

*Implementation of Discussion Forum
With AdServer*

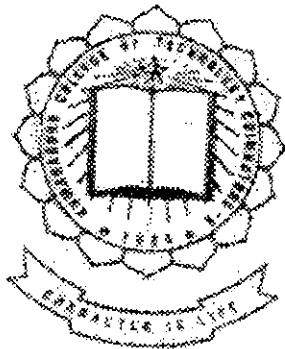
P-508

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR
THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

OF BHARATHIAR UNIVERSITY,
COIMBATORE.



Submitted By

**Vidyalakshmi.R
Mahalakshmi.R**

Guided By

Ms.N.Rajathi, B.E.

Department of Computer Science and Engineering
KUMARAGURU COLLEGE OF TECHNOLOGY

Coimbatore-641006

March 2001

KUMARAGURU COLLEGE OF TECHNOLOGY
(Affiliated to Bharathiar University)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Project Report entitled

*Implementation of Discussion forum
With AdServer*

is a bonafide record of work done by

Ms. R. VIDYALAKSHMI, R. MAHALAKSHMI.....

in partial fulfillment of the requirements for the
award of Degree of

BACHELOR OF ENGINEERING in
Computer Science and Engineering

N. Rajatni
12/3/2001

Guide

S. J. Srinivasan 10/3/01
Head Of Department

Submitted for the University Examination held on 12-03-2001 2001

University Register Number : 9727K0188, 9727K0153

S. Srinivasan
Internal Examiner

External Examiner

TO WHOM SO EVER IT MAY CONCERN

This is to certify that Ms. Vidyalakshmi.R and Ms. Mahalakshmi.R , final year students of Computer Science and Engineering , Kumaraguru College of Technology, Coimbatore have completed a project on IMPLEMENTATION OF DISCUSSION FORUM WITH ADSERVER as a partial fulfillment of the requirements for the Bachelor's Degree of Computer Science and Engineering in Java .

FOR TULEC


Project Co-ordinator


CENTRE HEAD

DECLARATION

We hereby declare that this project work entitled

Implementation of Discussion Forum with AdServer

submitted to Kumarguru College of Technology, Coimbatore (Affiliated to Bharathiar University) is a record of original work done by us under the supervision and guidance of Ms. N.Rajathi, Lecturer, Department of Computer Science, Kumarguru College of Technology, Coimbatore.

Place : Coimbatore

Date : 08-03-2001

Signature of Candidates

R. Vidyalakshmi

R. Mahalakshmi

ACKNOWLEDGEMENT

An endeavor over a long period can be successful only with the advice and support of many well wishers . We take this opportunity to express our gratitude and appreciation to all of them.

We are extremely grateful to **Dr. K.K.Padmanabhan**, Principal, Kumaraguru College of Technology, for having given us the golden opportunity to serve the purpose of our education.

We are bound to express our gratitude to **Prof. S.Thangasamy**, Head of the Department of Computer Science and Engineering, Kumaraguru College of Technology, for his constant encouragement throughout the course.

We wish to thank our Internal Guide, Lecturer. **N.Rajathi**, Department of Computer Science and Engineering, Kumaraguru College of Technology, for constantly encouraging us to pursue new goals, ideas and being supportive throughout the tenure of our project.

It is our privilege to express our sincere thanks to our guide **Mr.Andyappan** of **TULEC COMPUTER EDUCATION** for his meticulous guidance ,valuable suggestions and timely help offered to us during the tenure of our project period.

We wish to place our unsolicited gratitude and respect to **Ms.Lavanya**, Class Advisor, Department of Computer Science and Engineering, Kumaraguru College of Technology.

We take pleasant privilege in expressing our heartfelt thanks to all the staff members of the Computer Science and Engineering Department for all the help and support extended by them.

SYNOPSIS

The objective of the project is to design and implement a Discussion Forum with an AdServer. Publicity is the buzzword for the progress of any firm in today's competitive world. The Internet provides a large arena which promises competence. The AdServer is one such component which serves as a handy and an extremely effective tool for businesses, both small and large. The automation of such a tool facilitates prompt service and renewal of subscription details.

Yet another tool that serves to host people's ideas and views is the discussion forum. It serves as a platform to bring netizens closer, beyond spatial and temporal boundaries.

The system is developed using Java with MS Access as the back end tool. This project is the result of our exploration of the multitude facets of Java RMI and applets.

CONTENTS

1. Introduction
2. System Environment
3. Software Requirements
4. Overview of the language
5. System Design and Development
6. System Security
7. Conclusion
8. Scope for further development
9. Bibliography
10. Appendix
 - A. Sample Source Code
 - B. Sample Screen Formats

ABOUT THE ORGANIZATION

TULEC - A Perspective

TULEC Computer Services is an entity made out of their parent company M/s. Red Rose construction Limited, Coimbatore. They started their company as a software group in 1994 and the growth achieved has propelled them to start it as an export outfit. Their aggressiveness and dedication started off paying dividends from the year 1995 - 96 and the turnover has been showing increase after that.

They have set their aims and policies clear and every activity is viewed from an angle of customer satisfaction, human resources and manpower management. The company is also known for their continuous programs which encourages their employees.

They are in search of new blood and ideas to infuse into their team of high caliber in experienced professionals.

SERVICES

They concentrate primarily on the small and medium sized industry needs. Here they are a comprehensive integrated manufacturing application suite and make sure their software is able to understand the dynamics of specific business, can bring out / re-engineer the business. Every effort is taken care that there is a scalability of vertical integration of design so that the customization difficulties do not arise. The product "PRO - B" is capable of handling the need of OEM's in their respective segments.

TULEC Computer Services has been setup as a complete self-contained unit, providing technical domain knowledge in ONLINE, INTERNET APPLICATION which spans across markets and customer position to leverage their skills in web enabling existing operations, and they have solution that targets vertical segments of Industry mostly in manufacturing & retailing. Their range

covers widely on development of web based tools and product implementation, databases and consulting end to end for business application. It may be customer to customer, business to business or secured net based transactions.

INFRASTRUCTURE

Their infrastructure consists of marble topped floor of 3000 sq.ft under centralized AC. Necessitated things are layered out in an appropriate order manner such that the ergonomics facilitates a better turn. The present infrastructure includes an array of latest machines and the latest software in original. The organization is formed as Board of Directors --> Vice Chairman --> Consultant --> Project Leaders --> Programmers.

They also have a Medical Transcription lab which is having facility of 25 working in 3 shifts where in they are in a position to transcribe 8500 line to 10,000 lines per day. They have already initiated the process of manpower planning, recruiting and training. They have decided to go in for a leased line which will help them under clients for transferring information in an efficient manner. The data transfer is done at a faster rate.

SYSTEM ENVIRONMENT

SOFTWARE REQUIREMENTS:

Platform	Java 1.1.6
Operating System	Windows
Web Server	Microsoft Personal Web Server
Browser	Netscape Navigator

HARDWARE REQUIREMENTS:

Processor type	Pentium Processor
Speed	166 MHz
Main memory	32 MB RAM
Hard Disk	1.2 GB
Drive	1.44 MB
Visual Display	VGA Color Monitor
Keyboard	TVS 104 keys
Printer	HP Laser Printer

SOFTWARE REQUIREMENTS

1. INTRODUCTION

The project is a self contained, internal network , linking multiple users by means of Internet technology. This project encompasses two major modules : a simple AdServer with automated mailing list manager and a discussion forum. This specification provides the overall requirements of the software.

1.1. PURPOSE

This chapter deals with the general description of the software, functional requirements , performance requirements and design constructs. It is this document that acts as a major reference during further modifications or future enhancement of the software.

1.2. SCOPE

The requirements imposed for the project by the client necessitates the software to contain the following:

The AdServer module:

This module combines the B2B and B2C features of E-Commerce wherein the client , which is generally a business firm, specifies necessary details regarding its advertisement. Once the payment is made, the client's account will be activated and the advertisement will be displayed on the respective site and will flash for the specified duration. A hyperlink will be provided to the client's home page. This forms the B2C part of the plan.

Automated Mailing List Manager:

As the name suggests , this module is automated , in the sense, the mailer program keeps track of the number of days remaining for the advertisement to remain active on the site. When

the subscription is nearing exhaustion , this module automatically sends a mail to the respective subscribers requesting for either renewal of service or cancellation.

Discussion Forum:

This simple yet effective module helps users view and participate in various ongoing discussions . The user can either initiate a topic for discussion or merely post comments on ongoing topics. The user can also view comments posted by others with the aid of the MyBox feature.

2. GENERAL DESCRIPTION

2.1. PRODUCT PERSPECTIVE:

The perspective of the product is as follows:

- Very user friendly
- Adaptable to many platforms
- Easy to register and advertise
- Immediate Updates
- Flexible modes of payment
- Less paper work
- Convenient way of exchanging ideas

2.2 PRODUCT FUNCTIONS:

AdServer module:

This module comprises of two concepts-a B2B component wherein companies can register by specifying necessary such as the number of minutes, months of display, format of the advertisement and essential URLs to link their web site.

In the B2C component, the advertisements are posted on the requested website for the specified duration of time with suitable links provided.

Automated Mailing List Manager:

Here, the automatic mailing system keeps track of the time duration specified and in case the date nears exhaustion, the mailing list manager automatically sends a mail to corresponding clients requesting them to renew their subscriptions.

Discussion Forum:

This module helps to strengthen the environmental practice of any firm wherein easy exchange ideas and comments is possible. The user is provided with options - to view, to initiate a topic for discussion or post comments on ongoing discussions. The user is provided with an inbox feature where he can view comments addressed to him posted by other users.

2.3. USER CHARACTERISTICS:

This site is for all kinds of users. People who have little or no prior knowledge about computers can use this software as it is very user friendly. They can have their company logos advertised by just entering a few details and with the help of a few clicks. The discussion forum is another elegant module wherein a novice user can share his views on various topics.

2.4. GENERAL CONSTRAINTS:

- The advertisement will be valid only for the specified duration.
- Only registered users can participate in discussion forums.

3. SPECIFIC REQUIREMENTS

3.1. FUNCTIONAL REQUIREMENTS:

Introduction:

Functional Requirements specify the relationships among the inputs, actions and outputs. Here, inputs are got from firms, processed and corresponding output actions delivered to end users.

List of inputs:

Some sample inputs obtained in this software are :

- Specifying the time limit for the advertisement to be active.
- Getting format type of the advertising link
- Path for the URL link
- Registered users can either post comments or merely view them

Information processing required:

- Storing information regarding customers for advertisement section.
- Produce invoice once the firm registers for advertisement display.
- Keep track of time and remind clients of subscription cancellation or renewal.
- Updation of contents in the MyBox feature and the VIEW topics form.

List of outputs:

- An invoice regarding chosen options is displayed based on the selection made by the user.
- Once the user gets registered, the advertisements are displayed on the web site.
- The user can view replies and comments posted to him by others based on relevant topics.

3.2. PERFORMANCE REQUIREMENTS:

3.2.1. SECURITY:

In this software security is given prime importance. The user cannot modify the existing information. Only the administrator can alter the information that is stored in the database.

3.2.2 AVAILABILITY:

The software is made available any user irrespective of location, time and other factors.

3.2.3. CAPACITY:

The product is designed with web based technology so that it can be accessed from different platforms and can accommodate any number of users.

3.2.4. RESPONSE TIME:

The response time for various activities of the software depends upon the configuration of the client machine and as well as the network performance.

3.3. DESIGN CONSTRAINTS

Guidelines are specifications that initiate the proper construction of the product. There are many guidelines for requirements, documents, design specifications, structured coding conventions, documents for testing, installation manual and user manual contribute to the understandability and hence the maintainability of the software. Some of the guidelines complied are:

3.3.1. CODING GUIDELINES

- Goto statements are not used.
- The nesting depth of the program constructs do not exceed 30 lines.
- Long obscure comments are avoided.
- Descriptive names from the problem domains for user defined data types, variables, formal parameters etc are being used for providing proper comments.

3.3.2. HARDWARE LIMITATIONS:

The software developed is expected to work only under the following minimum specifications:

Processor : Intel 486 based PC

Primary memory : 32 MB RAM

Secondary memory : 1.2 GB

3.3.3. EXTERNAL INTERFACE REQUIREMENTS :

User interfaces :

The user is provided with friendly and legible interfaces. There are different interfaces used in the software , which are used for various purposes . There are different interface styles for communication , for registration for advertisements, facilities for the user to view and actively participate in discussions, automated reminder

Hardware interfaces:

Certain hardware interfaces are to be used in order to make the software available in the web environment . The application server processes the request from the user and directs the response to the requester .

All the above specified requirements are agreed by both the developers and the customers and will be brought into notification whenever modifications or enhancements are needed. The developers are required to take the client on a comprehensive tour of the system and ensure that all agreed specifications are met.



1778

OVERVIEW OF THE JAVA LANGUAGE

Java is related to C++ which is a direct descendent of C. Much of the characteristics of Java are inherited from these two languages. Java derives its syntax from C. Most of Java's object oriented features were influenced by C++. Moreover, the creation of Java was deeply rooted in the process of refinement and adaptation that has been occurring in computer programming languages for the past three decades. For these reasons, this section reviews the sequence of events and forces that led to Java. As we will see, each innovation in language design was driven by the need to solve a fundamental problem that the preceding languages couldn't.

THE CREATION OF JAVA

Java was conceived by JAMES GOSLING, PATRIC NAUGHTON, CHRIS WARTH, ED FRANK and MIKE SHERIDAN at Sun Microsystems, Inc. in 1991. It took 18 months to develop the first working version. This language was initially called OAK but was renamed JAVA in 1995. Between the initial implementation of OAK in the fall of 1992 and the public announcement of Java in the spring of 1995, many more people contributed to the design and evaluation of the language.

Somewhat surprisingly, the original impetus for Java was not the Internet. Instead, the primary motivation was the need for the platform independent language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls. As you can probably guess, many different types of CPUs are used as controllers. The trouble with C and C++ is that they are designed to comply a specific target. The problem is that the compilers are expensive and time consuming to create.

REMOTE METHOD INVOCATION

INTRODUCTION:

A distributed object model for Java to Java applications, in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts.

RMI enables the programmer to create distributed Java-to-Java applications, in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts. A Java program can make a call on a remote object once it obtains a reference to the remote object, either by looking up the remote object in the bootstrap-naming service provided by RMI, or by receiving the reference as an argument or a return value. A client can call a remote object in a server, and that server can also be a client of other remote objects. RMI uses Object Serialization to marshal and unmarshal parameters and does not truncate types, supporting true object-oriented polymorphism.

OVERVIEW :

The RMI system consists of three layers: the stub/skeleton layer, the remote reference layer, and the transport layer. The boundary at each layer is defined by a specific interface and protocol; each layer, therefore, is independent of the next and can be replaced by an alternate implementation without affecting the other layers in the system. To accomplish transparent transmission of objects from one address space to another, the technique of object serialization (designed specifically for the Java language) is used. Another technique, called dynamic stub loading, is used to support client-side stubs which implement the same set of remote interfaces as a remote object itself. This technique, used when a stub of the exact type is not already available to the client, allows a client to use the Java language's built-in operators for casting and type-checking.

NEED FOR RMI

In order to match the semantics of object invocation, distributed object systems require remote method invocation or RMI. In such systems, a local surrogate(stub) object manages the invocation on a remote object.

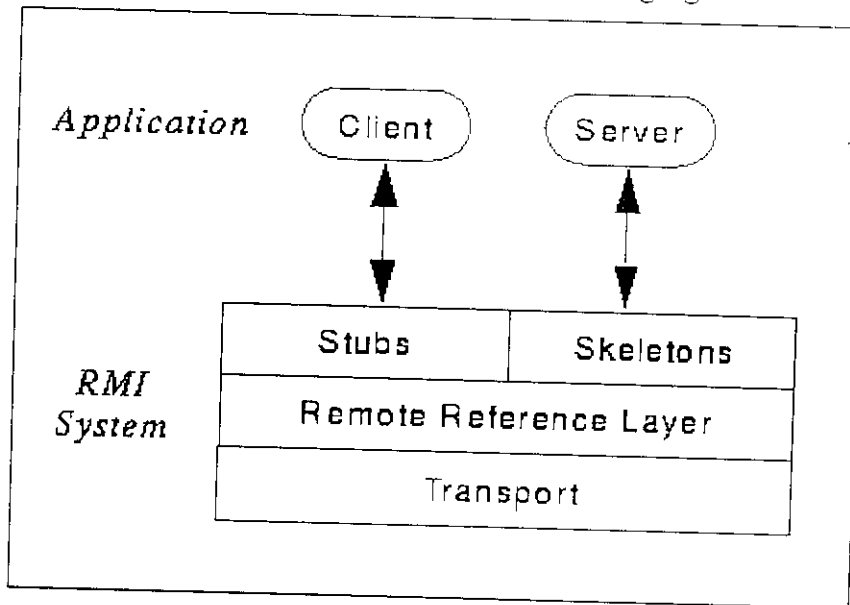
The Java remote method invocation system has been specifically designed to operate in the Java environment. While other RMI systems can be adapted to handle Java objects, these systems fall short of seamless integration with the Java system due to their interoperability requirement with other languages. For example, CORBA presumes a heterogeneous, multilanguage environment and thus must have a language - neutral object model. In contrast, the Java language's RMI system assumes the homogeneous environment of the Java.

ARCHITECTURAL OVERVIEW

The RMI system consists of three layers :

- The stub / skeleton layer – client – side stubs (proxies) and server-side skeletons.
- The remote reference layer - remote reference behavior (such as invocation to a single object or to a replicated object).
- The transport layer - connection set up and management and remote object tracking .

The relationship between the layers is shown in the following figure



A remote method invocation from a client to a server object travels down through the layers of the RMI system to the client-side transport, then up through the server-side transport to the server.

A client invoking a method on a remote server object actually makes use of a stub or proxy for the remote object as a conduit to the remote object. A client-held reference to a remote object is a reference to a local stub. This stub is an implementation of the remote interfaces of the remote object and forwards invocation requests to that server object via the remote reference layer. Stubs are generated using the `rmic` compiler.

The remote reference layer is responsible for carrying out the semantics of the invocation. For example, the remote reference layer is responsible for determining whether the server is a single object or is a replicated object requiring communications with multiple locations. Each remote object implementation chooses its own remote reference semantics-whether the server is a single object or is a replicated object requiring communications with its replicas.

Also handled by the remote reference layer are the reference semantics for the server. The remote reference layer, for example, abstracts the different ways of referring to objects that are implemented in

- (a) Servers that are always running on some machine
- (b) Servers that are run only when some method invocation is made on them (activation).

At the layers above the remote reference layer, these differences are not seen. The transport layer is responsible for connection setup, connection management, and keeping track of and dispatching to remote objects (the targets of remote calls) residing in the transport's address space.

In order to dispatch to a remote object, the transport forwards the remote call up to the remote reference layer. The remote reference layer handles any server-side behavior that needs to occur before handing off the request to the server-side skeleton. The skeleton for a remote object makes an up call to the remote object implementation which carries out the actual method call. The return value of a call is sent back through the skeleton, remote reference layer, and transport on the server side, and then up through the transport, remote reference layer, and stub on the client side.

In RPC (remote procedure call) systems, client-side stub code must be generated and linked into a client before a remote procedure call can be done. This code can be either statically linked into the client or linked in at runtime via dynamic linking with libraries available locally or over a network file system. In the case of either static or dynamic linking, the specific code to handle an RPC must be available to the client machine in compiled form. RMI generalizes this technique, using a mechanism called dynamic class loading to load at runtime (in the Java language's architecture neutral bytecode format) the classes required to handle method invocations on a remote object.

These classes are:

- The classes of remote objects and their interfaces.
- The stub and skeleton classes that serve as proxies for remote objects.
(Stubs and skeletons are created using the `rmic` compiler).

Other classes used directly in an RMI-based application, such as parameters to, or return values from, remote method invocations.

APPLETS

A program written in Java to run within a Java-compatible web browser, such as HotJava or Netscape Navigator. The applet API lets you take advantage of the close relationship that applets have with Web browsers. The API is provided by the `java.applet` package -- mainly by the `Applet` class and the `AppletContext` interface. Applets can do the following:

- Be notified by the browser of milestones.
- Load data files specified relative to the URL of the applet or the page in which it is running.
- Display short status strings.
- Make the browser display a document.
- Find other applets running in the same page.
- Play sounds.
- Get parameters specified by the user in the `<APPLET>` tag.

TABLE STRUCTURE

Table –1

Name : Users

Description : This table maintains details of registered clients.
The username and password are the contents of this table.

Field Name	Type	Size	Description
username	Text	50	name of user
password	Text	50	password

Table - 2

Name : Signup

Description : This table stores details of new users who register in the firm to post their advertisements on sites.

Field Name	Type	Size	Description
username	text	50	new username
passwd	text	50	password
email	text	50	mail id of user

Table – 3

Name : Register

Description : This table stores all details of users who register in the firm for their advertisements to be displayed on web sites.

Field Name	Type	Size	Description
minutes	number	10	time in minutes
month	number	10	duration
combo	text	50	format type
caption	text	50	company logo
url	text	50	hyperlink
path	text	50	path of image
name	text	50	name of firm
company	text	50	company
email	text	50	mail id
datereg	text	10	date of registration
dateexp	text	10	expiry date

Table – 4

Name : Invoice

Description: This table stores the unit price details of formats available in the site for registration.

Field Name	Type	Size	Description
type	text	50	format type
unitprice	text	50	unit price

Table –5

Name : Query

Description : This table stores details of queries posted by users who wish to initiate a topic for discussion.

Field Name	Type	Size	Description
Username	Text	20	name of user
Topic	Text	20	discussion topic
Question	Text	100	query posted
Datasent	Text	100	comment

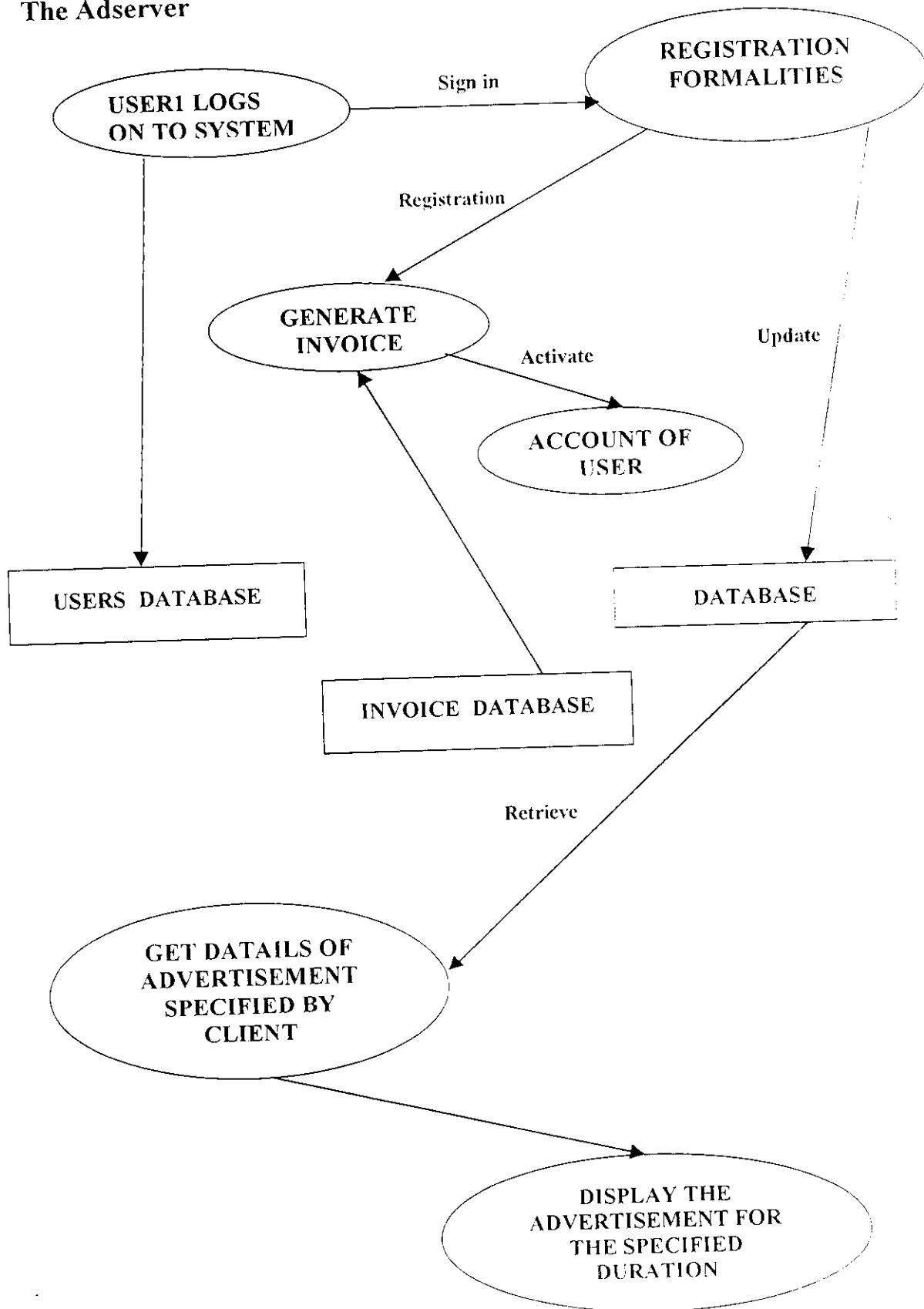
Table – 6

Name : Reply

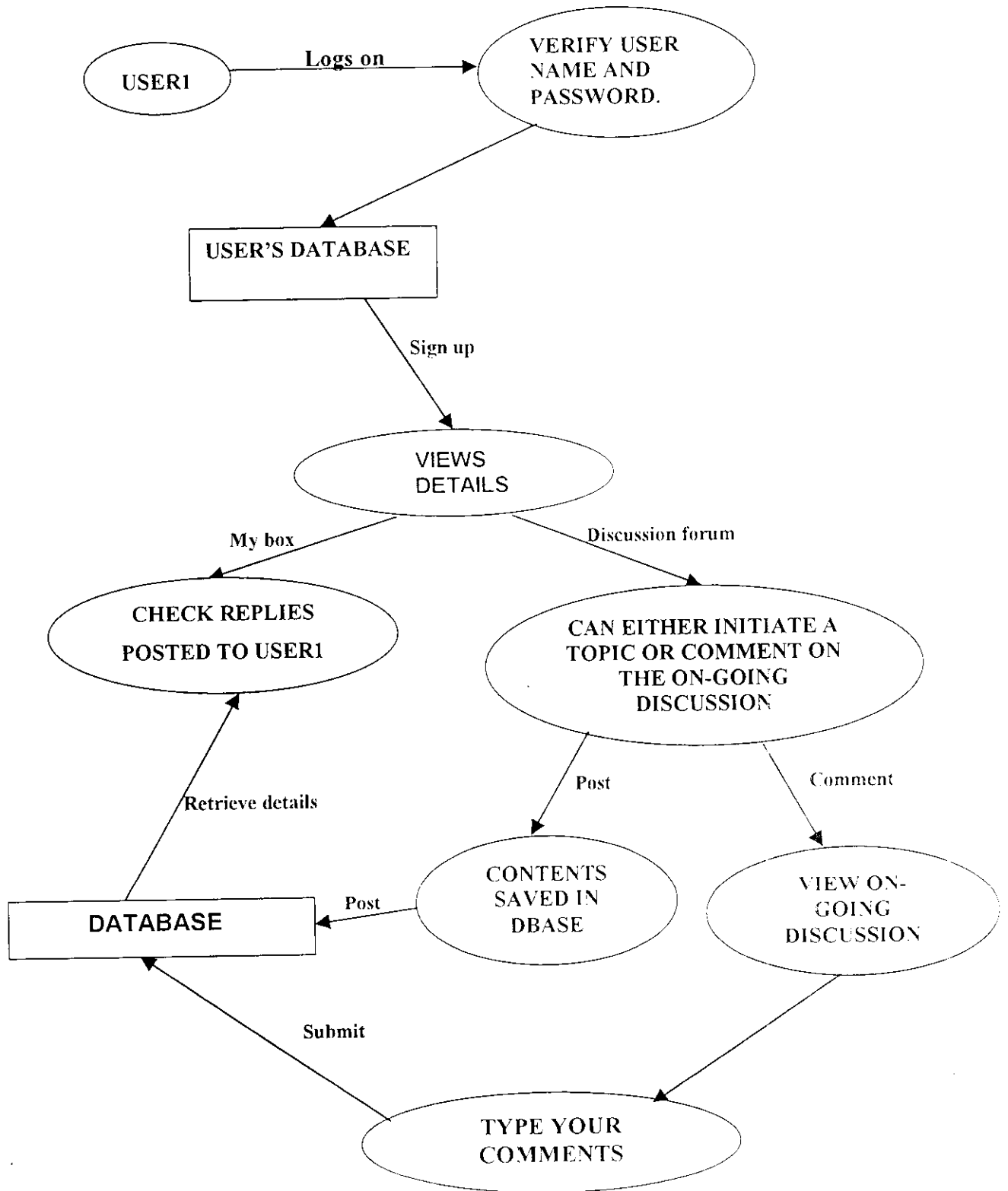
Description : This table stores the comments and queries posted to ongoing discussions by all users.

Field Name	Type	Size	Description
From1	Text	20	Posted by
To	Text	20	in reply to
Topic	Text	30	topic posted
Question	Text	100	query
Answer	Text	100	comments
Daterep	Text	50	date posted

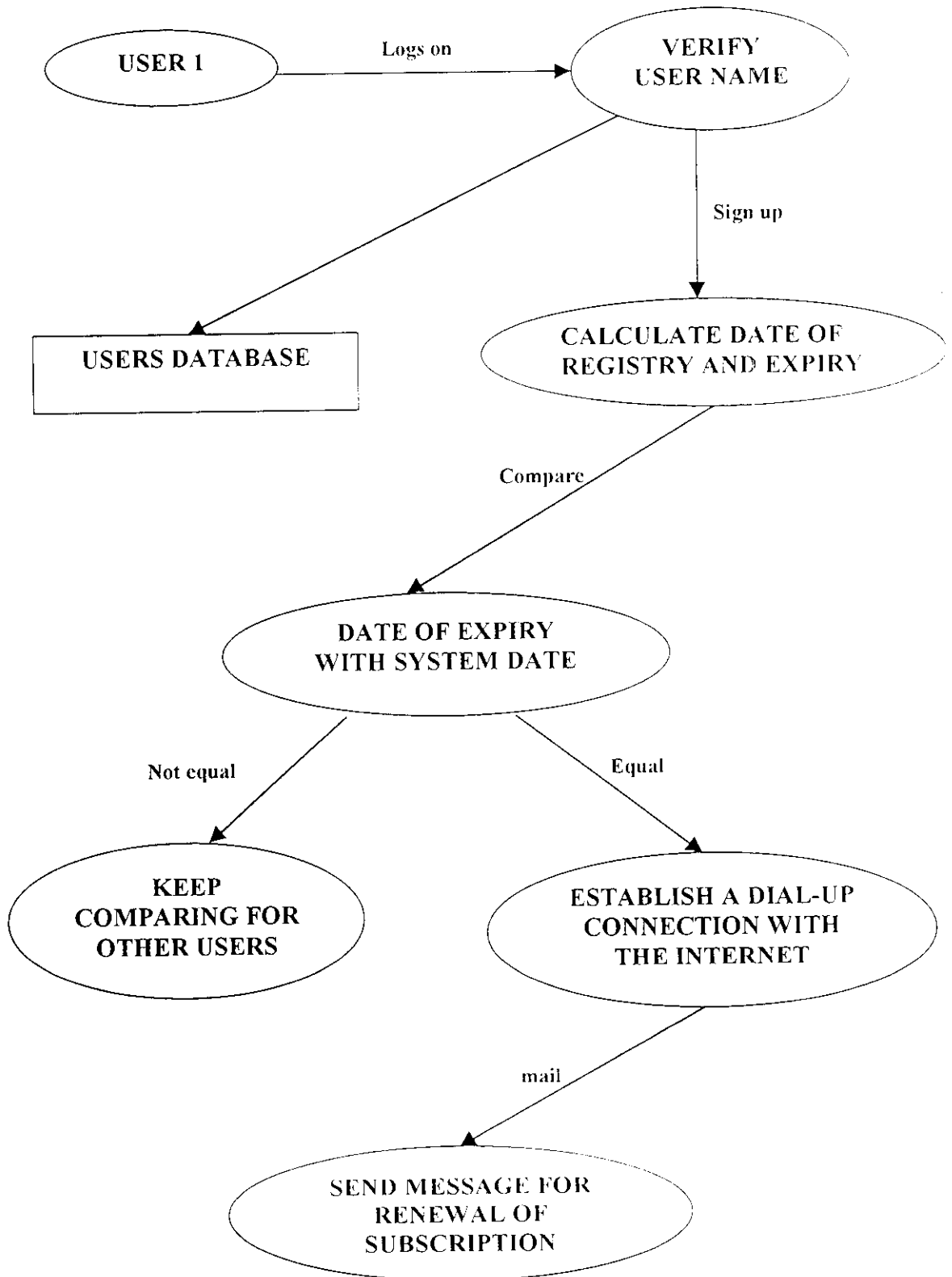
The Adserver



The Discussion Forum



The automated mailing list manager



SYSTEM SECURITY

Security is a critical stage in system development. Every candidate system must provide features for security and integrity of data. Without safeguards against unauthorized access, fraud, embezzlement, fire and natural disasters, a system could be so vulnerable as to threaten the survival of the organization.

To do an adequate job on security, the risks, exposure, costs and specific measures such as passwords should be analyzed to provide protection. In addition, backup copies of software and recovery restart procedures must be available when needed.

Threats to system security

The list of potential threats are:

- Errors and omissions
- Disgruntled and dishonest employees
- Fire
- Natural calamities and disasters
- External attack

System security measures

After system security risks have been evaluated, the next step is to select security measures.

The measures are:

- IDENTIFICATION
- ACCESS CONTROL
- AUDIT CONTROL
- SYSTEM SECURITY

IDENTIFICATION

It is a scheme for identifying persons to the systems based on "Something You Know" such as a password or a particular badge, "Something You Are" such as a finger print or voice print or "Something You Have" such as a credit card, key or special terminal.

ACCESS CONTROL

Controlling access to the computer facility is secured through encoded cards or similar devices. Encryption prevents intruders from accessing data by scrambling message across telephones or their destination.

AUDIT CONTROLS

Auditing must be supported at all levels of management. Audit controls protect a system from external security breaches and internal fraud or embezzlement. Various software programs are available to help in the audit function.

SYSTEM INTEGRITY

This line of defense safeguards the functioning of hardware, software, physical security and operating procedures. Proper backup of hardware and software are extremely important.

CONCLUSION

The system development life cycle comes to a completion with the final handover of the system. The system objectives underlined during the start of the system life cycle are all fully met with. The system tries to implement most of the suggestions given by the users during the system study. The others, which were not met implemented, due to conflicts, were left for future enhancements for the system. All the options were weighed upon and the best possible were taken into consideration and implemented.

The system development was a wonderful experience and it has given us a deep insight into real time development. It has also allowed us to uncover the advanced programming concepts of Java RMI and applets. Maximum justification has been done to each of the module in the allotted time span.

SCOPE FOR FURTHER DEVELOPMENT

The database used in this project can be developed using ORACLE in order to enhance speed when hosted on the net.

A variety of other topics can be included in the discussion forum thereby providing a wider base for exchange of ideas for users.

Facilities to ease renewal of subscriptions by users can be provided. The security of the system can be further enhanced.

BIBLIOGRAPHY

Patrick Naughton, Herbert Schildt , *The Complete Reference Java 2*, Founding member of the Original Java Project Team at Sun Microsystems, Tata McGraw Hill Publishing Company Ltd , Third Edition

Cay S, Horstman., *Computing concepts with java essentials*, John WileyandSons. NY. , 1998

Roger S. Pressman., *Software Engineering - A practitioners approach*, Mc Graw Hill, 1997, Fourth Edition

A. SAMPLE CODING

Server.java

```

package pack;
import java.rmi.server.*;
import java.rmi.dgc.*;
import java.rmi.registry.*;
import java.io.*;
import java.rmi.*;
import java.sql.*;

public class ser extends UnicastRemoteObject implements chatserver
{
    public ser() throws RemoteException
    {
    }
    public int dis(String str,String str1)
    {
        int j=2;

        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select count(*) from users where
            user='"+str+"'and password='"+str1+"'");
            if(rs.next())
                j= rs.getInt(1);
        }
        catch(Exception e)
        {
            System.out.println("the error is "+e);
        }
        return(j);
    }
    public int booking(String min,String mon,String combo,String caps,String usl,String
    pat,String name,String comp,String addr,String mail)
    {
        int h=2;
    }
}

```

```
try
{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
    PreparedStatement st = con.prepareStatement("insert into register
values(?,?,?,?,?,?,?,?,?)");

    st.setString(1,min);
    st.setString(2,mon);
    st.setString(3,combo);
    st.setString(4,caps);
    st.setString(5,usl);
    st.setString(6,pat);
    st.setString(7,name);
    st.setString(8,comp);
    st.setString(9,addr);
    st.setString(10,mail);
    int r = st.executeUpdate();
}
catch(Exception e)
{
    System.out.println("the error1 is "+e);
}
return(h);
}

public int store(String st1,String st2)
{
    int i=2;
    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
        PreparedStatement st = con.prepareStatement("insert into users values(?,?)");
        st.setString(1,st1);
        st.setString(2,st2);
        int t = st.executeUpdate();
    }
    catch(Exception e)
    {
        System.out.println("the error2 is "+e);
    }
    return(i);
}
```

```
public String[] recall()
{
    String ring[]=null ;

    int i=0;

    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
        Statement st = con.createStatement();
        ResultSet rs1 = st.executeQuery("select count(*) from register");
        rs1.next();
        ring = new String[rs1.getInt(1)];
        ResultSet rs = st.executeQuery("select
        minutes,month,combo,caption,name,comp,address,email from register");
        while(rs.next())
        {
            ring[i] =
            rs.getString(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" "+rs
            .getString(5)+" "+rs.getString(6)+" "+rs.getString(7)+" "+rs.getString(8)+" ";
            i++;
        }

    }
    catch(Exception e)
    {
        System.out.println("the error in invoice is "+e);
    }
    return(ring);
}

public String[][] check(String m1,String m2,String com)
{
    String str[][] = new String[10][2];
    int i=0;

    try
    { Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
      Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
      Statement st = con.createStatement();
      ResultSet rs0 = st.executeQuery("select * from invoice");
      while (rs0.next())
      {
```



```
        str[i][0]= rs0.getString(1);
        str[i][1]= rs0.getString(2);
        i++;
    }
}

catch(Exception e)
{
    System.out.println("the error3 is "+e);
}
return(str);
}
```

//DISCUSSION FORUM DATABASES

//COMPARING USER NAME PASSWORD

```
public int next(String str,String str1)
{
    int j=2;

    try
    { Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
      Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("select count(*) from signup where
        username='"+str+"'and password='"+str1+"'");
      if(rs.next())
      j= rs.getInt(1);
    }
    catch(Exception e)
    {
        System.out.println("the error in next is "+e);
    }
    return(j);
}
```

//STORING NEW USERS DETAILS

```
public int store1(String st1,String st2,String st3)
{
    int i=2;
```

```
try
{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
    PreparedStatement st = con.prepareStatement("insert into signup
        values(?,?,?)");
    st.setString(1,st1);
    st.setString(2,st2);
    st.setString(3,st3);
    int t = st.executeUpdate();
}
catch(Exception e)
{
    System.out.println("the error in store1 is "+e);
}
return(i);
}
```

//STORING THE QUERIES IN THE DATABASE

```
public int save(String st1,String st2,String st3,String date)
{
    int h=2;
    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
        PreparedStatement st = con.prepareStatement("insert into query
            values(?,?,?,?)");
        st.setString(1,st1);
        st.setString(2,st2);
        st.setString(3,st3);
        st.setString(4,date);
        int r = st.executeUpdate();
    }
    catch(Exception e)
    {
        System.out.println("the error in save is "+e);
    }

    public String[] return1()
    {
        String ring[]=null ;
        int i=0;
```

```
try
{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
    Statement st = con.createStatement();
    ResultSet rs1 = st.executeQuery("select count(*) from query");
    rs1.next();
    ring = new String[rs1.getInt(1)];
        ResultSet rs = st.executeQuery("select username,topic,question,datesent
        from query");
    while(rs.next())
    {
        ring[i] =
        rs.getString(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" ";
        i++;
    }

}
catch(Exception e)
{
    System.out.println("the error in returner1 is "+e);
}
return(ring);
}

public int save1(String st1,String st2,String st3,String st4,String st5,String date)
{
    int h=2;
    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
        PreparedStatement st = con.prepareStatement("insert into reply
        values(?,?,?,?,?,?)");
        st.setString(1,st1);
        st.setString(2,st2);
        st.setString(3,st3);
        st.setString(4,st4);
        st.setString(5,st5); }

    catch(Exception e)
    {
        System.out.println("the error in save1 is "+e);
    }
    return(h);
}
```

```
public String[] retry(String st1)
{
    String ring[]=null;
    int i=0;

    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection("jdbc:odbc:trial","","");
        Statement st = con.createStatement();
        ResultSet rs1 = st.executeQuery("select count(*) from reply where
to="+st1+"");
        rs1.next();
        ring = new String[rs1.getInt(1)];
        ResultSet rs = st.executeQuery("select from1,topic,question,answer,daterep from
reply where to="+st1+"");
        while(rs.next())
        {
            ring[i] =
rs.getString(1)+" "+rs.getString(2)+" "+rs.getString(3)+" "+rs.getString(4)+" "+rs
.getString(5)+" ";
            i++;
        }
        System.out.println("HA11");
    }
    catch(Exception e)
    {
        System.out.println("the error in retry is "+e);
    }
    return(ring);
}

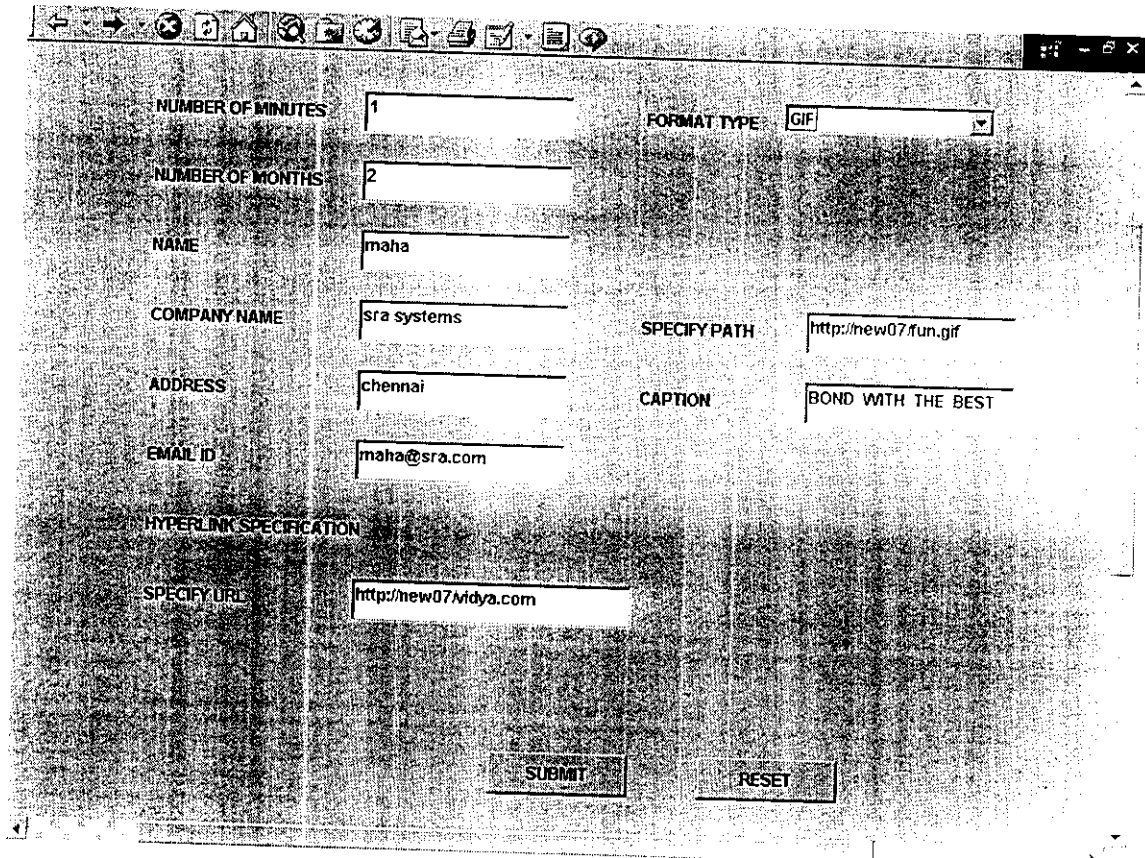
public static void main(String args[])
{
    System.setSecurityManager(new RMISecurityManager());
    try
    {
        System.out.println("Server is ready");
        Registry reg = LocateRegistry.createRegistry(1999);
        ser ob= new ser();
        reg.bind("tulec",ob);
        System.out.println("server is ok");
    }
}
```

```
    catch(Exception e)
    {}
}
}
```

chatserver.java

```
package pack;
import java.rmi.*;
interface chatserver extends Remote
{
    public int dis(String s1,String s2) throws RemoteException;
    public int booking(String min,String mon,String combo,String caps,String
    usl,String pat,String name,String comp,String addr,String mail) throws
    RemoteException;
    public int store(String st1,String st2) throws RemoteException;
    public String[] recall() throws RemoteException;
    public String[][] check(String m1,String m2,String com) throws
    RemoteException;
    public String[][] checker() throws RemoteException;
    public String[] returner() throws RemoteException;
    public int next(String s1,String s2) throws RemoteException;
    public int store1(String st1,String st2,String st3) throws RemoteException;
    public int save(String st1,String st2,String st3,String date) throws
    RemoteException;
    public String[] returner1() throws RemoteException;
    public int save1(String st1,String st2,String st3,String st4,String st5,String date)
    throws RemoteException;
    public String[] retry(String st1) throws RemoteException;
}
```

REGISTRATION FORM

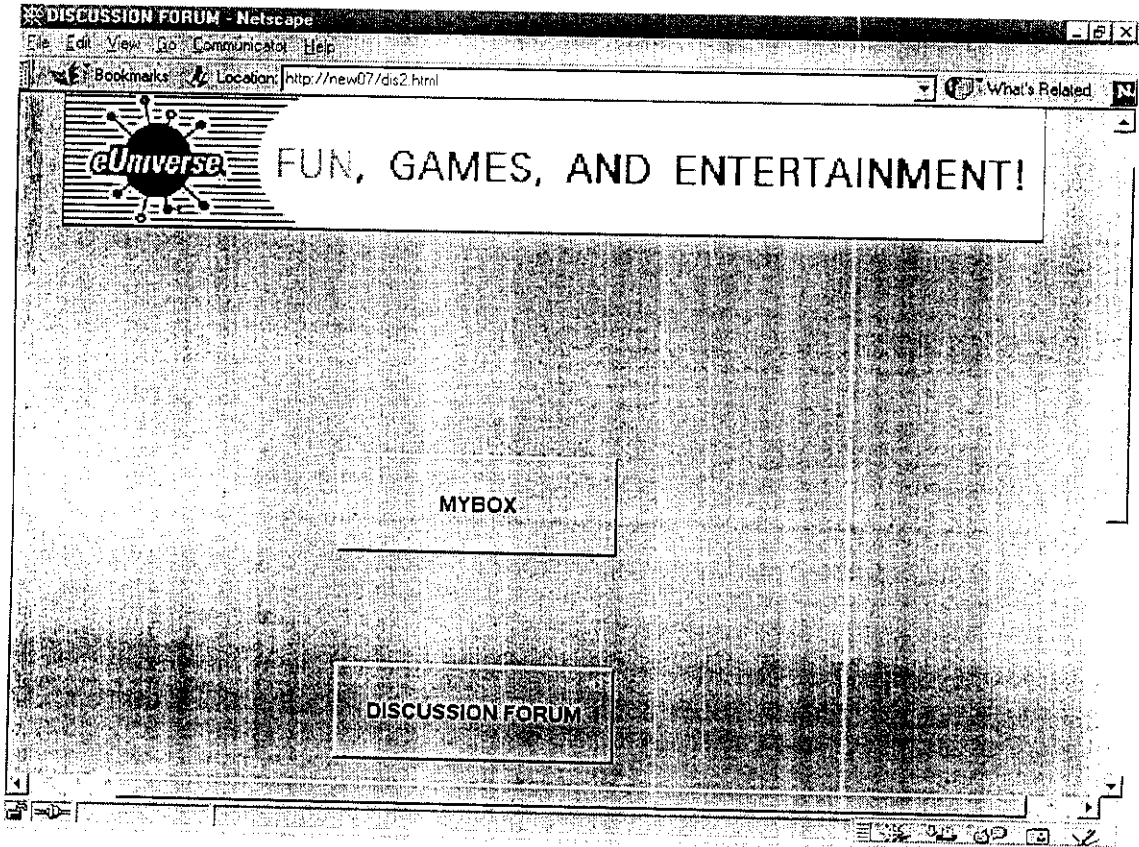


The image shows a screenshot of a web browser window displaying a registration form. The browser's address bar and toolbar are visible at the top. The form contains several input fields and two buttons. The fields are filled with the following information:

Field Label	Value
NUMBER OF MINUTES	1
NUMBER OF MONTHS	2
NAME	maha
COMPANY NAME	sra systems
ADDRESS	chennai
EMAIL ID	maha@sra.com
FORMAT TYPE	GIF
SPECIFY PATH	http://new07.fun.gif
CAPTION	BOND WITH THE BEST
SPECIFY URL	http://new07vidya.com

At the bottom of the form, there are two buttons: "SUBMIT" and "RESET".

DISCUSSION FORUM



INVOICE FORM

CLIENT - Netscape

File Edit View Go Communication Help

Back Forward Reload Home Search Netscape Print Security Sign

Bookmarks Location: http://new07/invoice.html

What's Related

NAME	maha	COMPANY	sra systems
EMAIL ID	maha@sra.com	ADDRESS	chemical
PARTICULAR:			
MINUTES		RATE	100
		TOTAL	1000
MONTH		RATE	500.0
		TOTAL	1000.0
FORMAT	GF	RATE	25.0
		TOTAL	25.0

Start

4:35 PM