Date

Expt. No. _____ Page No. 1

Vision of the Institute

To become a renowned centre of excellence based on learning & work towards academic, professional, cultural & social enrichment of the lives of individuals & communities.

Mission of the Institute

M1:- Focus on evaluations of learning outcomes & motivate students to inculcate research aptitude by project based learning.

M2:- Identify, based on informed perception of Indian, regional & global needs, areas of focus & provide platform to gain knowledge & solutions.

M3:- Offer opportunities for interaction between academia & industry.

M4:- Develop human potential to its fullest extent so that intellectually capable.

Teacher's Signature _____

Date

Expt. No. _____ Page No. 1

Vision of the Department

To become renowned centre of excellence in CSE & make competent engineers.

Mission of the Department

M1:- To impart outcome based education for emerging technologies in the field of CSE.

M2:- To provide opportunities for interaction b/w academia & Industry.

M3:- To provide platform for lifelong learning by accepting the change in technologies.

M4:- To develop aptitude of fullfilling social responsibilities.

Arya

Teacher's Signature _____

Date

Expt. No. _____ Page No. 9

Course Outcomes

CO1:- Develop in depth understanding of swings programming by writing user interfaceprograms in Swings.

CO2:- Develop client side application involving JDBC, RMI & Socket Programming in Java.

CO3:- Develop server side applications Involving J2EE, Servlets & JSP.

Teacher's Signature _____

Date

Expt. No. _____ Page No. 3

Do's

1. Please switch off mobile phone.
2. Enter lab with complete preparation
3. Check whether all peripherals are available at your desktop.
4. Intimate the lab incharge whenever you are uncomfortable.
5. Arrange all devices.
6. Properly shut down the system.
7. Keep bag outside the lab.
8. Enter lab on schedule time.
9. Maintain decorum of lab.
10. Get your external device checked by lab in charge.

Don'ts

1. No one is allowed to bring storage devices.
2. Don't misbandle the system.
3. Don't leave the system on standing for long time.
4. Don't bring any external device.
5. Don't make noise in lab.
6. Don't bring mobile in the lab.
7. Don't enter Lab without permission.
8. Don't litter in lab.
9. Don't delete or make any modification in system files.
10. Don't eat inside the lab.

Ajaya

Teacher's Signature _____

Experiment -1

1 (A)

- **Aim:-** To create a JAVA applet that prints a string "Hello World" on the screen.

- **Theory**

An applet is a small Java program that runs inside a web browser or an applet viewer. Applets are primarily used to create dynamic & interactive web applications. Unlike standalone applications, applets do not have a main method.

- **Life Cycle of Applet :-**

An applet's life cycle is managed by browser or the applet viewer & consist of the following methods.

(i) init :-
→ called when applet is first loaded.

(ii) start() :-
→ called after init() & invoked whenever the applet is started.

(iii) paint (Graphics g) :-
This method is called to draw the applet's output. It is called everytime the applet need to be repainted.

Ayaja

Teacher's Signature

(iv) stop():- This method is called when the applet is stopped.

(v) destroy():- This method is called when the applet is being destroyed.

Graphics method used :-
public abstract void drawString (String str, int x, int y).
It is used to draw a specified string at specific position (x, y) in pixels.

Steps to execute & compile an applet code:-
In CMD prompt

Step1:- Set path of javac to user directory.
set path = %path%; "C:\Programfiles\(jdk path)"

Step2:- javac AppletDemo.java (filename)

Step3:- appletviewer AppletDemo

Ananya

Teacher's Signature

Program Code:-

```
import java.applet.*;
import java.awt.*;

public class AppletDemo extends Applet {
    public void paint(Graphics g)
    {
        g.drawString("Hello Applet", 100, 300);
    }
// <applet code = "AppletDemo.class" width = 600
  height = 600 >
// </applet>
}
```

Date

Expt. No. _____ Page No. 7

1(B)

Aim:- To create a Java applet to print different shapes on screen.

Program Code

```
import java.applet.*;
import java.awt.*;
public class AppletDemo extends Applet {
public void paint (Graphics g)
{
    g.setcolor (color.red);
    g.drawstring ("Welcome", 50, 50);
    g.drawline (20, 30, 20, 300);
    g.drawRect (70, 100, 30, 30);
    g.fillRect (170, 100, 30, 30);
    g.drawOval (70, 200, 30, 30);

    g.setcolor (color.pink);
    g.fillOval (170, 200, 30, 30);
    g.drawArc (90, 250, 30, 30, 30, 270);
    g.fillArc (270, 150, 30, 30, 180);
}
// <applet code = "AppletDemo.class" width = 600,
   height = 300>
// </applet>
}
```

Ajaya

Teacher's Signature _____

Date

Expt. No. 2 Page No. 8

Experiment - 2

Introduction:-

AWT stands for Abstract window toolkit. It is an Application Programming interface (API) for creating Graphical User Interface (GUI) in Java. It allows Java programmers to develop window based application.

AWT provides various components like button label, checkbox etc. used as objects inside a Java Program. AWT components are platform independent. The class for AWT are provided by Java

Component class

The component class stands at the top of the AWT hierarchy, in an abstract class that contains all the properties of the component visible on the screen.

Container

The container is a component that contains other components like button, label, textfield etc.

Panel

It can be defined as a container that can be used to hold other components.

Teacher's Signature

Window :-

A window can be defined as a container that doesn't contain any border or menu bar. It creates a top-level view.

Frame :-

The frame is a subclass of window. It can be defined as a container with components like button, text field, label, etc.

Date

Expt. No. _____ Page No. 10

2(a)

Aim:- Write an AWT program to create button using frame.

Setpath ⇒ set path = %path% ; "C:\Programfiles\jdkpath)"

Program Code :-

```
import java.awt.*;
public class ButtonDemo extends Frame {
    ButtonDemo()
    {
        Button b = new Button ("Check");
        b.setBounds (30, 100, 80, 30);
        add (b);
        setSize (400, 400);
        SetVisible (true);
        Setlayout (null);
    }
    public static void main (String args [1]) {
        Button Demo bd = new ButtonDemo();
    }
}
```

Teacher's Signature _____

Date

Expt. No. 2(b) Page No. 11

2(b)

Aim:- Write an AWT program to close a window frame.

```
import java.awt.event.*;
import java.awt.*;
public class WindowClosingDemo extends Window
 Adapter
{
   Frame f;
   WindowClosingDemo()
   {
       f = new Frame();
       f.addWindowListener(this);
       f.setSize(600,600);
       f.setLayout(null);
       f.setVisible(true);
   }
   public void windowClosing(WindowEvent we)
   {
       f.dispose();
   }
   public static void main(String args[])
   {
       new WindowClosingDemo();
   }
}
```

Teacher's Signature

Date

Expt. No. 3 Page No. 12

Experiment - 3

Introduction :-

Swing in Java is lightweight GUI toolkit which has a wide variety of widget for building optimization windows based application. It is a part of JFC [Java foundation classes]. It is build on top of AWT API & entirely written in Java. It is platform independent unlike AWT & lightweight components.

Containes class :-

Any class which has other components in it called as a container class. For building GUI application at least one container class is necessary. Following are three types of container class.

→ Panel :- It is used to organize component on to a windows.

→ Frame :- A fully functional windows with icon & titles.

→ Dialogue :- It is like a pop up windows but not fully functional like Frame.

Ajaya

Teacher's Signature

Date

Expt. No. 3(A) Page No. 13

Aim:- Write a swing program to create button components.

```
import javax.swing.*;
public class JButtonDemo extends IFrame
{
    JButton b = new JButton ("Click Here");
    b.setBounds(50, 100, 95, 95);

    add(b);
    setsize(600, 600);
    setlayout(null);
    setvisible(true);
    }
    public static void main (String args[])
    {
        new JButtonDemo();
    }
}
```

Ayaya

Teacher's Signature

Date

Expt. No. 3(B) Page No. 14

Aim:- Write swing program to make an action button component.

```
import javax.swing.*;
import java.awt.event.*;

public class JButtonDemo extends JFrame
    implements ActionListener {
        JTextField tf;
        JButtonDemo()
        {
            JButton b = new JButton("Click Here");
            b.setBounds(50,200,150,20);
            tf = new JTextfield();
            tf.setBounds(50,50,150,120);
            b.addActionListener(this);
            add(b);
            add(tf);
            setSize(400,400);
            setLayout(null);
            setVisible(true);
}
public void actionPerformed(ActionEvent e){
        tf.setText("The button was clicked");
    }
  public static void main(String args[])
      {
        new JButtonDemo(); }
}
```

Ayaz

Teacher's Signature

Date

Expt. No. 4

Page No. 15

Experiment -4

Introduction

1. **Inet Address:-**
It is a class in Java that represent an IP address. It is used to identify a host (such as computer) by its IP address to retreive IP address. We can use Inet Address to retreive host or to perform DNS lookups. For eg. Inet address getByName ("hostname") will return IP address associate with hostname.

2. **Server Socket :-**
It is a java class used to create server that listens for incoming communication over a N/W. It establish communication channel for the server side client server architecture.

3. **Socket :-**
It is a class that represents a connection b/w a client & server over a network. It allows for sending & receiving data through I/O streams.

4. **Data Input Streams & Data Output Streams:-**
These are used to handle binary input & output b/w a client & server. These streams allows reading & writing primitive data types like int, char etc & king, in machine.

Ajaya

Teacher's Signature

Date

Expt. No. 4(a) Page No. 16

Aim:- Write a program to find IP address of local remote host

```
import java.net.*;
public class IPDemo {
public static void main (String args[]){
try
{
InetAddress localhost = InetAddress.getLocalHost();
System.out.println("IP address is: " +localhost
        getHostAddress());
}
catch (Exception e){
    e.printStackTraced();
    }
}
```

Teacher's Signature

Date

Expt. No. 4(B) Page No. 17

Aim:- To establish one way communication b/w client & server.

Server Side

```
import java.net.*;
import java.io.*;
public class MyServerDemo{
public static void main (String args[]){
try{
ServerSocket SS = new ServerSocket(5000);
Socket S = SS.accept();
DataInputStream din = new DataInputStream
    (S.getInputStream());
String str = (String)din.readUTF();
System.out.println(str);
din.close();
ss.close();
s.close();
}
catch(Exception e)
{
    e.printStackTrace();
    }
  }
}
```

Ajanta

Teacher's Signature

Date

Expt. No. _____ Page No. 10

Client side

```
import java.net.*;
import java.io.*;
public class MyClientDemo
{
  public static void main (String args [ ])
{
try {
Socket s = new Socket ("localhost", 6000);
Data Output Stream dout = new Data Output Stream
      (s. getoutput stream( ));
dout. write UTF ("Hello Server");
dout. Flush ( );
dout. close ( );
s. close( );
 }
 catch (exception e)
{
    e. print Stack Trace ( );
   }
 }
}
```

Ajanta

Teacher's Signature _____

Date

Expt. No. 4(C) Page No. 19

Aim:- To establish two way communication b/w client & server in Java.

Server side

```
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class MyServerCode {
    public static void main (String[] args) {
        try (ServerSocket ss = new ServerSocket(port:
            Socket s = ss.accept(5000);
            Socket s = ss.accept();
            DataInputStream din = new DataInputStream
                (s.getInputStream());
            DataOutputStream dout = new DataOutputStream
                (s.getOutputStream());
            Scanner scanner = new Scanner(System.
        System.out.println("Client Connected");
        String input;
        while (true) {
            String str = din.readUTF();
            System.out.println("Received from Client:" +str);
            if("end.equalsIgnoreCase(str)) {
                System.out.println("Client ended conversation");
                break;
            }
            System.out.println("Server");
```

Ayaya

Teacher's Signature

```
            input = scanner.nextLine();
            dout.writeUTF(input);
            dout.flush();
            if("end".equalsIgnoreCase(input))
            {
                System.out.println("Server ended the
                        conversation");
                break;
            }
        }
    }catch(Exception e){
        e.printStackTrace();
    }
}
}
```

Client Side Code

```
import java.net.*;
import java.io.*;
import java.util.Scanner;
public class MyClientDemo{
    public static void main(String[] args){
        try(Socket socket = new Socket(host: "localhost",
                                port: 5000);
        DataOutputStream dout = new DataOutputStream
                (socket.getOutputStream());
        DataInputStream din = new DataInputStream
                (socket.getInputStream());
        Scanner scanner = new Scanner(System.in);
```

```
String input;
while(true){
    System.out.print("Client: ");
    input = scanner.nextLine();
    dout.writeUTF(input);
    dout.flush();
if ("end".equalsIgnoreCase(input)){
    System.out.println("Client ended conversation"
                                        );
    break;
}
String response = din.readUTF();
System.out.println("Received from server: "
                        + response);
if ("end".equalsIgnoreCase(response))
{
    System.out.println("Server ended the
        conversation");
    break;
    }
  }
}catch (Exception e){
    e.printStackTrace();
  }
 }
}
```

Date

Expt. No. 5 Page No. 22

Experiment-5

Aim:- Case study on Model view controller (MVC)

Theory:-

What is MVC

→ Model view controller (MVC) framework is an architectural/design pattern that seperates an application into three main logical components Model, Views, & Controller.

→ Each component is built to handle specific development aspects of applications. It isolates the business logic & presentation layer from each other.

→ It was traditionally used for desktop GUIs. It is also used for designing mobile apps.

→ MVC was created by Trygve Reenskaug. The main goal of this design pattern was to solve problems of users controlling a large & complex data set.

When to use MVC

① When you want to maintain seperation of seperation of concern:- MVC helps to seperate the business logic (Model), the UI (View) & control flow (C).

② For scalable & maintainable applications:- Seperation of components makes it easier to scale & maintain.

Ananya

Teacher's Signature

③ For web applications that requires testability & reusability :- MVC allows for easy testing of each individual component & reusability of code.

④ For applications with multiple views of same data :- The model is independent of view, the same data can be displayed in different ways without duplicating.

Types of MVC

① Classic MVC
The controller receives input, updates the model & the View is automatically updated when model's data changes.

② Action Based MVC
Most commonly used in web development frameworks like Django, Ruby, ASP.net MVC. Controller handles HTTP requests, manipulate the model & passes necessary data to views.

③ MVVM - Model-View-View Model
Used in desktop & mobile applications, such as WPF & Angular JS. The ViewModel acts as abstraction of view, containing UI logic & binds model data to views.

Arjun

Teacher's Signature

④ Push Based MVC
Controller determines which view should be rendered "pushing data to view. Often used in Java frameworks like Spring MVC

⑤ Pull Based MVC
View is responsible for pulling data from model.

⑥ HMVC
It organizes components in hierarchy. Each MVC triad can act as a sub module of a parent module.

Advantages of MVC
① Codes are easy to maintain.
② The MVC model component can be tested seperately.
③ It supports test driven development.
④ SEO friendly.

Disadvantages
① Dificult to read, change, test.
② Useful building small applications.
③ Inefficiency of data access in views.
④ Increased complexity.

Date

Expt. No. 6 Page No. 25

Experiment-6

Aim:- Case study on JDBC

Theory

What is JDBC?

→ JDBC stands for Java Database connectivity.
→ It is an API provided by Oracle corporation to enable java application to interact with various types of databases.
→ JDBC allows developers to execute SQL statements, retrieve & manipulate data & manage databases connections in a secure & efficient manner.
→ It acts as a bridge between the Java Programming language & a wide variety of databases such as MySQL, PostgreSQL, Oracle, etc.

When to use JDBC:

① Data driven applications:- JDBC provides a standard interface to interact with relational database, such as MySQL, PostgreSQL, Oracle, SQLite.

② Custom SQL execution:- JDBC allows you to create, retrieve, update, delete data using SQL statements.

③ Standalone Java applications :- JDBC is good option

Ajaya

Teacher's Signature

to connect to databases.

(4) Dynamic Web applications:- In web applications that need to access databases, JDBC is used to establish the connection with database & retrieve data

(5) Cross Platform portability:- JDBC provides vendor neutral API for database access, which you can switch b/w different relational systems without changing much of your Java code.

Types of JDBC
Based on the basis of drivers

(1) JDBC ODBC Driver
This driver uses ODBC API to interact with the database. JDBC calls are translated into ODBC calls & then ODBC driver communicates with the database.

(2) Partial Java Driver
This driver converts JDBC calls into database specific calls using native libraries provided by database vendor.

(3) Network Protocol Driver
It uses middleware to convert JDBC calls into a

Ayesha

Teacher's Signature

database independent protocol, which is then forwarded to server that communicates with the database.

④ Pure Java Driver
This is 100% Java based driver that converts JDBC calls into database-specific protocol.

Advantages
① Provides an standard API to connect to various relational databases.
② It works accross all platforms that support Java.
③ Offers full control over executing raw SQL queries.
④ It supports transactions.

Disadvantages
① Requires a lot of repetitive code for connection management
② Primarily work on relational databases.
③ Developers must explicitly handle connection, statements, & result sets, which can lead to resource leaks.
④ Without optimization JDBC can lead to performance bottlenecks

Ananya

Teacher's Signature

Date

Expt. No. _____ Page No. 28

Experiment-7

Aim:- Case study on Enterprise Java Beans

Theory

What is EJB

→ EJB is an acronym for enterprise java Bean.
→ It is specification provided Sun Microsystems to develope secured, robust, scalable distributed applications.
→ To get information about distributed applications.
→ To run EJB application, you need an application server such as JBoss, Glassfish, Weblogic, Websphere
→ It performs life cycle management, security, transaction management & object pooling.
→ It is also called as server side component.

When use Enterprise Java Bean

→ Application needs Remote access.
→ Application needs to be scalable.
→ EJB application supports load balancing, clustering & fail over.
→ Application needs encapsulated business logic.
→ EJB application is seperated from presentation & persistent layer.

Ajaya

Teacher's Signature _____

Date

Expt. No. _____ Page No. 29

Types of EJB

1. Session Beans
It handles business logic for a client request
Types
   1. Stateless Session bean
   2. Stateful Session Bean
   3. Singleton Session Bean

2. Message Driven Bean
Used to process messages sent to a Java Message Service (JMS) queue or topic.

3. Entity Bean
It encapsulates the state that can be persisted in database.
Now it is replaced with JPA.

Advantages of EJB

1. Handles complex task like transactions, security & concurrency.
2. Automatically manages transactions, ensuring data integrity.
3. Designed for large scale applications.

Ajaya

Teacher's Signature _____

④ Provides role based access control.

Disadvantages of EJB.

① Configuration & setup can be complex.

② Features like transaction management & remote calls can introduce performance slowdowns.

③ EJB concepts can be difficult for new developers.

④ Requires more resources as compared to Spring.

Date

Expt No. 8

Page No.

Experiment-8

Aim:-

Theory:-

What is RMI?
Remote Method Invocation (RMI) is an API that allows an object to invoke a method on an object that exist in another address space, which could be on the same machine or on a remote machine.

Through RMI, an object running in java virtual machine (JVM) present on computer can invoke methods on an object present in another JVM.

The client & server communication is handled by using two intermidiate objects:

Stub Object:- The stub object on client machine builds an information block & sends this info. to the server.
The block consist of:
- An identifier of the remote object to be used.
- Method name which is to be invoke.
- Parameters to Remote JVM.

Page 12

Teacher's Signature

Date

Page No.

Expt No.

Skeleton Object :- The skeleton object passes the request from the stub object to remote object.
- It calls the desired method on the real object present on server.
- It forwards the parameters received from the stub object to the method.

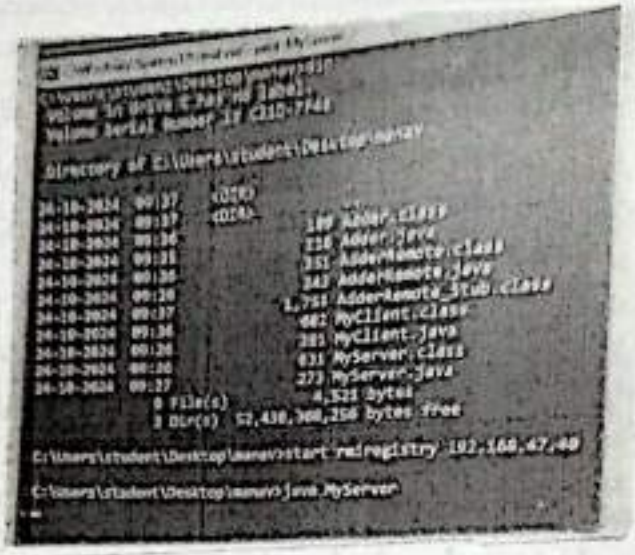
Steps to implement interface

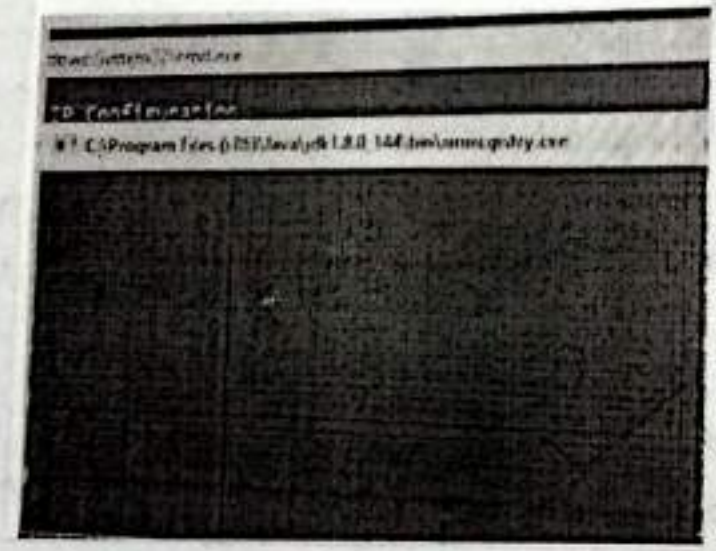
1. Defining a remote interface.
2. Implementing the remote interface.
3. Creating a stub & skeleton objects.
4. Start the rmiregistery.
5. Create & execute the server application program.
6. Create & execute the client application program.

Commands:-

1. javac Adder.java
2. javac AdderRemote.java
3. javac MyServer.java
4. javac MyClient.java
5. Dir (4 of java, 4 class of java)
6. rmic AdderRemote (Dir of stub, class added)
7. Start rmiregistry 5000
8. java MyServer
9. java MyClient

Teacher's Signature


Directory of C:\Users\student\Desktop\manav
24-10-2024 09:37 <DIR> .
24-10-2024 09:37 <DIR> ..
24-10-2024 09:36 189 Adder.class
24-10-2024 09:36 116 Adder.java
24-10-2024 09:35 351 AdderRemote.class
24-10-2024 09:35 143 AdderRemote.java
24-10-2024 09:36 1,751 AdderRemote_Stub.class
24-10-2024 09:37 482 MyClient.class
24-10-2024 09:36 281 MyClient.java
24-10-2024 09:26 631 MyServer.class
24-10-2024 09:26 273 MyServer.java
24-10-2024 09:37 4,321 bytes
8 File(s)
3 Dir(s) 52,430,368,256 bytes free
C:\Users\student\Desktop\manav>start rmiregistry 192.168.47.40
C:\Users\student\Desktop\manav>java MyServer




Windows IP Configuration
Ethernet adapter Ethernet:
Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::cd09:2839:1f9a:dc0%12
IPv4 Address. . . . . : 192.168.47.40
Subnet Mask . . . . . : 255.255.254.0
Default Gateway . . . . . : 192.168.46.1
E:\prakhar>java MyClient
30

Date

Expt. No. _____ Page No. _____

```
import java.rmi.*;
public interface Adder extends Remote {
    public int add(int x, int y) throws
                                Remote Exception;
}

import java.rmi.*;
import java.rmi.server.*;
public class AdderRemote extends UnicastRemote Object implement
                                Adder{
    AdderRemote() throws RemoteException{
    super();
}
public class int add(int x, int y) throws
                                Remote Exception{
        return x+y;
    }
}
```

Ayaya

Teacher's Signature _____

Date

Expt. No. _____ Page No. _____

```
import java.rmi.*;
import java.rmi.registry.*;
public class MyServer {
    public static void main(String args[]){
    try {
        Adder stub = new AdderRemote();
        Naming.rebind("rmi:localhost:5000/rmidemo
                                    stub);
    }
    catch(Exception e)
    {
        System.out.println(e);
    }}}

import java.import rmi.*;
public class MyClient {
    public static void main(String args[])
    {
        try {
            Adder stub = (Adder) Naming.lookup(
                "rmi://localhost:5000/rmidemo");
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
    }
}
```

Teacher's Signature _____

Date

Expt. No. 9 Page No.

Experiment-9

Aim:- Write a servlet program to display "Hello World on browser.

Servlets:-
Servlets are java classes which service HTTP requests & implements the javax.servlet.Servlet interface. Web app developers typically write servlets that extend java.servlet.http.HttpServlet.

Code:-

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        printwriter out = response.getWriter();
        out.println("<h1>" + "Hello World" + "</h1>");
    }
    public void destroy(){
        //code
    }
}
```

Teacher's Signature

Date

Expt No. _____ Page No. _____

b) Write a servlet program user registration & amp; then control will be transferd to second page.

```
<html>
<head>
<title> </title>
</head>
<body>
<p>
<%
String fname = request.getParameter("fname");
String lname = request.getParameter("lname");
string uname = request.getParameter("uname");
String email = request.getParameter("email");
String pwd = request.getParameter("pwd");
string cpwd = request.getParameter("cpwd");
string add = request.getParameter("add");
out.print("First name is: "+fname+"<br>");
out.print("last name is: "+lname+"<br>");
out.print("User name is: "+uname+"<br>");
Out.print("Email is: "+email+"<br>");
out.print("Password is: "+pwd+"<br>");
out.print("Confirm Password is: "+cpwd+"<br>");
out.print("Address is: "+add+"<br>");
%>
</p>
</body>
</html>
```

Ahuja

Teacher's Signature _____

Date

Expt. No. 10 Page No. 37

Experiment - 10

Aim:- Write a JSP program for user login form with static & dynamic database.

Static Database

* Filename

index.html:- for username & password

process.jsp:- for static database

welcome.jsp:- Welcome file

index.html

```
<html>
<head>
<meta http.equiv = "Content - Type" Content = "text
  /html"; Charset = UTF - 8">
</head>
<body>
<form action = "process.jsp" method = "POST">
Username: <input type = "text" name = "uname">
password: <input type = "password" name = "pass">
<input type = "submit">
</form>
</body>
</html>
```

Teacher's Signature

process.jsp

```
<%@ page contentType = "text/html" page encoding
  = "UTF-8" %>
<!DOCTYPE html>
<%
String a = request.getParameter("uname");
String b = request.getParameter("pass");
if (a.equals("queue") && b.equals("queue"))
{
    response.sendRedirect("Welcome.jsp");
}
%>
```

Welcome.jsp

```
<%@ page Content Type = "text/html" page Encoding
  = "UTF-8" %>
<!DOCTYPE html>
out.println("Welcome to world of jsp");
%>
```

Date

Expt. No. 11 Page No. 39

Experiment - 11

Aim:- Case Study on J2EE

Theory

What is J2EE?

Java 2 Platform, Enterprise Edition is a platform independent Java centric environment used for developing, building & deploying web based enterprise application. It extends the Java Standard SE with specifications for enterprise level services.

Key Features of J2EE

1. Platform Independence :- Applications written in J2EE can run on any platform.
2. Multi-Tier Architecture :- Divides applications into tiers, promotes scalability & maintability.
3. Component Based Development :- Uses reusable components like servlets, JSPs & EJBs.
4. Scalability & Reliability :- Designed to handle large scale & reliable enterprise-level applications.
5. Security :- Includes built in mechanism for authentication, authorization & data encryption.
6. Support for Web Services :- Enables integration with other systems & services via protocols like Soap & Rest.

Teacher's Signature

J2EE Architecture

J2EE follows a multi-tiered Architecture

1. Client Tier:- Includes web browser or standalone applications.
2. Web-Tier:- Handles HTTP requests, often using servlets & JSP.
3. Business Tier:- Contains business logic, often implemented with EJBs
4. Enterprise Information System Tier:- Manages interaction with databases & other backend systems.

Core J2EE Technologies:-

1. Servlets:- Java program that run on a server & handle HTTP request.
2. Java Server Pages:- Simplifies the development of dynamic web pages.
3. EJB:- Server side components that encapsulates business logic.
4. JDBC:- API for connecting & executing queries on database.
5. RMI:- Allows methods to be invoked on objects running on different JVMs.

Teacher's Signature

Advantages of J2EE

1. Simplifies complex enterprise applications.
2. Promotes modular design, making applications easier to maintain.
3. Provides a robust framework for building scalable & secure systems.
4. Allows developers to leverage reusable components.