*;
import CHAT.chatclientI;
import CHAT.chatserverI;
import LOAD.*;
import LOAD.Download;
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import Java.awt.event. *;
import java.awt.Component. *;
import java.io.IOException;
import java.io.*;
import javax.swing. JEditorPane;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing. JTextField;
import javax.swing.event.HyperlinkEvent;
import javax.swing.event.HyperlinkListener;
import java.util.*;
import java.util.regex.*;
import javax.swing.*;
import javax.swing.JProgressBar;
import Java.awt. *;
import java.awt.event.*;
import javax.swing.event.*;
import java.net.*;
import java.lang.*;
import java.net.URL.*;
import javax. swing.text.Document;
import javax. swing.text.*;
import java.awt.Dialog.*;
import javax.swing.text. *;
import java.awt. *;
import java.awt.event. *;
import java.awt.datatransfer. *;
import javax.swing.*;
import java.util.Vector;
```

```
import java.lang.Runtime;
import java.lang.String;
import Javax.swing.tree.DefaultMutableTreeNode;
import j avax.swing. JTree.*;
import java.net.*;
import java.util.*;
import javax.swing.border. *;
import Java.beans.*;
import java.awt.TextComponent;
import javax.swing.text.DefaultHighlighter;
import javax. swing.text.Highlighter;
import javax. swing.text.BadLocationException;
class WebBrowserPane extends JEditorPane{
public WebBrowserPaneQ
 { setEditable(false);
 public String getThePage( String location )
   {
     try   { setPage( location );
     } catch (lOException ioException )
      { JOptionPane.showMessageDialog(this,
          "Error retrieving specified URL", "Bad URL",
          JOptionPane.ERROR_MESSAGE );
    } return location;
)
public class ReadNewFile extends JFrame
{
                              private final String strings[] = { "Metal", "Motif, "Windows" };
```

```
private UIManager.LookAndFeelInfo looks[];
private JRadioButtonMenuItem lookandfeel[];
private ButtonGroup lookandfeelButtonGroup;
private JTabbedPane tabbedPane=new JTabbedPane();
public JToolBar tb=new JToolBarQ;
public WebToolBar array [J=new WebToolBar[50];
public static int i=l;
private int k;
private int SIZE=1;
private JTree tree;
public String childlnfo;
public DefaultMutableTreeNode top;
public DefaultMutableTreeNode var;
public DefaultMutableTreeNode child;
public static String forsave;
private Vector arr==new Vector();
private Vector krr=new VectorQ;
private JSplitPane
jsp=newJSplitPane(JSplitPane.HORIZONTAL_SPLIT);
private int ip=0;
public Vector fav_vector = new VectorQ;
public Scrollbar favsb = new Scrollbar(Scrollbar. VERTICAL);
public JMenu fav_view = new JMenu("View");
public JPanel fav_panel = new JPanelQ;
public JFrame fav_frame = new JFrameQ;

public ReadNewFile()
{
super( "JBrowser");
getContentPane().add(jsp, BorderLayout.CENTER);
      j sp. setContinuousLayout(true);
      j sp. setOneTouchExpandable(true);
      j sp. setVisible(true);
      createNewTab(O);
      jsp.setRightComponent(tabbedPane);
      JMenu fileMenu^new JMenu("File");
      JMenu helpMenu=new JMenu("Help");
      JMenu toolMenu = new JMenu("Tools");
      JMenu editMenu=new JMenu("Edit");
      JMenuItem prefrencesItem = new JMenuItem("Prefrences");
```

```
JMenuItem saveltem new JMenuItem("Save As");
JMenuItem save = new JMenuItem("Save");
    JMenuItem aboutItem=new JMenuItem("About");
    JMenuItem contentItem=new JMenuItem("Contents And Index");
    JMenuItem openItem==new JMenuItem("Open");
    JMenuItem selectItem=new JMenuItem("Select All");
    JMenuItem copyItem=new JMenuItem("Copy");
    JMenuItem chatItem=new JMenuItem("chat");
    JMenuItem history=new JMenuItem("history");
    JButton hb = new JButton("Find");
    JMenu favMenu == new JMenu("Favorites");
    JMenuItem fav_add = new JMenuItem("Add");
    final JTextField find=new JTextField(10);
    copy!tem.setAccelerator(KeyStroke.getKeyStroke(
        KeyEvent.VK_C, ActionEvent.CTRL_MASK));
    selectItem.setAccelerator(KeyStroke.getKeyStroke(
    KeyEvent.VK_A,ActionEvent.CTRL_MASK));
    copyItem.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent h)
      { returntab(tabbedPane.getSelectedIndex()).webBrowserPane.copy();
      } });
    select!tem.addActionListener(new ActionListenerQ { public void
                actionPerformed(ActionEvent h)
      { returntab(tabbedPane.getSelectedIndex()).webBrowserPane.select
A11Q;
      } });
    prefrencesItem.addActionListener(newActionListener()
    {
      public void actionPerfbrmed(ActionEvent event)
      {
            JDialog d==new optionDialogQ;
            d.setSize(300,300);
            d.show();
      } });
    history.addActionListener(newActionListener()
    {
public void actionPerformed(ActionEvent event)
  {
        String[] paren;
        String [] chi;
```

```
            String temp;
             intj=0;
int v;
int count;
                for(k=0;k<i;k++){
                      j=array[k].his.size();
                       for(v=0;v<j;v++){
                                                arr.addElement((array[k] .his.elementAt(v)));
                                                krr.addElement((array[k].bck.elementAt(v)));
                       }
                }
                  if(SIZE==l){
                      top=newDefaultMutableTreeNode("top");
                   var=new
DefaultMutableTreeNode((krr.elementAt(0)));
                      top.add(var);}   for(v=  1   ;v<(arr.size())-1
                  ;v++)
                    . {
                      System.out.println(krr.elementAt(v));
                                            if(((Strmg)(arr.elementAt(v))).endsWith(" 1"))
                       {
child=new DefaultMutableTreeNode((krr.elementAt(v)));
                            var.add(child);
                       } else
                       {
                            var=new
DefaultMutableTreeNode((krr.elementAt(v)));
                           top.add(var);
                           }
                          }
            tree=new JTree(top);
                    jsp.setVisible(true);
                    jsp.setLeftComponent(tree);
                      jsp.setOneTouchExpandable(true);
                       jsp.showQ;
                       ++SIZE;
          }
        });
        saveltem.addActionListener(
        new ActionListenerQ
```

```
{
public void actionPerformed(ActionEvent e)
{
String name;
StringQ words;
String s;
JFileChooser fc;
File fFile;

name=retumtab(tabbedPane.getSelectedIndex()).urlTextField.getText();
    words=name.split("\\.");

fFile=new File(words[0]);

fc == new JFileChooserQ;
fc.setCurrentDirectory(new File("."));
fc.setSelectedFile (fFile);
    int return Val = fc.showSaveDialog(ReadNewFile.this);

fFile = fc.getSelectedFile ();
    writeFile
(fFile, returntab(tabbedPane.getSelectedIndex()).webBrowserPane.getText ());
}});
save.add Action Listener(
  new ActionListenerQ
{
public void actionPerformed(ActionEvente)
  {
String name;

String[] words;
String s;
File fFile;
name=retumtab(tabbedPane.getSelectedIndex()).urlTextField.getText();
        words=name.split("\\.");
fFile==new File(words[l]);
writeFile(fFile,    retumtab(tabbedPane.getSelectedIndex()).webBrowserPane.getText
());

}});
chatltem.addActionListener(
```

```
            new ActionListener()
    {
                        public void actionPerformed(ActionEvent e)
                        {
                    JFrame f= new chatclientIQ;
                }
                }
                );
                 find.addActionListener(
                new ActionListener()
                 {
                        public void actionPerformed(ActionEvent e)
                        {
                        int offset=0;
                        int i=0;
                        int len=0;
                        System.out.prmtln("here"+offset);
String text = retumtab(tabbedPane.getSelectedIndex()).webBrowserPane.getText();
                    offset = text.indexOf(find.getText());
                    len=fmd.getText().length();
                    System.out.prmtln("ofiset"+offset);
                    if (offset .'=-!)

                     {
```

```
    Highlighter
h=returntab(tabbedPane.getSelectedIndex()).webBrowserPane.getHighlig hter ( );
            h.removeAllHighlights( );
               try{
returntab(tabbedPane.getSelectedIndex()).webBrowserPane.setSelectionS tart(offset);

returntab(tabbedPane.getSelectedIndex()).webBrowserPane.setSelectionE nd(offset+len);

        for(i==offset;i<offset+len;i++)
        {h.addHighlight(i,i+1 , DefaultHighlighter.DefaultPainter);
        System. out.println(text. substring^, i+1));}}
        catch (BadLocationException ble){
                }
             }
             }
         }
        );
         openItem. addActionLi stener( new ActionListener()
           { public void actionPerformed(ActionEvent e)
           { JFileChooser fc;
             fc = new JFileChooserQ;
             int return Val= fc.showOpenDialog(ReadNewFile.this);
             fe.setCurrentDirectory(newFile("."));
             File file = fc.getSelectedFileQ;
             if(file!=null)
               {
                try{
               URL url = file. toURL();

    returntab(tabbedPane.getSelectedIndex()).urlTextField.setText( url.toString());
    (
                      new ActionListenerQ
                     {
                       public void actionPerformed( ActionEvent event)
                       {

returntab(tabbedPane.getSelectedIndex()).webBrowserPane.getThePage(
event.getActionCommand());

returntab(tabbedPane.getSelectedIndex()).bck.addElement(newString(
```

```
returntab(tabbedPane.getSelectedIndex()).urlTextField.getText()));
                        }
                        }
                    );
catch (Exception urle){System.out.println("cannot convert to uri");}
                } else {Sy stem. out.println(" cannot open file");}
          }})<
          fav_add.addActionListener(newActionListener()
            { public void actionPerformed(ActionEvent event)
        {
             try{
           RandomAccessFile       favfile       =       new
RandomAccessFile("fav.txt","rws");
        long len = favfile. lengthQ;
        favfile.seek(len);
        String s =
retumtab (tabbedPane.getSelectedIndex()).urlTextField.getTextO+"\n";
        favfile.writeBytes(s);
        favfile.closeQ;
        fav_vector.addElement(new String(s));
        favfunct();
```

```
}
        catch(IOException e){}
        }
        });
        favfunctQ;

    prefrencesItem.setMnemonic('P');
    toolMenu.add(prefrencesItem);
    fileMenu.add(openItem);
    fileMenu.add(save);
    fileMenu.add(saveItem);
    flleMenu.addSeparator();
    helpMenu.add(contentItem);
    helpMenu. add(aboutItem);
    editMenu.add(selectItem);
    editMenu. add(copyItem);
    fileMenu.add(chatItem);
    fileMenu.add(history);
    JMenu newMenu=new JMenu("new");
    newMenu.add(new NewTabActionQ);
    fileMenu.add(newMenu);
    favMenu. add(fav_add);
    favMenu.addSeparatorQ;
    favMenu.add(fav_view);

    JMenu lookAndFeelMenu = new JMenu("look and feel");
    lookandfeel = new JRadioButtonMenuItem[strings.lengthj;
    lookandfeelButtonGroup=newButtonGroup();
    ItemHandler handler = new ItemHandler();

    for(int count=0;count<strings. length; count++)
    {
    lookandfeel[count]=newJRadioButtonMenuItem(strmgs[countj);
     lookAndFeelMenu.add(lookandfeel[countj);
     lookandfeelButtonGroup.add( lookandfeel[ count ]);
     lookandfeel[ count ].addItemListener( handler);
    }
     looks = UIManager.getInstalledLookAndFeelsQ;
    lookandfeel[0].setSelected(true)
;
```

```
        lookAndFeelMenu.addSeparatorQ;

         JMenuBar bar = new JMenuBarQ;

         setJMenuBar(bar);

         bar.add(fileMenu);
         bar.add(lookAndFeelMenu);
         bar. add(toolMenu);
         bar.add(helpMenu);
         bar.add(editMenu);
         bar.add(find);
         bar.add(hb);

         bar. add(favMenu);

         setSize( 400, 300);
         setVisible( true);
       }
     public void favfunct()
      {
      try
        {
RandomAccessFile favf= newRandomAccessFile("fav.txt","r");
        String s;
        fav_vector.removeAHElements();
        while(favf.readBooleanQ)
          { s = favf.readLineQ;
         fav_ vector. addElement(new String(s));
         favf.close();
   } catch(IOException e){ }
        JButton ok = new JButton("OK");

     JFrame fav_frame = new JFrame("favorites");
          JPanel fav_panel = new JPanelQ;
          fav_frame .getContentPaneQ. add(fav_panel);

          fav_view. add(fav_panel);

         JList favList = new JList(fav_vector);
```

```
            fav_panel. add(favList);

            fav_panel.setPreferredSize(new Dimension(700, 700));

           fav_frame.setSize(700,700);

favList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

}

public void createNewTab(int i) {

JPanel panel== new JPanel(new BorderLayoutQ);
final WebBrowserPane browserPane = new WebBrowserPaneQ;
WebToolBar toolBar = new WebToolBar(browserPane);
array [i]=toolBar;
panel.add(array[i], BorderLayout.NORTH);
panel.add(new JScrollPane(browserPane), BorderLayout. CENTER);
tabbedPane.addTab("Browser " + tabbedPane.getTabCountQ, panel);

final JPopupMenu x = new JPopupMenuQ;


JMenuItem cp = new JMenuItem("Copy");

JMenuItem sl = new JMenuItem("Select All");
```

```
        cp.setAccelerator(KeyStroke.getKeyStroke(
        KeyEvent.VK_C,ActionEvent.CTRL_MASK));

        sl.setAccelerator(KeyStroke.getKeyStroke(
        KeyEvent.VK_A,ActionEvent.CTRL_MASK));

        x.add(cp);
        x.addSeparatorQ;
    x.add(sl);

            cp.addActionListener(new ActionListener() {
            public void actionPerformed( ActionEvent e )
            {
            browserPane.copy();
            }
            });

            sl.addActionListener(new ActionListener() {
            public void actionPerformed( ActionEvent e )
            {
            browserPane. selectAll();
            }
            });
            browserPane.addMouseListener( new MouseAdapterQ {
            public void mousePressed( MouseEvent e ) {
            checkForTriggerEvent(e);
            }
            public void mouseReleased( MouseEvent e ) {
            checkForTriggerEvent(e);
            } private void checkForTriggerEvent( MouseEvent e ) {
            If( e.isPopupTriggerQ)
            x.show( e.getComponent(), e.getX(), e.getY() );
            }
            });
    }
```

```
public WebToolBar returntab(int number){
        return (array[number]);
}
public static boolean writeFile (File file, String dataString) {
 try{
PrintWriter out =
new PrintWriter (new BufferedWriter (new FileWriter (file)));
   out.print (dataString);
   out.flush ();
   out.close ();
}
catch (IOException e) {
   return false;
}
return true;}
private class NewTabAction extends AbstractAction {
  public NewTabActionQ {
   putValue(Action.NAME, "New Browser Tab");
   putValue(Action.SHORT_DESCRIPTION, "Create New Web Browser Tab");
   putValue(Action.MNEMONIC_KEY, new IntegerCN'));
 }
 public void actionPerformed(ActionEvent event) {
   createNewTab(i);
   i++;
  }
 }
  public static void mam(String args[j) {
  ReadNewFile application = new ReadNewFileQ;
   application.setDefaultCloseOperation(EXIT_ON_CLOSE);}        private        void
    changeTheLookAndFeel( int value )
   {
   try
     {
     UIManager.setLookAndFeel( looks[ value J.getClassNameQ );
       SwingUtilities.updateComponentTreeUI( this );
     }
     catch (Exception exception )
     {
     exception.printStackTraceQ;
     }
```

```
    }
    private class Item-Handler implements ItemListener

   {
            public void itemStateChanged( ItemEvent event )
        {
                    for ( int count = 0; count < lookandfeel. length; count++ )
          { if( lookandfeel{ count J.isSelectedQ )
            (
              changeTheLookAndFeel( count);
        } } }
      }
    class WebToolBar extends JToolBar implements HyperlinkListener{ public
WebBrowserPane webBrowserPane;
    public JTextField urlTextField=new JTextField(25);
    public JButton back=new JButton("Back");
    public JButton go = new JButton("Go");
    public JButton refresh = new JButton( "Refresh");
    public JButton forward = new JButton("Forward");
    public JButton stop = new JButton(" Stop");
    public JButton load == new JButton("Download");
    private String location;
    public Vector bck==new VectorQ;
    public Vector his=new Vector();
    private int ip=0;
public    WebToolBar(WebBrowserPane    browser)    {    super("Web
  Navigation");
      webBrowserPane = browser;
      webBrowserPane.addHyperlinkListener(this);
      add(urlTextField);
       add(go);
              add(refresh);
                add(back);
              add(forward);
         add(stop);
         add(load);

       go. addAction Listener(
       new ActionListenerQ
       {
```

```
        public void actionPerformed( ActionEvent event)
        {
      webBrowserPane.getThePage( uriTextField.getTextQ);
      bck.addElement(new String(urlTextField.getText()));
      his.addElement(new String(urlTextField.getText()+0);
      }
      }
   );
              load.addActionListener(
               new ActionListener ( )
      {
        public void actionPerformed( ActionEvent event ) {
          String sl = urlTextField.getTextQ;
          URL u = null;
          try{
              u=newURL(sl);
          } catch(Exception e) {System.out.println("Error");}
         Download d = new Download(u);
  }
  }
);
              back.addActionListener(
               new ActionListenerQ
      {
      public void actionPerformed(ActionEvent be)
        {
        int i = bck.size();
         if (i>=l)
          {
```

```
                ip++;
                String s = bck.elementAt(bck.sizeO-l-ip).toStrmg();
                System.out.println("Back to : "+s);
                uri TextField. setText(s);
                webBrowserPane.getThePage(s );
            }
            }
        }
);
 forward.addActionListener(
 newActionListenerQ
    {
    public void actionPerformed(ActionEvent fe)
    {
    int i= bck.size();
    if(i >= 1)
    { IP";
     String s = bck.elementAt(bck.sizeO-(ip)-l).toString(),
     System.out.println("Forwrd to : "+s);
    url TextFi eld. setText(s);
    webBrowserPane.getThePage(s);
    }
 }
}
);
refresh.addActionListener(
new ActionListenerQ
{
public void actionPerformed( ActionEvent event )
        {
        webBrowserPane.getThePage( uriTextField.getText() );
    }
    }
);
stop.addActionListener(
new ActionListener ()
{
    public void actionPerformed( ActionEvent event)
    {
       webBrowserPane.getThePage(
```

```
"file:/C:/j7bin/pagecannotbeopened.html");
                }
            }
    );
urlTextField.addActionListener(new ActionListenerQ {
        public void actionPerformed(ActionEvent event)
{
webBrowserPane.getThePage( event.getActionCommand() );
    bck.addElement(new String(urlTextField.getText()));
    his.addElement(new String(urlTextField.getText())+0);
        }});}
    public void hyperlinkUpdate( HyperlinkEvent event)
    {
    if(event.getEventType()==
    HyperlinkEvent.EventType.ACTIVATED)
    {
    Location=webBrowserPane.getThePage( event.getURL().toString() );
    his.addElement(new String(location)+l);
    urlTextField.setText(location);
    bck.addElement(new Strmg(location));
        }
    }
}
    class optionDialog extends JDialog
          { public JButton button=new JButton("OK");
          public optionDialogO
           { JTabbedPane tab;
           tab=new JTabbedPaneQ;
       JPanelpl;
                p 1 =makepanel();
              pl.setSize(100,100);
       tab.add("connection",p 1);
```

```
        getContentPaneQ. add(tab);
        pack();
       }
       private JPanel makepanelQ
       {
       JPanel ans;
              final JTextField txt[];
              ans=new JPanelQ;
     JRadioButton options[];
         JLabel text[j;
    String str[]={"No Proxy","Socks Proxy","Http proxy"};
              String STR[]={"Host","Port"};
               options=newJRadioButton[str.lengthj;
               text=newJLabel[STR.length];
               txt=new JTextField[STR. length];
               for(mt i=0;i<str.length;i++)
               {
                        options[i]=new JRadioButton(str[ij);
             ans.add(options[i],BorderLayout.NORTH);
               }
               for(int j-0;j<STR.length;j++)
               {
                text(j]=newJLabel(STR[J]+": ");
               txt[j]=new JTextField(5);
       text[j].setLabelFor(txt(jj);
                        ans.add(text[j],BorderLayout.WEST);
                        ans.add(txt(j],BorderLayout.CENTER);
              }
      button.addActionListener(new ActionListenerQ {
      public void actionPerformed(ActionEvent event) {
    System.setProperty("http.proxyHost",txt[0].getText());
    System.setProperty("http.proxyPort",txt[l].getText());
    setVisible(false);
    }
}
);
    ans.add(button,BorderLayout. SOUTH);
                        return ans;
       }
```

}

APPENDIX B

B.I Chat Server

```
package CHAT;
import java.io.*;
import java.net.*;
import java.util.*;

public class chatserveri
{
static Vector serverthreads;
static Vector members;
static String names;

public static void main(String args[])throws lOException
{
serverthreads = new Vector();
 ServerSocket ss=new ServerSocket(l000);
 members=new VectorQ;

 while(true)
    {
     Socket s=ss.accept();
     serverthread st = new serverthread(s);
     st.start();
     serverthreads.addElement(st);
    }
}
public synchronized static void echoall(Strmg str)
{
 int strlen=str.length();
System.out.println("str="+str);
if(str.endsWith("Enters the Room!"))
  members.addElement(str.suhstring(0,strlen-17));
if(str.endsWith("Leaves the Room.'"))
   members.removeElement(str.substring(0,strlen-17));
```

```
        names=="";
        Enumeration emem=members.elements();
        while(emem.hasMoreElements()) names+=emem.nextElement()+"";
        //*********************************************/
        /* System, out.println(names);

         for(int i=0;i<members.size();i++)
         names+=members.elementAt(i)+" ";
        System.out.println(names);

         //*********************************************/
        Enumeration e=serverthreads.elements();
        while(e.hasMoreElementsQ)
           {
           serverthread st =(serverthread)e.nextElement();
           st.echo(names+":: "+str);
        }
        }
      }
      class serverthread extends Thread
       {
       BufferedReader br;
      PrintWriter pw;
      static int n=0;
      public serverthread(Socket s)
 {
 try{
      pw==new PrintWriter(s .getOutputStream(),true);
       br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
      n++;
      setName(Integer.toString(n));
     }catch(Exception e) {}
}
      public void run ( )
{

try{
while(true)
        {
```

```
        String str=null;
        chatserveri cs=new chatserverl ( );
        if((str=br.readLine())!=null)
        cs.echoall(str);
       }
    } catch(Exception e) {}
}
    public void echo(String str)
    {
    try{
       pw.println(str);
       }catch(Exception e) {}
    }
}
```

B.2 Chat Client

```
package CHAT;
import java.io.*;
import java.net.*;
import java.awt.*;
import java.sql.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.plaf.metal.*;
import com.sun.java.swing.plaf.windows.*;
import com.sun.java.swing.plaf.motif.*;

public class chatclienti extends JFrame implements
ActionListener,Runnable
{
public String str="";
public String sl="";
private JTextField tfh;
private JTextField tf;
private JPasswordField tfp;
private TextArea ta;
private Socket s;
private BufferedReader br;
private PrintWriter pw;
private List list;
private JButton Close;
private JButton Sign;
private JButton Browse;
private File f;
JMenuItem jl 1 J12J13J14;
JMenuItem jo1,jo2;
JMenuItem lfl,lf2,lf3;
JMenuItem jhl,jh2,jh3,jh4;
InetAddress add;
int counter=0;
public String name;
MetalLookAndFeel metalLf;
MotifLookAndFeel motifLf;
WindowsLookAndFeel windowsLf;
```

```
private Color color;

String msg=      "Welcome to the Java Chatting Program\n\n"+
                                "This Software is Developed by\n"+
                                  . "NITINW+
                          "B.E. (Computer Science & Engineering)\n"+
                          "M.E. (Software Engineering)\n"+
                          "Thapar Institute of Engineering and Technology\n"+
                          "(Deemed University), Patiala, Punjab\n"+
                                "Pin-147004, INDIA";

        String currmsgl=  "You are now changing your enviomment";
       String currmsg2 =  "You are presently Working in the same
     enviomment";
       String currmsg3 =  "You are now changing your enviornment";

    public chatclientl ( )
    { metalLf=new MetalLookAndFeel ( );
    motifLf==new MotifLookAndFeel( );
    windowsLf=new WindowsLookAndFeel ( );

    JMenuBarjmb=newJMenuBar( );
    setJMenuBar (jmb);
    JMenu jlogin=(JMenu)jmb.add(new JMenu("Login"));
    jlogin.setMnemonic('L');

    jlogm.add(jll=new JMenuItem(" Change User/Password"));
    j11.setMnemonic('U');
    j11.addActionListener(this);
jlogin.add(jl2=newJMenuItem("Disconnect"));
    jl2.setMnemonic('D');
    jl2.addActionListener(this);
jl2.addActionListener(newActionListenerO
    {public void actionPerformed (ActionEvent e)
     {                                ' chatclienti app = new chatclientl();
       } });

    jlogin.add(new JSeparatorQ);
    jlogin.add0'13=new JMenuItem("Close"));
    jl3.setMnemonic('e');
```

```
jl3 .addActionListener(this);
jlogin.add(jl4=newJMenuItem("Exit"));
jl4.setMnemonic('E');
jl4.addActionListener(this);
jmb.add(jlogin);

JMenu joptions=(JMenu)jmb.add(new JMenu("Options"));
Joptions.setMnemonic('O');
j options. add(j'ol=newJMenuItem( "Colors"));
jo1.setMnemomc('C');
jo1.addActionListener(this);
j options. add(new JSeparatorQ);
JMenu lf= new JMenu("Look & Feel"),
lf.setMnemonic('&');
lf.add(lfl=new JMenuItem("Motif"));
Ifl .setMnemonic('M');
Ifl .addActionListener(this); A lf.add
(lf2=newJMenuItem("Metal"));
lf2.setMnemonic('t');
jf2.addActionListener(this);
lf.add(lf3=newJMenuItem("Windows"));
lf3.setMnemonic('W);
jf3.addActionListener(this);
j options, add(lf);

JMenu jhelp=(JMenu)jmb.add(new JMenu("Help"));
jhelp.setMnemonic('H');
jhelp.add(jhl=newJMenuItem("Getting Started"));
jhl.setMnemonic('m');
jhl.addActionListener(new ActionListener ( )
{
public void actionPerformed (ActionEvent e)
{
TextAreaDemoH app = new TextAreaDemol4();
}
});
jhelp.add(new JSeparator());
jhelp.add(jh2=new JMenuItem( "About Project"));
jh2.setMnemomc('A');
jh2.addActionListener(new ActionListener ( )
```

```
public void actionPerformed (ActionEvent e)
{
JOptionPane.showMessageDialog(null, "" + msg /'About Java Chat
Program",JOptionPane.INFORMATION_MESSAGE);
}
});
getContentPane().setLayout(newGridBagLayout());
GridBagConstraints gbc=new GridBagConstraints();
gbc.fill=GridBagConstraints.HORIZONTAL;

gbc.weightx==0;
add(new JLabel("Enter Chat ID:"),gbc,0,0,1,1);
gbc.weightx==2;
tfn=new JTextField(15);
add(tfn,gbc,1,0,1,1);
Font f=new Font("Serif',Font.BOLD,14);
tfn.setFont(f);
tfn.setForeground(Color.black);

gbc.weightx=2;                                          Members in Chat
add(new JLabel("
Room :"),gbc,2,0,1,1);
gbc.weightx=2;
list=new List(7,false);
add(list,gbc,3,0,1,1);
list.addActionListener(this);

gbc.weightx=0;
add(new JLabel("Enter Password:"),gbc,0,1,1,1);
gbc.weightx=2;
tfp=new JPasswordField(8);
add(tfp,gbc,1,1,1,1);
tfp.addActionListener(this);

add(new JLabel("      "),gbc,0,2,1,1);

gbc.fill=GridBagConstraints.BOTH;
gbc.weightx=0;
add(new JLabel("Received Messages:"),gbc,0,3,1,1);
gbc.weightx=2;
```

```
gbc.weighty= 100;
ta=new TextArea();
add(ta,gbc,l,3,2,l);
ta.setEditable(false);

add(new JLabel("Click SignUp to Register"),gbc,3,0,0,0);
Sign=new JButton(" SignUp Now"), Sign.setMnemonic('S');
Close=new JButton("Logout");
Close.setMnemonic('G');
JPanel p=new JPanelQ;
p.setLayout(new GridLayout(2,1,100,180));
p.add(Sign);
p.add(Close);
add(p,gbc,3,3,2,l);
Sign.setToolTipText("SignUp to make New CHATID");
Sign.addActionListener(this);
Close.setToolTipText("Logout");
Close.addActionListener(this);

gbc.weightx=0;
gbc.weighty=0;
add(new JLabel("    "),gbc,0,4,1,1);
add(new JLabel("Send Messages:"),gbc,0,5,2,l);
gbc.weightx=2;
tf=new JTextField(30);
add(tf,gbc, 1,5,2,1);
tf.addActionListener(this);
tf.setEditable(false);
Font fff=new Font("Monospaced",Font.BOLD, 14);
tf.setFont(fff);
tf.setForeground(Color.red);
add(new JLabel("You are Welcome in Chat Room"),gbc,2,7,l,l);
add(new JLabel("Send Attachments"),gbc,3,7,50,50);
Browse=new JButton("Browse");
Browse.setMnemonic('B');
Browse.setToolTipText("Send an Attatchment..");
add(Browse,gbc,3,5,1,1);
Browse.addActionListener(this);
add(new JLabel("       "),gbc,0,6,l,l);
```

```
setSize(800,600);
setVisible(true)

try{
        s= new Socket(InetAddress.getByName(" 172.16.10.38"), 1000);

        br=newBufferedReader(newInputStreamReader(s.getInputStream()));
        pw=new PrmtWriter(s.getOutputStream(),true);
        } catch(IOException io) { }
        Thread t = new Thread(this);
    t.start ( );
    }
    private void add(Component c,GridBagConstraints gbc,int x,int y,int w,int h)
     {
     gbc.gridx=x;
     gbc.gridy=y;
     gbc.gridwidth=w;
     gbc.gridheight=h;
     getContentPane().add(c,gbc);
     }
    int n=12,start=0;
    public void run()
     {
     try{
     while(true)
       {
       Font f=new Font("SanSerif',Font.BOLD, 14);
       ta.setFont(f);
       ta.setForeground(Color.green);
       str=br.readLine();
       String msg=str.substring(str.mdex0f(":::")+3,str.length());
       if(!str.endsWith("pcb"))
       ta.appendText(msg+"\n");
        String memb=str.substring(0,str.index0f(":::")+1);
        String arr[]==new String[50];
        inti=0,j;
       while(memb.index0f(" ")!=-1)
         {
         arr[i]=memb.substring(0,memb.index0f(" "));
              memb=memb.substring(memb.index0f(" ")+1 ,memb.length());
```

```
            i++;
}
list.removeAll ( )
            Font ff=new Font("Courier",Font.BOLD,14);
            list.setFont(ff);
            list.setForeground(Color.blue);

     for(j=0;j<i;j++) list.add(arr[j]);
     }
    if((msg.endsWith("replyforpcb"))&&(msg.startsWith(tfn.getText())))
    {
    String pname=msg.substring(msg.indexOf(' '),msg.lastIndexOf("));
     String pn=pname.substring(l,pname.length()-2);
     String chknode=" select NODE_ADD from chatdata where ID='"+pn+"'";
          jdbcl db5=newjdbcl(chknode);
          String mn=db5.echonode();
          String mname=mn.substring(0,mn.indexOf('/'));
          PrivateChatBox pcb=new PrivateChatBox(name,mname);
          pcb.setTitle("Private Chat Window");
          }
        }
        } catch(Exception e) {}
       }
      public void actionPerformed(ActionEvent ae)
       { String sl=null;
       if(ae.getSource()==j 11)
         {
         if(!tfn.getText().equals(""))
           {
           chanpassdig dlg=new chanpassdlg(this,"Change Password");
           dlg.setVisible(true);
              }
            else
      JOptionPane.showMessageDialog(null,"First Log in \n Then Change
      your Password", "Warning", JOptionPane.WARNING_MESSAGE);
           }
  if ae.getSource ( ) list)
  {

          String query="select
```

```
NAME,ADDRESS,CITY,NATIONALITY,SEX,E_MAIL,NODE_ADD
    from chatdata where ID='"+list.getSelectedItem()+"'";
       jdbc1 db1=newjdbc1 (query);
       pw.println(list.getSelectedItem()+" "+name+" reply forpcb");
       PrivateChatServer pcs ==new PrivateChatServer(name);
       pcs.setTitle("Private Chat Window");
       }
      if(ae.getSource()===tf)
       {
       s1=tf.getText();
       try{
           pw.println(name+s 1);
     }catch(Exception e) {}
     tf.setText("");
       }
      if (ae.getSource()==Sign)
        {
        FillForm mf=new FillForm();
         mf.setSize(640,480);
       mf.setVisible(true);
       mf.setTitle("SignUp Form");
       }
      If (ae.getSourceO^^tfp)
        {
       tfp=(JPasswordField) ae.getSourceQ;
        s1=new String(tfp.getPassword());
   String passquerry="select PASSWORD from chatdata where
ID='"+tm.getTextO+"'";
        jdbc1 db2==newjdbc1(passquerry);
        String password=db2.echopass();
        if(!s1 .equals(password))
          JOptionPane.showMessageDialog(null, "Invalid "+
          "Password \nTry Again... \n or\n"+
          "Sign in for New Id" /'Warning",
                    JOptionPane.WARNING_MESSAGE);
       else
          {

            counter++;
            if(counter<=1)
```

```
            {
             name=tfn.getText()+": ";
            tfn.setEditable(false);
                JOptionPane.showMessageDialog(null,"Logged on Successfully"
                        Welcome to Chatroom",
                    JOptionPane.INFORMATION_MESSAGE);
            tfp.setEditable(false);
            tf.setEditable(tme);
            pw.println(tfn.getText()+" Enters the Room!");
           }
           tfp.setText("");
           setTitle("                                    "+-
                                        "Chat Window of"+
                tfn.getTextQ);
           try{
              add==InetAddress. getLocalHost();
             } catch(UnknownHostException uhe) {}
             String update="update chatdata set NODE_ADD='"+add+'"
where ID='";
           update+=tfn.getText()+'"";
           jdbcl db3=newjdbcl (update);
       }
    }

    if((ae.getSource()==Close))
    {
     pw.println(tfn.getText()+" Leaves the Room!");
     System.exit(0);
     }
    if((ae.getSource()==J'13)[|(ae.getSourceO==jl4))
     {
     pw.println(tfn.getText()+" Leaves the Room!");
     System.exit(0);
     }
```

```
if(ae.getSource()==Browse)
 {
 try
    {
    String readfile;
    JFileChooser fc=new JFileChooser();
    fc.setFileSelectionMode(JFileChooser.FILES_ONLY);
    int result=fc.showOpenDialog(this);
    if(result=-JFileChooser.CANCEL_OPTION)
    f=null;
    else
      f=fc.getSelectedFile();
     String filename=f.toString();
     int size=(int)f.length();
     pw.write("Size of the file "+f+" = "+size);
                  if(filename.endsWith(".jpg")||filename.endsWith(".gif"))
      {
      byte [] buffer==new byte [size];
      FileInputStream fis=new FileInputStream(f);
      fis.read(buffer);
      pw.println(buffer);
      }
     else
      {
      FileReader fr = new FileReader(f);
      BufferedReader brd = new BufferedReader(fr);
      while((readfile=brd.readLine())!=null) pw.println(readfile);
      }
    }
    } catch(IOException io) {}
 if(ae.getSource()==lfl)
  {
  try
    {
    JOptionPane.showMessageDialog(null, "" + currmsgl, "Current
Status", JOptionPane.INFORMATION_MESSAGE);
```

```
        UIManager.setLookAndFeel(motifLf);
        SwmgUtilities.updateComponentTreeUI(this);
        } catch(Exception e) {}
     }
   if(ae.getSource()==lf2)
     { try
       {
  JOptionPane.showMessageDialog(null, "" + currmsg2, "Current
  Status", JOptionPane.INFORMATION_MESSAGE);
      UIManager.setLookAndFeel(metalLf);
      SwingUtilities.updateComponentTreeUI(this);
      } catch(Exception e) {}
    }
    if(ae.getSource()==lf3)
     {
      try{
    JOptionPane.showMessageDialog(null, "" + currmsg3, "Current
    Status", JOptionPane.INFORMATION_MESSAGE);
        UIManager.setLookAndFeel(windowsLf);
        SwingUtilities.updateComponentTreeUI(this);
        } catch(Exception e) {}
   }
   if(ae.getSource()==jo 1)
     {
    color=JColorChooser.showDialog(this,"Choose a Color",color);
    if(color==null)
      color=Color.lightGray;
    else
     getContentPane().setBackground(color);
    getContentPane().repaint();
    }
  }
  public boolean handleEvent(Event e)
   {
   if(e.id==Event.WINDOW_DESTROY)
     {
     pw.println(tfn.getText()+" Leaves the Room!");
     System.exit(O);
}
return super.handleEvent(e);}}
```

B. 3 JDBC Connectivity

```
package CHAT;
import javax.swing.*;
import java.sql.*;
import java.lang.*;
import java. sql. Connection;
import java.sql.DriverManager;
import java.sql. Connection;
import java.sql. Statement;
import java.sql. SQLException;
import java.net.*;
import java.security.*;

class jdbcl
{
public String pass=" ";
String ckid=" ";
String node=" ";

public jdbcl (String condition)
{
System.out.println(condition);
String driver = "oracle.jdbc.driver.OracleDriver";
String uri = "jdbc:oracle:thin:@//oradata:1521/oradata";
try
{
   String username == "031275";
String password = "hello 123";
Class. forName(driver);
Connection c = DriverManager.getConnection(url, usemame, password);
    Statement stmt=c.createStatement();
    if(condition.startsWith(" select"))
      { ResultSetrs==stmt.executeQuery(condition);
      if(condition.startsWith("selectNAME"))
  { String profile="";
  while(rs.nextQ) profile="Name= "+rs.getString(l)+"\n"+
          "Address= "+rs.getString(2)+"\n"+
          "City= "+rs.getString(3)+"\n"+
          "nationality= "+rs.getString(4)+"\n"+
```

```
            "Sex== "+rs.getString(5)+"\n"+
            "Email= "+rs.getString(6)+"\n"+
            "Node Address="+rs.getStrmg(7);
      JOptionPane.showMessageDialog(null,profile,"Profile",      JOptionPane.PLAIN_MES
SAGE);
        }
        if(condition.startsWith("select PASSWORD"))
         {
         while(rs.next()) pass=rs.getString(l);
       // return pass;
          System.out.println(pass);
          echopass();
        }
        if(condition.startsWith("selectNODE_ADD"))
         {
         while(rs.next()) node=rs.getString( 1);
         echonodeQ;
       }
       If(condition.startsWith("select CITY"))
           {
            while(rs.next())
              ckid==rs.getString( 1);
           echoid ( );
       }
       } else stmt.executeUpdate(condition);
      } catch(java.lang.Exception ex) {ex.prmtStackTrace();}
    } String echopassQ
 { return pass;
}
String echoid ( )
{return ckid; }
String echonodeQ
{ return node; }
}
```

B.4 Private Chat Server

```
package CHAT;
import java.io.*;
import java.net.*;
import java.awt.*;
import java.awt.event.*;
```

```
public class PrivateChatServer extends Frame implements
ActionListener,Runnable
 {
 TextField tf;
 TextAreata;
String str;
private BufferedReader pbr;
 private PrintWriter ppw;
 String pname;
public PrivateChatServer(String name)
 {
 setLayout(new BorderLayoutQ);
 tf=new TextField();
 add(tf,BorderLayout. SOUTH);
 tf.addActionListener(this);
 ta=new TextArea();
 add(ta,BorderLayout.CENTER);
 ta.setEditable(false); .
 pname=name;

 setSize(400,300);
 setVisible(true);
   try{
ServerSocket pss=new ServerSocket(2000);
    Socket ps=pss.accept();
Pbr=new BufferedReader(new
InputStreamReader(ps.getInputStream()));
   Ppw=newPrintWrite^ps.getOutputStreamO^rue);
   }catch(IOException io) { }
 Thread t = new Thread(this);
 t.start();
 addWindowListener(new Adapt());
}
 public void actionPerformed(ActionEvent ae)
 {
 if(ae.getSource()=='=tf)
  {
  try{
     ppw.println(pname+tf.getText()+"pcb");
```

```
          ta.append(pname+tf.getText()+"\n");
           tf.setText("");
         }catch(Exception e){}
  }
}
public void run()
    {
    try{
      while(true)
        {
       Font f=new Font("SansSerif",Font.BOLD,12);
         ta.setFont(f);
         ta.setForeground(Color.red);
         if((str=pbr.readLine()).endsWith("pcb"))   ta.appendText(str.substring(0,str.length()-
    3)+"\n");
      }
      } catch(IOException io){}
     }

    class Adapt extends WindowAdapter
    {
    public void windowClosing(WindowEvent e)
    {
    try{
         set Visible(false);
        dispose();
        } catch(Exception g) {}}}}
```

B.5 Private Chat Box

```
package CHAT;
import java.net.*;
import java.io.*;
import java.awt.*;
import java.awt.event.*;

public class PrivateChatBox extends Frame implements
ActionListener,Runnable
 {
 TextFieldtf;
```

```
TextArea ta;
String sir;
private BufferedReader pbr;
private PrintWriter ppw;
String pname;
Socket ps;

public PrivateChatBox(String name,String mname)//parameterized
constructor
 {
 pname=name;
 try{
ps= new Socket(InetAddress.getByName(mname),2000);
pbr=new BufferedReader(new
InputStreamReader(ps.getInputStream()));
  ppw=newPrintWriter(ps.getOutputStream(),true);
} catch(IOException io) { }

setLayout(new BorderLayout());
tf=new TextFieldQ;
add(tf,BorderLayout. SOUTH);
tf.addActionListener(this);
ta==new TextArea();
add(ta,BorderLayout.CENTER);
ta.setEditable(false);
Font f=new Font("SansSerif',Font.BOLD, 12);
ta.setFont(f);
ta.setForeground(Color.red);
setSize(400,300);
setVisible(true);

  Thread t = new Thread(this);
  t.startQ;
  addWindowListener(new AdaptQ);
 )
 public void actionPerformed(ActionEvent ae)
  { i f(ae.getS ource()==tf)
   { try{
                        ppw.println(pname+tf.getText()+"pcb");
    ta.append(pname+tf.getText()+"\n");
```

```
        tf.setText("");
        } catch(Exception e) {}
      }
     }
     public void run() //thread is running
      {
      while(true)
        {
        try{
      if((str=pbr.readLine()).endsWith("pcb"))
      ta.append(str.substring(0,str.length0-3)+"\n");
      } catch(Exception e){}
     }
     }
     class Adapt extends WindowAdapter
     { public void windowClosing(WindowEvent e)
( try{
          setVisible(false);
         dispose();
         } catch(Exception g) {}
         }
        } }
```

B.6 Change Password

```
package CHAT;
import java.awt.*;
import j ava. awt. event. *;
import javax. swing.*;

public class chanpassdig extends Dialog implements ActionListener
 {
 TextField tfi;
TextField tfo;
TextField tfn;
Button bo;
Button be;
Button br;
public String getid,getold,getnew;

chanpassdlg(Frame parent,String title)
 {
 super(parent,title);
 setLayout(new FlowLayoutQ);
 setSize(3 00,200);

 Panel p 1 = new PanelQ;
 pl.add(newLabel("EnteryourTD   :"));
 tfi=new TextField(15);
 pl.add(tfi);
 add("North",pl);

 Panel p2 = new Panel ( );
 p2.add(new Label("Old Password :"));
 tfo=new TextField(15);
 p2.add(tfo);
 add("North",p2);
 tfo.setEchoChar('*');

 Panel p3 = new PanelQ;
 p3.add(new Label("New Password :"));
 tfn=new TextField(15);
 p3.add(tfn);
 add("North",p3);
```

```
tfn.setEchoChar('*');


    Panel p4 = new PanelQ;
    bo-new Button("0k");
    p4.add(bo);
    bo.addActionListener(this);
    bc-new Button("Cancel");
    p4.add(bc);
    bc.addActionListener(this);
    br=new Button(" Reset");
    p4.add(br);
    br.addActionListener(this);
    add("North",p4);
  }
  public void actionPerformed(ActionEvent ae)
  {
  if(ae.getSource()==bc)  dispose
();
  if(ae.getSource()==br)
    {
    tfi.setText("");
    tfo.setText("");
    ifn.setText("");
    }
  if (ae.getSource()==bo)
  {
    { getid=tfi.getText();
    getold=fo .getText () ;
    getnew=tfh.getText();

   String chkpass="select password from chatdata where id='"+getid+"'";
    jdbcl db0=newjdbcl(chkpass);
    String gotpassword=db0.echopass();
    if(getold.equals(gotpassword))
        {
        String changepassword="update chatdata set password='"+getnew+"'
   where id='"+getid+"'";
    System.out.println(changepassword);
    jdbcl db=newjdbcl(changepassword);
```

```
      JOptionPane.showMessageDialog(null, "Password Changed
Successfully"
                  ,"Login with new Password"
,JOptionPane.INFORMATION_MESSAGE);
            disposeQ;

          else
            {
       JOptionPane.showMessageDialog(null,"Try Again,Password Not Changed"
                          "Login with old Password next time"
                          ,JOptionPane.WARNING_MESSAGE);
            tfo.setTextC"");
            tfn.setText("");
         }
    }
 }

 }
```

B.7 Fill Form

```
//this is the form which u have to fill up at the time of signup
package CHAT;
import java.io.*;
import java.net.*;
import Java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

class FillForm extends JFrame implements ActionListenerJtemListener
 {
 private JTextField name;
 private JPasswordField pass;
 private JPasswordField cpass;
 private JTextField address;
 private JTextField city;
 private JTextField email;
 private JTextField nationality;
 private JTextField id;
 private ButtonGroup bg;
 private JRadioButton male,female;
 private JButton reset;
 private JButton cancel;
 private JButton submit;
 private JButton delete;
 public String sex="";
 InetAddress add;

FillForm()
 { getContentPane().setLayout(newGridBagLayout());
 GridBagConstraints gbcs=new GridBagConstraintsQ;
 gbcs.fill=GridBagConstraints.HORIZONTAL;

 gbcs.weightx=0;
 add(new JLabel("Enter Name:"),gbcs,0,0,1,l);
 gbcs.weightx=2;
 name=new JTextField(15);
 add(name,gbcs, 1,0,1,1);
add(new JLabel("      "),gbcs,2,1,1,1);
```

```
gbcs.weightx=0;
add(new JLabel("Chat Id:"),gbcs,0,2,1,1);
gbcs.weightx=2;
id=new JTextField( 15);
add(id,gbcs,1,2,1,1);

add(new JLabel("      "),gbcs,2,3,1,1);

gbcs.weightx=0;
add(new JLabel("Enter Password: "),gbcs,0,4,1,1);
gbcs.weightx=2;
pass=new JPasswordField(15);
add(pass,gbcs,1,4,1,1);

gbcs.weightx=0;
reset=new JButton("Reset");
reset.setMnemonic('R');
JPanel jp0=new JPanel();
jpO.add(reset);
addGpO,gbcs,2,5,2,1);
reset.addActionListener(this);

gbcs.weightx=0;
add(new JLabel("Confirm Password: "),gbcs,0,6,1,1);
gbcs.weightx==2;
cpass=new JPasswordField(15);
add(cpass,gbcs, 1,6,1,1);

add(new JLabel("      "),gbcs,2,7,1,1);

gbcs.weightx=0;
add(new JLabel("Permanent Address: "),gbcs,0,8,1,1);
gbcs .weightx==2;
address=new JTextField(15);
add(address,gbcs, 1,8,1,1);

add(new JLabel("      "),gbcs,2,9,1,1);

gbcs.weightx^O;
```

```
add(new JLabel("City:"),gbcs,0,10,1,1);
gbcs.weightx=2;
 city=new JTextField(15);
 add(city,gbcs, 1,10,1,1);

 gbcs.weightx=0;
cancel=new JButton("Cancel");
cancel. setMnemomc('C');
JPanel jpl=new JPanelQ;
jpl.add(cancel);
add(Jpl,gbcs,2,l 1,2,1);
cancel.addActionListener(this);

gbcs.weightx=0;
add(new JLabel("Nationality:"),gbcs,0,12,1,1);
gbcs.weightx=2;
nationality=new JTextField(15);
add(nationality,gbcs, 1,12,1,1);

add(new JLabel("      "),gbcs,2,13,l,l);
gbcs.weightx=0;
male=newJRadioButton("Male",false);
male.setMnemonic('M');
add(male,gbcs, 1,14,1,1);
male.addItemListener(this);

gbcs.weightx=0;
female=newJRadioButton("Female",false);
female. setMnemonic('F');
add(female,gbcs, 1,15,1,1);
female. addItemListener(this);

bg=new ButtonGroupO;
bg.add(male);
bg.add(female);

gbcs.weightx=0;
add(newJLabel("E-Mail:"),gbcs,0,16,l,l);
gbcs.weightx=2;
email=new JTextField(15);
```

```
    add(email,gbcs, 1,16,1,1);
    gbcs.weightx=0;
submit=new JButton("Submit");
submit. setMnemomc('S');
JPanel jp=new JPanelQ;
jp.add(submit);
addGp,gbcs,2,14,2,1);
submit. addActionListener(this);

gbcs.weightx=0;
delete=new JButton("Delete");
delete.setMnemonic('d');
JPanel jp4=new JPanelQ;
jp4.add(delete);
add0p4,gbcs,2,16,2,1);
delete.addActionListener(this);

addWindowListener(new AdaptQ);
}

Private void add(Component c,GridBagConstraints gbc,int x,int y,int w,int h)
{
gbc.gridx=x;
gbc.gridy=y;
gbc.gridwidth=w;
gbc.gridheight=h;
getContentPane().add(c,gbc);
}

public void itemStateChanged(ItemEvent ie)
{
if(ie.getSource()==male)
sex="m";
else sex="f;
},

public void actionPerformed(ActionEvent ae)
{ if(ae.getSource()==submit)
{
                    if(name.getText().length() < 3 || name.getText().length() > 25)
```

```
the maximum limit of your Space Quota","Warning",JOptionPane.WARNING_MESSAGE);
                    else{
                    try{
    add=InetAddress.getLocalHost();
            } catch(UnknownHostException uhe) {}
        String checkid="select city from chatdatabase where
id='"+id.getTextO+"'";
            jdbcl insid=newjdbcl(checkid);
            String oldid=insid.echoid();
            if(oldid.equals(""))
        {
    String ins="insert into chatdata values('" + id.getTextQ +
          "','" + name.getText() + "','" + pass.getText() +
          "','" + address.getTextQ + "','" +city.getText()+
          "','" +nationality.getTextO+"','" +sex+"','" +
    email.getTextO+"','"+add+"')";
    jdbcl insert=new jdbcl (ins);
    setVisible(false);
    disposeQ;
    JOptionPane.showMessageDialog(null,"Your Sign Up Form is
Submitted","Congratulations",JOptionPane.INFORMATION_MESSAG E);
        } else
    JOptionPane.showMessageDialog(null,"Try another CHAT ID as this
already"+"Exits","Wammg",JOptionPane.WARNING_MESSAGE);
      }
    }
    if(ae.getSource()==cancel)
      { setVisible(false);
       dispose( );
}
      if(ae.getSource()==reset)
        { name.setText("">;
        pass.setText("");
        cpass.setText("");
        address.setText("");
        city.setText("");
        email.setText("");
        nationality. setText("");
        id.setText("");
       }
```

```
if(ae.getSource()==delete)
{
if(id.getText().length() < 3 || id.getTextQ.lengthQ > 10)
JOptionPane.showMessageDialog(null,"You must fill Your CHAT ID and it should be
greater then or equal\n"+
"to 3 Characters and must be less then or equal to 10", "Warning",
JOptionPane.WARNING_MESSAGE);

else if(pass.getText().length() < 4 || pass.getText().length() > 8)
JOptionPane.showMessageDialog(null,"You must fill Your PASSWORD and it
should be greater then or equal\n"+
"to 4 Characters and must be less then or
equal to 8","Warning", JOptionPane.WARNING_MES SAGE);

else
JOptionPane.showMessageDialog(null,"Your Information is deleted from
Database","Sorry", JOptionPane.PLAIN_MESSAGE);

String del="delete from chatdatabase where
id="'+id.getTextO+'" ";
jdbcl delete=newjdbcl(del);
id.setText("");
pass.setText("");
}
}
Class Adapt extends window Adapter
<
Public void windowClosing(Window Event e)
{
Try {
SetVisibl (False);
Dispose ():
} catch(exception g) {}
}
}
}
```

B.8 Chat GUI

```
package CHAT;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.util.*;
public class TextAreaDemoM extends JFrame{ private
JTextArea tl;
        public TextAreaDemol4Q
        <
                super("Getting Started");
                Box b = Box.createHorizontalBox();
  String s == "Welcome to the Java Chat Server Help.This Java Chat Server help you to
join the\n"+
                        "chat room for chatting.\n"+
                        "Use the different menu items given to start the service of the Java Chat
Server\n"+
                        "Java Chat Server have many feature like you can chat through it, sign
up etc\n"+
                        "Thankyou for using the Java Chat Server as a chatting program
\n"+
        "using login u can logo onto u can change the password u can do chat u can change
\n"+
        "change your enviomment which helps you to look and feel in a different way\n"-+-
        "you can also sign up for new id and using browse buttin u can send the attachment
\n"+
                        "you can also change the color of your browser\n"+
    "you can also see the profile of the person which are currently present in the chatting
roon.\n"+
        "Apart from allthese u can chat in a better way.\n";
                tl= new JTextArea(s, 10, 15);
                tl.setBackground(Color.cyan);
                tl.setForeground(Color.black);
                tl.setEditable(false);

        b.add(new JScrollPane(tl));
        Container c = getContentPaneQ;
        c.add(b);
        setSize(500, 235);
```

```
            show();

        }

        public static void mam(String args[])
        {
            TextAreaDemoH app = new TextAreaDemo14();
            app. addWindowListener( new WindowAdapter () {
                    public void windowClosing(WindowEvent e)
                {
                        System.exit(0);
            }
        });
      }

}
```

APPENDIX C

Download Manager

```
package LOAD;
import java.io.*;
import java.net.*;
public class Download implements Runnable {

 // Max size of download buffer. public static final int MAX BUFFER_SIZE = 1024;

// These are the status codes. public static final int DOWNLOADING = 0;
      public static final int COMPLETE = 2;
      public static final int ERROR = 4;

      public URL uri; // download URL
      public int size; // size of download in bytes
     public int downloaded; // number of bytes downloaded
      public int status; // current status of download

      // Constructor for Download. public Download(URL url) {
     this.url = uri;
     size = -1;
     downloaded = 0;
     status = DOWNLOADING;
     System.out.println("in constructor \n");
     // Begin the download.
     Download ();
}
private void downloadQ {
         Thread thread = new Thread(this);
         thread. startQ;
     System.out.println("in download function \n");
      }
     // Get file name portion of URL. private String getFileName(URL uri) {
System.out.println("in getfilename function \n");
        String fileName = urI.getFileQ;
     return fileName. substring(fileName.lastIndexOf('/') + 1);
     }
// Download file.
public void run() {
```

```
System.out.println("in run \n");
RandomAccessFile file = null;
InputStream stream = null;

  try{
  System.out.println("in try block of run \n");

// Open connection to URL.
HttpURLConnection con =
    (HttpURLConnection) url.openConnection();
System. out.println("after httpconnection \n");

    // Specify what portion of file to download. con. setRequestProperty( "Range",
"bytes=" + downloaded + "-");
        // Connect to server. con.connectQ;
        System. out.println("connection established \n");
        // Make sure response code is in the 200 range. if (con.getResponseCodeQ / 100
    != 2) { System. out.println("Errorl");
        }

        // Check for valid content length.
        int contentLength = con.getContentLength ( );
        if(contentLength < 1) {
            System.out.println("Error2");
        }

/* Set the size for this download if it hasn't been already set. */          . if
  (size ==-1) {
          size = contentLength;

}

      // Open file and seek to the end of it.
      file = new RandomAccessFile(getFileName(url), "rw");
      file.seek(downloaded);
      System.out.println("back to run after getfilename \n");

      stream = con.getInputStreamQ;
  while (status == DOWNLOADING) { /* Size buffer according to how much of the file is
  left to download. */ byte buffer[];
```

```
            if (size - downloaded > MAX_BUFFER_SIZE) { buffer = new byte
                                        [MAX_BUFFER_SIZE];
        } else {
           buffer = new byte[size - downloaded];

        }
        // Read from server into buffer. int read = stream.read(buffer);
                        if (read == -1)
                            break;

        // Write buffer to file. file.write(buffer, 0, read);
        downloaded = downloaded + read;
        }
        System.out.println("\n status = downloading end");

/* Change status to complete if this point was reached because downloading has
  finished. */ if (status == DOWNLOADING) { status = COMPLETE;

} } catch (Exception e) {
    System.out.println("Error3");
     /* JOptionPane.showMessageDialog( this,
     "Error retrieving specified URL", "Bad URL",
     JOptionPane.ERROR_MESSAGE ); */
} finally {
//Close file.
if (file != null) {


            try{
               file.closeQ;
            } catch (Exception e) {}
          )

          // Close connection to server. if (stream !=null) {
            try{
               stream.close();
          } catch (Exception e) {}
    } }
 }
 // Notify observers that this downlead's status has changed.
 }
```

PERSONAL DETAILS

Name – Kunal Jain

Enrollment number – 071309

Course - B.tech (CSE)

Semester – 8th

Batch – 2007-2011

Contact number: +91-9736252395

Email – kunaljain1701@gmail.com

Contact Address: – 20/1, Mohini road, Dalanwala, Dehradun-248001

PERSONAL DETAILS


No.17043

Name – Manav Bhasin

Enrollment number – 071344

Course - B.tech (CSE)

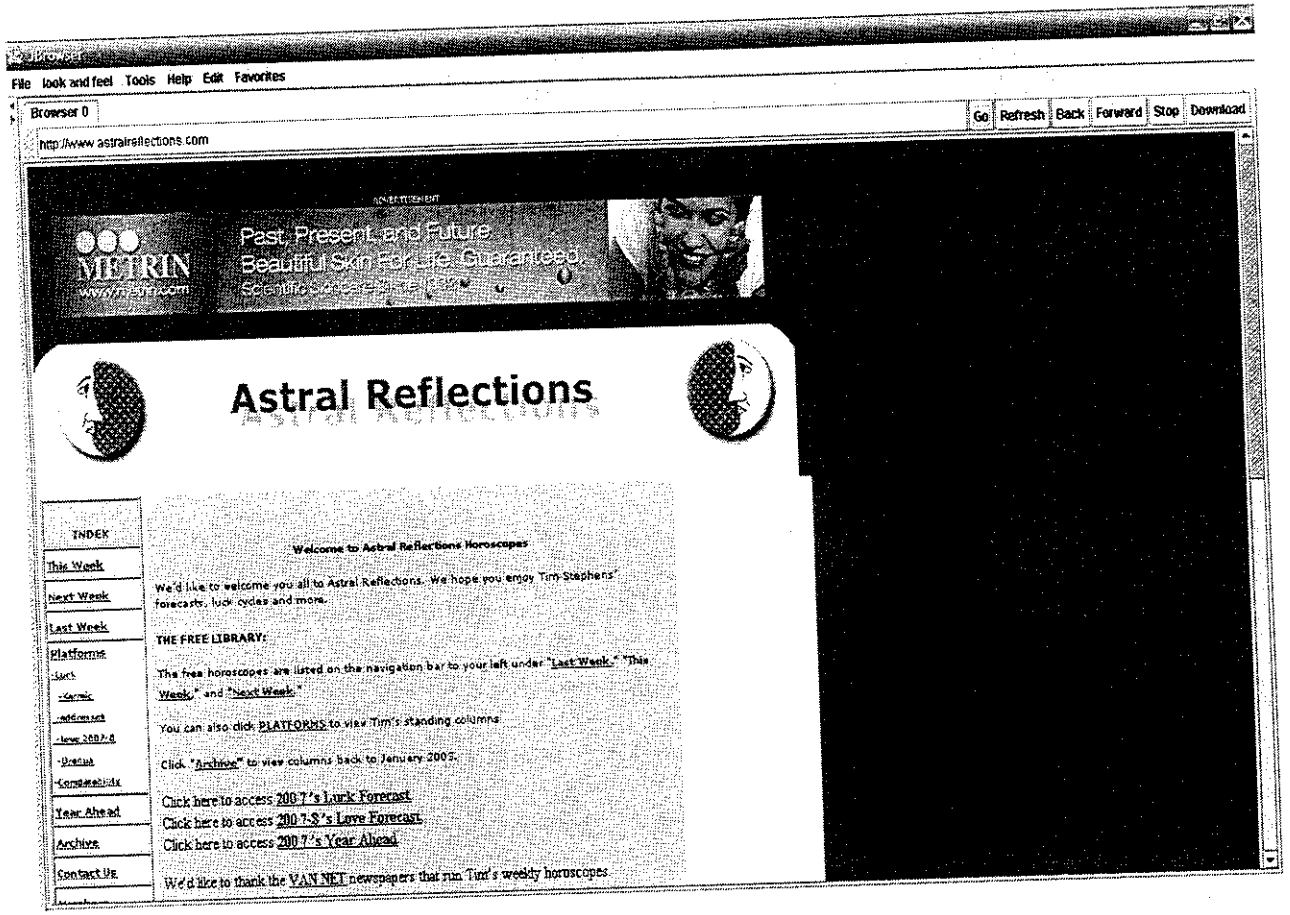
Semester – 8th

Batch – 2007-2011

Contact number: +91-9816319047

Email – manavbhasin11@gmail.com

Contact Address: –585, sector #19, Faridabad-121001


Browser
File look and feel Tools Help Edit Favorites
Browser 0
Go Refresh Back Forward Stop Download
http://www.astralreflections.com
METRIN
Past, Present and Future
Beautiful Skin For Life, Guaranteed.
Astral Reflections
INDEX
This Week
Next Week
Last Week
Platforms
Archive
Contact Us
Welcome to Astral Reflections Horoscopes
We'd like to welcome you all to Astral Reflections. We hope you enjoy Tim Stephens' forecasts, luck cycles and more.
THE FREE LIBRARY:
The free horoscopes are listed on the navigation bar to your left under "Last Week," "This Week," and "Next Week."
You can also click PLATFORMS to view Tim's standing columns.
Click "Archive" to view columns back to January 2005.
Click here to access 2007's Luck Forecast
Click here to access 2007's Love Forecast
Click here to access 2007's Year Ahead
We'd like to thank the VAN NET newspapers that run Tim's weekly horoscopes.