Index

Topic	Program	Title	Date	Sign
1 Java Networking	1	Implement the given code and see the output of program		
	2	1) Develop a tcp/ip base client server application establishing the talk between two users. User types in one window, must be visible to another user's top panel and into his own window in bottom panel and vice versa. 2) Develop a TCP/IP application supplying the series of data from the client (one application) and send it to server(another application) and sort it from server side and display their result onto the client side. 3) Implement program1 using UDP sockets.		
	3	Theory Assignment		
2 JDBC Programming	1	1) Implement the given code and see the output of program		
	2	2) Create an application to fill student registration form and submit data into table of Oracle/MS ACCESS. (use JDBC) 3) Write an application which list content of table of a database 4) Write an application to update content of table. Get values from key board.(Use parameterized query) 5) Using the JDBC API, display all the records from the database table, selected from command line argument or table selected from combo box 6) Write a Java application to invoke a stored procedure using a CallableStatement. For this a stored procedure called incrementSalary may be developed to increase all the employee's salary by a percentage specified in the parameter.		
	3	Theory Assignment		
3 Servlet	1	1) a) Develop server side application using servlet for collecting employee information, validation the existing record of employee and retrieval of employees records. Develop 3 separate servlets for collection, validation and for retrieval. b) Develop a client side web base form or applet for the server side employee information as in a). 2) For above application implement a login		

		page. A user can use above pages only after login using correct userID and password. Implement logout facility also. Do session tracking using any one of the four methods discussed in class.		
	2	Theory Assignment		
4 JSP	1	1) Write a simple JSP program for user Registration & then control will be transfer it into login page. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database. 2) In above practical perform session tracking.		
5 JSF		Implement a simple hello world web application using JSF.		
6 JSF		Write a JSF application for user Registration which forward to login page if successful registration. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database		
7 Hibernate		Create a java application using hibernate to use persistence object. Consider emp1000 table.		
8 Hibernate		Write a java program which uses HQL to access records from emp1000 table		
9 Spring MVC		Implement a simple hello world web application using Spring.		
10 Spring MVC		Write a Spring application for user Registration which forward to login page if successful registration. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database		

Practical -1 Network Programming

AIM: Develop a TCP/IP base client server application establishing the talk between two users. User types in one window, must be visible to another user's top panel and into his own window in bottom panel and vice versa.

***Client Application*:**

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

public class LocalTcpClient_1{
	public static void main(String[] args){
	try(Scanner scan = new Scanner(System.in)){

		/*
		* Two Classes are being used: Socket and ServerSocket
		* Socket and ServerSocket classes are used for connection-oriented socket
programming.
		* The Socket class is used to communicate client and server.
		* Through this class, we can read and write message.
		* The ServerSocket class is used at server-side.
		*/

		//To create the client application pass the IP address or hostname of the Server
and a port number
		Socket socket = new Socket("localhost",6969);

		//returns the InputStream attached with this socket.
		DataInputStream in = new DataInputStream(socket.getInputStream());

		//returns the OutputStream attached with this socket.
		DataOutputStream out = new DataOutputStream(socket.getOutputStream());

		// Returns the local port to which this socket is bound.
		System.out.println("Client started on local port: " + socket.getLocalPort());

		// Returns the remote port to which this socket is connected.
		System.out.println("Server Listens on remote port: " + socket.getPort());

		String from_server,to_server;
		while(true){
			System.out.print("Client: ");
			to_server = scan.nextLine();
```

```
                        out.writeUTF(to_server); // write a string to the output stream using UTF-
8 encoding in portable manner.
                        if(to_server.toLowerCase().equals("bye"))break;
                        from_server = in.readUTF(); //  read a string that has been encoded using
the UTF-8 format.
                        System.out.printf("Server: %s\n",from_server);
                        if(from_server.toLowerCase().equals("bye"))break;
                }
                socket.close();

            }catch(Exception e){System.out.println(e);}
        }
}
```

***Server Application*:**

```
import java.io.*;
import java.net.*;
import java.util.Scanner;
public class LocalTcpServer_1{
        public static void main(String[] args){
                try(Scanner scan = new Scanner(System.in)){
                        /*
                        * Two Classes are being used: Socket and ServerSocket
                        * Socket and ServerSocket classes are used for connection-oriented socket
programming.
                        * The Socket class is used to communicate client and server.
                        * Through this class, we can read and write message.
                        * The ServerSocket class is used at server-side.
                        */

                        //To create the server application pass the port number
                        ServerSocket server = new ServerSocket(6969);

                        /*
                        * The accept() method of ServerSocket class blocks the console
                        * until the client is connected. After the successful connection
                        * of client, it returns the instance of Socket at server-side.
                        */
                        //  Listens for a connection to be made to this socket and accepts it.
                        Socket socket = server.accept();

                        //returns the InputStream attached with this socket.
                        DataInputStream in = new DataInputStream(socket.getInputStream());

                        //returns the OutputStream attached with this socket.
```

```
                        DataOutputStream out = new
DataOutputStream(socket.getOutputStream());

                        String from_client,to_client;
                        while(true){
                                from_client = (String)in.readUTF(); //  read a string that has been
encoded using the UTF-8 format.
                                System.out.printf("Client: %s\n",from_client);
                                if(from_client.toLowerCase().equals("bye"))break;
                                System.out.print("Server: ");
                                to_client = scan.nextLine();
                                out.writeUTF(to_client); // write a string to the output stream using
UTF-8 encoding in portable manner.
                                if(to_client.toLowerCase().equals("bye"))break;
                        }
                        server.close();
                } catch(Exception e){System.out.println(e);}
        }
}
```

```
E:\javaperformance\collage\SEM6>java LocalTcpClient_1
Client started on local port: 56214
Server Listens on remote port: 6969
Client: Hello!!
Server: Hey!
Client: How's Going?
Server: Doing Advance java. wby?
Client: ooooohhhhhh.. Are you doing "Advance"?
Server: Yeah
Client: i have Already done.
Server: oh, Now who is Advance?
Client: hahaha
Server: Bye
```

```
E:\javaperformance\collage\SEM6>java LocalTcpServer_1
Client: Hello!!
Server: Hey!
Client: How's Going?
Server: Doing Advance java. wby?
Client: ooooohhhhhh.. Are you doing "Advance"?
Server: Yeah
Client: i have Already done.
Server: oh, Now who is Advance?
Client: hahaha
Server: Bye
```

AIM: Develop a TCP/IP application supplying the series of data from the client (one application) and send it to server (another application) and sort it from server side.

***Client Application*:**

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

public class LocalTcpClient{
	public static void main(String[] args){
	try{
		Scanner scan = new Scanner(System.in);

		/*
		* Two Classes are being used: Socket and ServerSocket
		* Socket and ServerSocket classes are used for connection-oriented socket
programming.
		* The Socket class is used to communicate client and server.
		* Through this class, we can read and write message.
		* The ServerSocket class is used at server-side.
		*/

		//To create the client application pass the IP address or hostname of the Server
and a port number
		Socket socket = new Socket("localhost",6969);

		//returns the InputStream attached with this socket.
		DataInputStream in = new DataInputStream(socket.getInputStream());

		//returns the OutputStream attached with this socket.
		DataOutputStream out = new DataOutputStream(socket.getOutputStream());

		// Returns the local port to which this socket is bound.
		System.out.println("Client started on local port: " + socket.getLocalPort());

		// Returns the remote port to which this socket is connected.
		System.out.println("Server Listens on remote port: " + socket.getPort());

		System.out.print("Enter size of array: ");
		int n = scan.nextInt();
		System.out.println("Enter "+n+" array elements:");
		int[] arr = new int[n];
		for(int i=0;i<n;++i)arr[i]=scan.nextInt();
```

```
                out.writeInt(n);
                for(int i=0;i<n;++i){
                        out.writeInt(arr[i]); // write an int to the output stream as bytes
                }
                System.out.print("Sorted elements: ");
                int x;
                for(int i=0;i<n;++i){
                        x = in.readInt(); // read input bytes and return an int value.
                        System.out.print(x+" ");
                }

            }catch(Exception e){System.out.println(e);}
        }
}
```

***Server Application*:**

```
import java.io.*;
import java.net.*;
import java.util.Arrays;
public class LocalTcpServer{
        public static void main(String[] args){
        try{
                /*
                * Two Classes are being used: Socket and ServerSocket
                * Socket and ServerSocket classes are used for connection-oriented socket
programming.
                * The Socket class is used to communicate client and server.
                * Through this class, we can read and write message.
                * The ServerSocket class is used at server-side.
                */

                //To create the server application pass the port number
                ServerSocket server = new ServerSocket(6969);

                /*
                * The accept() method of ServerSocket class blocks the console
                * until the client is connected. After the successful connection
                * of client, it returns the instance of Socket at server-side.
                */
                //  Listens for a connection to be made to this socket and accepts it.
                Socket socket = server.accept();

                //returns the InputStream attached with this socket.
                DataInputStream in = new DataInputStream(socket.getInputStream());
                //returns the OutputStream attached with this socket.
```

```
                DataOutputStream out = new DataOutputStream(socket.getOutputStream());

                int n = in.readInt(); // read input bytes and return an int value.
                System.out.println("array size "+n+" recived.");
                int[] arr = new int[n];
                for(int i=0;i<n;++i){
                        arr[i] = in.readInt();
                }
                Arrays.sort(arr);
                for(int i=0;i<n;++i)out.writeInt(arr[i]); // write an int to the output stream as bytes
        } catch(Exception e){
                System.out.println(e);

        }
        }
}
```

```
E:\javaperformance\collage\SEM6>javac LocalTcpClient.java

E:\javaperformance\collage\SEM6>java LocalTcpClient
Client started on local port: 56201
Server Listens on remote port: 6969
Enter size of array: 8
Enter 8 array elements:
12 -98 2 34 9 8 11 1
Sorted elements: -98 1 2 8 9 11 12 34
```

```
E:\javaperformance\collage\SEM6>javac LocalTcpServer.java

E:\javaperformance\collage\SEM6>java LocalTcpServer
array size 8 recived.
```

AIM: Implement program1 using UDP sockets.

***Client Application*:**

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

class Udp_client{
        public static void main(String []args){
                try(Scanner scan = new Scanner(System.in)){
```

```
/*
* DatagramSocket and DatagramPacket classes are used for connection-less
socket programming.
* A datagram socket is the sending or receiving point for a packet delivery
service. Each packet
* sent or received on a datagram socket is individually addressed and routed.
Multiple packets
* sent from one machine to another may be routed differently, and may arrive in
any order.
*/
// creates a datagram socket and binds it with the available Port Number on the
localhost machine.
DatagramSocket datagram_socket = new DatagramSocket();

// Gets the local address to which the socket is bound.
InetAddress ip = InetAddress.getLocalHost();
byte buffer[] = null;
// Returns the local port to which this socket is bound.
System.out.println("Client started on local port: " +
datagram_socket.getLocalPort());

while(true){
    System.out.print("Client: ");
    String input = scan.nextLine();
    buffer = input.getBytes();

    /*
    * Datagram packets are used to implement a connectionless packet
delivery service.
    * Each message is routed from one machine to another based solely on
information
    * contained within that packet. Multiple packets sent from one machine to
another
    * might be routed differently, and might arrive in any order. Packet
delivery is not guaranteed.
    */
    // Constructs a datagram packet for sending packets of length length to the
specified port number on the specified host.
    DatagramPacket send_datagram = new
DatagramPacket(buffer,buffer.length,ip,6969);

    // Sends a datagram packet from this socket
    datagram_socket.send(send_datagram);
    if(input.toLowerCase().equals("bye"))break;

    // Clear the buffer after every message.
```

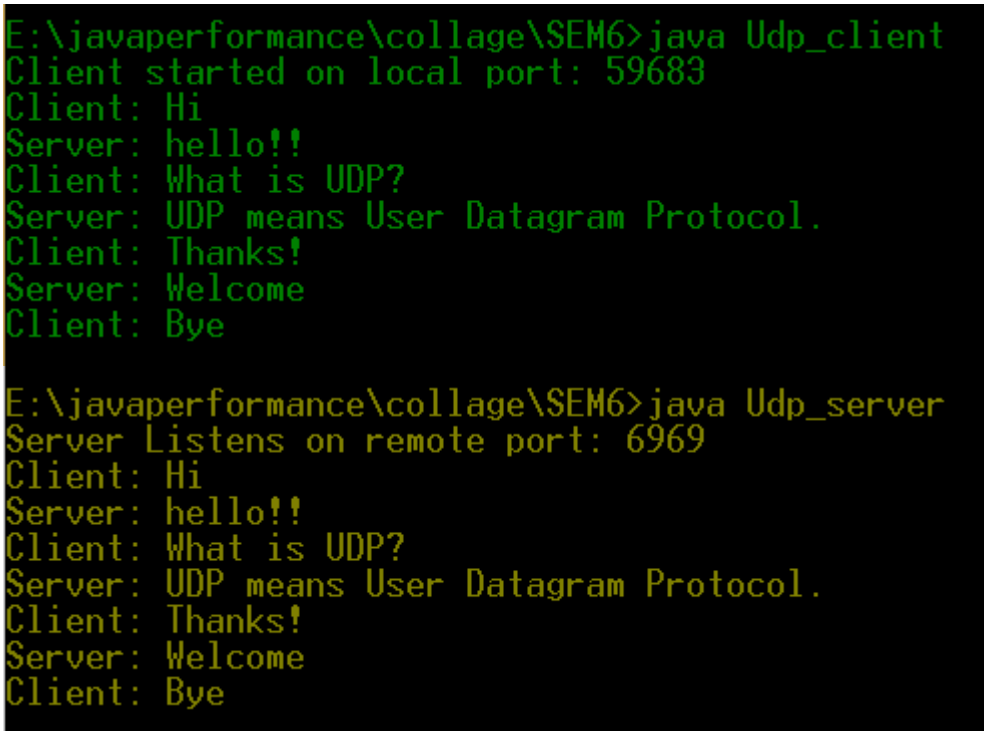
```
                        buffer = new byte[1024];

                        //Constructs a DatagramPacket for receiving packets of given length.
                        DatagramPacket receive_datagram = new
DatagramPacket(buffer,buffer.length);

                        // Receives a datagram packet from this socket.
                        datagram_socket.receive(receive_datagram);

                        // The getData() method of Java DatagramPacket class returns the data
buffer.
                        // Any data that is to be received or is to be sent, firstly starts from the
                        // offset(starting index, by default 0) in the buffer and then runs for length
long.
                        input = new String(receive_datagram.getData());

                        System.out.println("Server: "+input.trim());
                        if(input.trim().toLowerCase().equals("bye"))break;
                }

                }catch(IOException e){
                        e.printStackTrace();
                }catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

***Server Application*:**

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

class Udp_server{
        public static void main(String []args){
                try(Scanner scan = new Scanner(System.in)){

                /*
                * DatagramSocket and DatagramPacket classes are used for connection-less
socket programming.
                * A datagram socket is the sending or receiving point for a packet delivery
service. Each packet
                * sent or received on a datagram socket is individually addressed and routed.
Multiple packets
```

```
* sent from one machine to another may be routed differently, and may arrive in
any order.
*/
// creates a datagram socket and binds it with the given Port Number.
DatagramSocket datagram_socket = new DatagramSocket(6969);

// Returns the local port to which this socket is bound.
System.out.println("Server Listens on remote port: " +
datagram_socket.getLocalPort());
byte[] buffer = new byte[1024];
while(true){

/*
* Datagram packets are used to implement a connectionless packet
delivery service.
* Each message is routed from one machine to another based solely on
information
* contained within that packet. Multiple packets sent from one machine to
another
* might be routed differently, and might arrive in any order. Packet
delivery is not guaranteed.
*/
// Constructs a DatagramPacket for receiving packets of given length.
DatagramPacket receive_datagram = new
DatagramPacket(buffer,buffer.length);

// Receives a datagram packet from this socket.
datagram_socket.receive(receive_datagram);

// Returns the address to which this socket is connected.
InetAddress ip = receive_datagram.getAddress();

// The getData() method of Java DatagramPacket class returns the data
buffer.
// Any data that is to be received or is to be sent, firstly starts from the
// offset(starting index, by default 0) in the buffer and then runs for length
long.
String input = new String(receive_datagram.getData());

System.out.println("Client: "+input.trim());
if(input.trim().toLowerCase().equals("bye"))break;
System.out.print("Server: ");
input = scan.nextLine();
// Clear the buffer after every message.
buffer = new byte[1024];
buffer = input.getBytes();
```

```
                        // Constructs a datagram packet for sending packets of given length to the
specified port number on the specified host.
                        DatagramPacket send_datagram = new
DatagramPacket(buffer,buffer.length,ip,receive_datagram.getPort());

                        // Sends a datagram packet from this socket
                        datagram_socket.send(send_datagram);

                        buffer = new byte[1024];
                        if(input.toLowerCase().equals("bye"))break;
                }

                }catch(IOException e){
                        e.printStackTrace();
                }catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```



1) Explain InetAddress class and its use in network programming.

The java.net.InetAddress class is Java's encapsulation of an IP address. It is used by most of the other networking classes, including Socket, ServerSocket, URL, DatagramSocket, DatagramPacket, and more.Java InetAddress class represents an IP address. The java.net.InetAddress class provides methods to get the IP of any host name for example www.collegeek.com, www.google.com, www.facebook.com, etc.
An IP address is represented by 32-bit or 128-bit unsigned number. An instance of InetAddress represents the IP address with its corresponding host name. There are two types of address types: Unicast and Multicast. The Unicast is an identifier for a single interface whereas Multicast is an identifier for a set of interfaces.Moreover, InetAddress has a cache mechanism to store successful and unsuccessful host name resolutions.There are no constructors for this class but static methods which returns instances of InetAddress class for general use.

2) With the help of example show the use of URL and URLConnection class.

URL Class
The URL class is the gateway to any of the resource available on the internet. A Class URL represents a Uniform Resource Locator, which is a pointer to a "resource" on the World Wide Web. A resource can point to a simple file or directory, or it can refer to a more complicated object, such as a query to a database or to a search engine.Uniform Resource Locator-URL is a string of text that identifies all the resources on Internet, telling us the address of the resource, how to communicate with it and retrieve something from it.

A Simple URL looks like:
https://www.collegeek.com/8085_microprocessor/
protocol hostname filename

Components of a URL:-
Protocol: HTTP is the protocol here
Hostname: Name of the machine on which the resource lives.
File Name: The path name to the file on the machine.
Port Number: Port number to which to connect (typically optional).

URLConnection
URLConnection is an abstract class whose subclasses form the link between the user application and any resource on the web. We can use it to read/write from/to any resource referenced by a URL object.
There are mainly two subclass that extends the URLConnection class
HttpURLConnection: If we are connecting to any url which uses "http" as its protocol, then HttpURLConnection class is used.
JarURLConnection: If however, we are trying to establish a connection to a jar file on the web, then JarURLConnection is used.Once the connection is established and we have a

URLConnection object, we can use it to read or write or get further information about when was the page/file last modified, content length etc.

Example:

```
import java.io.*;
import java.net.*;
public class URLConnectionExample {
                        public static void main(String[] args){
                          try{
                                URL url=new URL("https://sidpro-
hash.github.io/Converter_Calculator/");

                                URLConnection urlcon=url.openConnection();
                                InputStream stream=urlcon.getInputStream();
                                int i;
                                while((i=stream.read())!=-1){System.out.print((char)i);}

                                System.out.println("Protocol: "+url.getProtocol());
                                System.out.println("Host Name: "+url.getHost());
                                System.out.println("Port Number: "+url.getPort());
                                System.out.println("File Name: "+url.getFile());

                          }catch(Exception e){System.out.println(e);}
                        }
}
```

```
    <p>To convert a binary number t
gits from right
      to left in the binary number
    <p>For example, the binary numb
s 8, and 11 is 3.</p>
  </div>
</body>
</html>
Protocol: https
Host Name: sidpro-hash.github.io
Port Number: -1
File Name: /Converter_Calculator/

F:\work of pro\SEM6\AJ>
```

3) How can we do network programming using UDP in java? Explain DatagramSocket and DatagramPacket in Java.

DatagramSockets are Java's mechanism for network communication via UDP instead of TCP. Java provides DatagramSockets can be used to both send and receive packets over the internet. It is also build on top of Ip.

UDP socket communication between a server and a client consists of several of phases:
Socket():Firstly a socket is defined in both server and client.
Bind(): Defined socket is assigned an id and a port in the running machine. This is optional for the client socket. Even if the client socket is not bound, it will automatically happen whenever the client initiates connecting to the server.
Recvfrom():After binding to a port in the machine, the server socket waits for a connection from a client socket. In the meantime, further execution of the current thread is halt (blocked) until the server socket receives a connection. This is the same for the client socket when waiting for a server response.
Sendto(): After connecting with a client, the server socket sends data to the client. This same method is used by the client socket to make a connection request to the server.
Close(): After successful data exchange, both sockets are closed i.e. system resources allocated for the sockets are released.

Datagram Socket
A datagram socket is the sending or receiving point for a packet delivery service. Each packet sent or received on a datagram socket is individually addressed and routed. Multiple packets sent from one machine to another may be routed differently, and may arrive in any order.
Where possible, a newly constructed DatagramSocket has the SO_BROADCAST socket option enabled so as to allow the transmission of broadcast datagrams. In order to receive broadcast packets a DatagramSocket should be bound to the wildcard address. In some implementations, broadcast packets may also be received when a DatagramSocket is bound to a more specific address.

Example: DatagramSocket s = new DatagramSocket(null); s.bind(new InetSocketAddress(8888)); Which is equivalent to: DatagramSocket s = new DatagramSocket(8888); Both cases will create a DatagramSocket able to receive broadcasts on UDP port 8888.

Constructs of Datagram Socket
DatagramSocket()
Constructs a datagram socket and binds it to any available port on the local host machine.
DatagramSocket(DatagramSocketImpl impl)
Creates an unbound datagram socket with the specified DatagramSocketImpl.
DatagramSocket(int port)
Constructs a datagram socket and binds it to the specified port on the local host machine.
DatagramSocket(int port, InetAddress laddr)
Creates a datagram socket, bound to the specified local address.

DatagramSocket(SocketAddress bindaddr)
Creates a datagram socket, bound to the specified local socket address.

void **bind(SocketAddress addr):**Binds this DatagramSocket to a specific address & port.
void **close():**Closes this datagram socket.
void **connect(InetAddress address, int port):**Connects the socket to a remote address for this socket.
void **connect(SocketAddress addr):**Connects this socket to a remote socket address (IP address + port number).
void **disconnect():**Disconnects the socket.
boolean **getBroadcast():**Tests if SO_BROADCAST is enabled.
DatagramChannel **getChannel():**Returns the unique DatagramChannel object associated with this datagram socket, if any.
InetAddress **getInetAddress():**Returns the address to which this socket is connected.
InetAddress **getLocalAddress():**Gets the local address to which the socket is bound.
int **getLocalPort():**Returns the port number on the local host to which this socket is bound.
SocketAddress **getLocalSocketAddress():**Returns the address of the endpoint this socket is bound to.
int **getPort():**Returns the port number to which this socket is connected.
int **getReceiveBufferSize():**Get value of the SO_RCVBUF option for this DatagramSocket, that is the buffer size used by the platform for input on this DatagramSocket.
SocketAddress **getRemoteSocketAddress():**Returns the address of the endpoint this socket is connected to, or null if it is unconnected.
boolean **getReuseAddress():**Tests if SO_REUSEADDR is enabled.
int **getSendBufferSize():**Get value of the SO_SNDBUF option for this DatagramSocket, that is the buffer size used by the platform for output on this DatagramSocket.
int **getSoTimeout():**Retrieve setting for SO_TIMEOUT.
int **getTrafficClass():**Gets traffic class or type-of-service in the IP datagram header for packets sent from this DatagramSocket.
boolean **isBound():**Returns the binding state of the socket.
boolean **isClosed():**Returns whether the socket is closed or not.
boolean **isConnected():**Returns the connection state of the socket.
void **receive(DatagramPacket p):**Receives a datagram packet from this socket.
void **send(DatagramPacket p):**Sends a datagram packet from this socket.
void **setBroadcast(boolean on):**Enable/disable SO_BROADCAST.
static void **setDatagramSocketImplFactory(DatagramSocketImplFactory fac)**:Sets the datagram socket implementation factory for the application.

void **setReceiveBufferSize(int size):**Sets the SO_RCVBUF option to the specified value for this DatagramSocket.
void **setReuseAddress(boolean on):**Enable/disable the SO_REUSEADDR socket option.
void **setSendBufferSize(int size):**Sets the SO_SNDBUF option to the specified value for this DatagramSocket.
void **setSoTimeout(int timeout):**Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds.
void **setTrafficClass(int tc):**Sets traffic class or type-of-service octet in the IP datagram header for datagrams sent from this DatagramSocket.

Datagram packets
Datagram packets are used to implement a connectionless packet delivery service. Each message is routed from one machine to another based solely on information contained within that packet. Multiple packets sent from one machine to another might be routed differently, and might arrive in any order. Packet delivery is not guaranteed.

DatagramPacket(byte[] buf, int length)
Constructs a DatagramPacket for receiving packets of length length.
DatagramPacket(byte[] buf, int length, InetAddress address, int port)
Constructs a datagram packet for sending packets of length length to the specified port number on the specified host.
DatagramPacket(byte[] buf, int offset, int length)
Constructs a DatagramPacket for receiving packets of length length, specifying an offset into the buffer.
DatagramPacket(byte[] buf, int offset, int length, InetAddress address, int port)
Constructs a datagram packet for sending packets of length length with offset ioffsetto the specified port number on the specified host.
DatagramPacket(byte[] buf, int offset, int length, SocketAddress address)
Constructs a datagram packet for sending packets of length length with offset ioffsetto the specified port number on the specified host.
DatagramPacket(byte[] buf, int length, SocketAddress address)
Constructs a datagram packet for sending packets of length length to the specified port number on the specified host.

InetAddress **getAddress():**Returns the IP address of the machine to which this datagram is being sent or from which the datagram was received.
byte[] **getData():**Returns the data buffer.
int **getLength():**Returns the length of the data to be sent or the length of the data received.
int **getOffset():**Returns the offset of the data to be sent or the offset of the data received.
int **getPort():**Returns the port number on the remote host to which this datagram is being sent or from which the datagram was received.
SocketAddress **getSocketAddress():**Gets the SocketAddress (usually IP address + port number) of the remote host that this packet is being sent to or is coming from.

void **setAddress(InetAddress iaddr):**Sets the IP address of the machine to which this datagram is being sent.
void **setData(byte[] buf):**Set the data buffer for this packet.
void **setData(byte[] buf, int offset, int length):**Set the data buffer for this packet.
void **setLength(int length):**Set the length for this packet.
void **setPort(int iport):**Sets the port number on the remote host to which this datagram is being sent.
void **setSocketAddress(SocketAddress address):**Sets the SocketAddress (usually IP address + port number) of the remote host to which this datagram is being sent.

Practical - 2 JDBC Programming

1. Implement the given code and see the output of program.

```
//SET PATH=F:\SOFT\JDK 7\bin
import java.sql.*;
public class Demo1 {
      public static void main(String[] args) {
          try{
                  // Load and register the driver
                  try {
                              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                  }
                  catch(Exception e) {
                      e.printStackTrace();
                  }
                       //Driver myDriver = new sun.jdbc.odbc.JdbcOdbcDriver();
                       //DriverManager.registerDriver(myDriver);

                       // Establish the connection to the database server
                       // urlstring,enter workspace username or
SYSTEM,password
                       Connection cn =
DriverManager.getConnection("jdbc:odbc:CollegeekDSN","admin","admin");

                       // Create a statement
                       Statement st = cn.createStatement();

                       // Execute the statement
                       ResultSet rs = st.executeQuery("select * from
Student");
                  // Retrieve the results
                       while(rs.next())
                              System.out.println(rs.getString(1)+"
"+rs.getString(2));

                       // Close the statement and connection
                       st.close();
                       cn.close();
            }
            catch(SQLException e) {
                       System.out.println(e.getMessage());
            }

        }
    }
```

```
E:\javaperformance\collage\SEM6>java Demo1
180170107046 Kava
180170107033 Gohil
180170107077 Patel
180170107080 Patel
180170107027 Dodiya
180170107048 Kosrekar
180170107031 Gajjar
180170107003 Asodariya
180170107053 Makwana
180170107028 Doshi
180170107060 Panchal
180170107050 Lad
180170107030 Gabu
```

Fig2.1 – Type-1 JDBC Driver

2. Create an application to fill student registration form and submit data into table of Oracle/MS ACCESS. (use JDBC)

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import java.util.Scanner;


class Student_Reg{
        public static void main(String []args){
                String url = "jdbc:oracle:thin:@localhost:1521:XE";
                String uname = "admin";
                String pass = "admin";
                String query = "SELECT * FROM Student";

                try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
                catch(Exception e) {
                        e.printStackTrace();
                }

                try(Scanner scan = new Scanner(System.in);
```

```
                        Connection con =
DriverManager.getConnection(url,uname,pass);){

                        System.out.println("----> STUDENT REGISTRATION <----");
                        System.out.println();
                        String FirstName,LastName,Gender,City;
                        long EnNo;
                        System.out.print("Enter FirstName: ");
                        FirstName = scan.nextLine();
                        System.out.print("Enter LastName: ");
                        LastName = scan.nextLine();
                        System.out.print("Enter Enrollment: ");
                        EnNo = scan.nextLong();
                        System.out.println("1 Male");
                        System.out.println("2 Female");
                        System.out.println("3 Other");
                        System.out.print("Select Gender 1/2/3 ?: ");
                        int option = scan.nextInt();
                        switch(option){
                                case 1:Gender="Male";break;
                                case 2:Gender="Female";break;
                                case 3:Gender="Other";break;
                                default:
                                        Gender="Other";
                        }
                        System.out.println("1 Ahmedabad");
                        System.out.println("2 Bhavnagar");
                        System.out.println("3 Gandhinagar");
                        System.out.println("4 Khambhat");
                        System.out.println("5 Rajkot");
                        System.out.println("6 Surat");
                        System.out.println("7 Surendranagar");
                        System.out.println("8 Valsad");
                        System.out.print("Select City [1-8] ?: ");
                        option = scan.nextInt();
                        switch(option){
                                case 1:City="Ahmedabad";break;
                                case 2:City="Bhavnagar";break;
                                case 3:City="Gandhinagar";break;
                                case 4:City="Khambhat";break;
                                case 5:City="Rajkot";break;
                                case 6:City="Surat";break;
                                case 7:City="Surendranagar";break;
                                case 8:City="Valsad";break;
                                default:
                                        City="Ahmedabad";
                        }
                        System.out.println("Your details are:");
                        System.out.println(EnNo+" "+FirstName+" "+LastName+"
"+Gender+" "+City);

                        PreparedStatement stmt = con.prepareStatement("INSERT
INTO Student VALUES(?,?,?,?,?)");
                        stmt.setLong(1,EnNo);
                        stmt.setString(2,LastName);
                        stmt.setString(3,FirstName);
```

```
                        stmt.setString(4,Gender);
                        stmt.setString(5,City);
                        int n = stmt.executeUpdate();
                        stmt.close();

                        System.out.println(n+" row(s) inserted.");

                        Statement stmt1 = con.createStatement();
                        ResultSet rs = stmt1.executeQuery(query);
                        while(rs.next())
                                System.out.println(rs.getLong("ENNO")+"
"+rs.getString("LASTNAME")+" "+rs.getString(3)+" "+rs.getString(4)+"
"+rs.getString(5));

                        stmt1.close();
                }
                catch(SQLException e){
                        e.printStackTrace();
                }
                catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

```
E:\javaperformance\collage\SEM6>java Student_Reg
----> STUDENT REGISTRATION <----

Enter FirstName: Siddharth
Enter LastName: Gabu
Enter Enrollment: 180170107030
1 Male
2 Female
3 Other
Select Gender 1/2/3 ?: 1
1 Ahmedabad
2 Bhavnagar
3 Gandhinagar
4 Khambhat
5 Rajkot
6 Surat
7 Surendranagar
8 Valsad
Select City [1-8] ?: 7
Your details are:
180170107030 Siddharth Gabu Male Surendranagar
1 row(s) inserted.
```

Fig2.2 – Student Registration from

3. Write an application which list content of table of a database.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.DatabaseMetaData;
import java.util.Scanner;
class Table_content{
        public static void main(String []args){
                String url = "jdbc:oracle:thin:@localhost:1521:XE";
                String uname = "admin";
                String pass = "admin";
                if(args.length==0){
                        System.out.println("Usages: java Table_content
TableName");
                        System.exit(1);
                }
                String query = "select * from "+args[0];
                try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
                catch(Exception e) {
                        e.printStackTrace();
                }
                try(Scanner scan = new Scanner(System.in);
                        Connection con =
DriverManager.getConnection(url,uname,pass);){
                        Statement stmt1 = con.createStatement();
                        ResultSet rs = stmt1.executeQuery(query);
                        ResultSetMetaData rsmd = rs.getMetaData();
                        System.out.println("Total
columns:"+rsmd.getColumnCount());

                        int count = rsmd.getColumnCount();
                        System.out.println("Table       Column      Data Type
Length  Nullable");

                        System.out.printf("%-12s %-12s %-10s %-5d
%6s",args[0].toUpperCase(),rsmd.getColumnName(1),rsmd.getColumnTypeName(1),rs
md.getColumnDisplaySize(1),(rsmd.isNullable(1)==0)?"false":"true");

                        System.out.println();
                        for(int i=1;i<count;++i){
                                System.out.printf("%-12s %-12s %-10s %-5d
%6s","",rsmd.getColumnName(i+1),rsmd.getColumnTypeName(i+1),rsmd.getColumnDis
playSize(i+1),(rsmd.isNullable(i+1)==0)?"false":"true");
                                System.out.println();
                        }
```

```
                        stmt1.close();
                }
                catch(SQLException e){
                        e.printStackTrace();
                }
                catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

```
E:\javaperformance\collage\SEM6>java Table_content Student
Total columns:5
Table        Column       Data Type  Length  Nullable
STUDENT       ENNO          NUMBER     22      false
              LASTNAME      VARCHAR2   255      true
              FIRSTNAME     VARCHAR2   255      true
              GENDER        VARCHAR2   15       true
              CITY          VARCHAR2   255      true

E:\javaperformance\collage\SEM6>java Table_content emp10
Total columns:6
Table        Column       Data Type  Length  Nullable
EMP10         EMAIL         VARCHAR2   255     false
              LASTNAME      VARCHAR2   255      true
              FIRSTNAME     VARCHAR2   255      true
              GENDER        VARCHAR2   15       true
              EXPERIENCE    NUMBER     22       true
              PASSWORD      VARCHAR2   255      true

E:\javaperformance\collage\SEM6>java Table_content emp1000
Total columns:2
Table        Column       Data Type  Length  Nullable
EMP1000       EMPNO         NUMBER     22       true
              SALARY        NUMBER     22       true

E:\javaperformance\collage\SEM6>java Table_content emp100
Total columns:4
Table        Column       Data Type  Length  Nullable
EMP100        EMP_ID        NUMBER     22      false
              EMP_NAME      VARCHAR2   255      true
              JOB_NAME      VARCHAR2   255      true
              SALARY        NUMBER     22       true
```

Fig2.3 – Describe Table

4. Write an application to update content of table. Get values from key board.(Use parameterized query).

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;

import java.util.Scanner;


class Update_Table{
        public static void main(String []args){
                String url = "jdbc:oracle:thin:@localhost:1521:XE";
                String uname = "admin";
                String pass = "admin";
                String query;
                PreparedStatement stmt;
                try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
                catch(Exception e) {
                        e.printStackTrace();
                }

                try(Scanner scan = new Scanner(System.in);
                        Connection con =
DriverManager.getConnection(url,uname,pass);){


                        System.out.println("----> UPDATE INFORMATION <----");
                        System.out.println();
                        System.out.println("NOTE: Everything is Case-
sensitive");
                        System.out.println();
                        System.out.println("what do you want to update?");
                        System.out.println("1 Enrollment No");
                        System.out.println("2 LastName");
                        System.out.println("3 FirstName");
                        System.out.println("4 Gender");
                        System.out.println("5 City");
                        System.out.print("Select Option [1-5] ?: ");
                        String[] data=new String[5];
                        long EnNo=0;
                        String[] column =
{"ENNO","LASTNAME","FIRSTNAME","GENDER","CITY"};
                        int n=0,op,option = scan.nextInt();
                        switch(option){
                                case 1:
```

```
                                                System.out.print("Enter new
Enrollment NO: ");
                                                EnNo = scan.nextLong();
                                                scan.nextLine();
                                                System.out.print("Enter
FirstName: ");
                                                data[2] = scan.nextLine();
                                                System.out.print("Enter
LastName: ");
                                                data[1] = scan.nextLine();
                                                break;
                                case 2:
                                                scan.nextLine();
                                                System.out.print("Enter new
LastName: ");
                                                data[1] = scan.nextLine();
                                                System.out.print("Enter
Enrollment: ");
                                                EnNo = scan.nextLong();
                                                break;
                                case 3:
                                                scan.nextLine();
                                                System.out.print("Enter new
FirstName: ");
                                                data[2] = scan.nextLine();
                                                System.out.print("Enter
Enrollment: ");
                                                EnNo = scan.nextLong();
                                                break;
                                case 4:
                                                System.out.println("1 Male");
                                                System.out.println("2 Female");
                                                System.out.println("3 Other");
                                                System.out.print("Select Gender
1/2/3 ?: ");
                                                op = scan.nextInt();
                                                switch(op){
                                                        case
1:data[3]="Male";break;
                                                        case
2:data[3]="Female";break;
                                                        case
3:data[3]="Other";break;
                                                        default:
                                                                data[3]="Other";
                                                }
                                                System.out.print("Enter
Enrollment: ");
                                                EnNo = scan.nextLong();
                                                break;
                                case 5:
                                                System.out.println("1
Ahmedabad");
                                                System.out.println("2
Bhavnagar");
```

```
                                                        System.out.println("3
Gandhinagar");
                                                        System.out.println("4
Khambhat");
                                                        System.out.println("5 Rajkot");
                                                        System.out.println("6 Surat");
                                                        System.out.println("7
Surendranagar");
                                                        System.out.println("8 Valsad");
                                                        System.out.print("Select City
[1-8] ?: ");
                                                        op = scan.nextInt();
                                                        switch(op){
                                                                case
1:data[4]="Ahmedabad";break;
                                                                case
2:data[4]="Bhavnagar";break;
                                                                case
3:data[4]="Gandhinagar";break;
                                                                case
4:data[4]="Khambhat";break;
                                                                case
5:data[4]="Rajkot";break;
                                                                case
6:data[4]="Surat";break;
                                                                case
7:data[4]="Surendranagar";break;
                                                                case
8:data[4]="Valsad";break;
                                                                default:

        data[4]="Ahmedabad";
                                                        }
                                                        System.out.print("Enter
Enrollment: ");
                                                        EnNo = scan.nextLong();
                        }
                        switch(option){
                                case 1:
                                                        query = "UPDATE Student SET
ENNO=? WHERE LASTNAME=? AND FIRSTNAME=?";
                                                        stmt =
con.prepareStatement(query);
                                                        stmt.setLong(1,EnNo);
                                                        stmt.setString(2,data[1]);
                                                        stmt.setString(3,data[2]);
                                                        n = stmt.executeUpdate();
                                                        stmt.close();
                                                        break;
                                case 2:
                                case 3:
                                case 4:
                                case 5:
                                                        query = "UPDATE Student SET
"+column[option-1]+"=? WHERE ENNO=?";
```

```
                                                stmt =
con.prepareStatement(query);
                                                stmt.setString(1,data[option-
1]);
                                                stmt.setLong(2,EnNo);
                                                n = stmt.executeUpdate();
                                                stmt.close();
                                                break;
                        }
                        System.out.println(n+" row(s) Updated.");
                        if(n>0){
                                        query = "SELECT * FROM Student WHERE
ENNO="+EnNo;
                                        Statement stmt1 =
con.createStatement();
                                        ResultSet rs =
stmt1.executeQuery(query);
                                        while(rs.next())

        System.out.println(rs.getLong("ENNO")+" "+rs.getString("LASTNAME")+"
"+rs.getString(3)+" "+rs.getString(4)+" "+rs.getString(5));

                                        stmt1.close();
                        }
                }
                catch(SQLException e){
                        e.printStackTrace();
                }
                catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

```
E:\javaperformance\collage\SEM6>java Update_Table
----> UPDATE INFORMATION <----

NOTE: Everything is Case-sensitive

what do you want to update?
1 Enrollment No
2 LastName
3 FirstName
4 Gender
5 City
Select Option [1-5] ?: 3
Enter new FirstName: Sidpro
Enter Enrollment: 180170107030
1 row(s) Updated.
180170107030 Gabu Sidpro Male Surendranagar

E:\javaperformance\collage\SEM6>java Update_Table
----> UPDATE INFORMATION <----

NOTE: Everything is Case-sensitive

what do you want to update?
1 Enrollment No
2 LastName
3 FirstName
4 Gender
5 City
Select Option [1-5] ?: 1
Enter new Enrollment NO: 180180180030
Enter FirstName: Sidpro
Enter LastName: Gabu
1 row(s) Updated.
180180180030 Gabu Sidpro Male Surendranagar
```

Fig2.4 - Update content of table

5. Using the JDBC API, display all the records from the database table, selected from command line argument or table selected from combo box.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;

import java.util.Scanner;


class Display_Table{
        public static void main(String []args){
                String url = "jdbc:oracle:thin:@localhost:1521:XE";
                String uname = "admin";
                String pass = "admin";

                if(args.length==0){
                        System.out.println("Usages: java Display_Table
TableName");
                        System.exit(1);
                }
                String query = "SELECT * FROM "+args[0];
                try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
                catch(Exception e) {
                        e.printStackTrace();
                }

                try(Scanner scan = new Scanner(System.in);
                        Connection con =
DriverManager.getConnection(url,uname,pass);){

                        Statement stmt1 = con.createStatement();
                        ResultSet rs = stmt1.executeQuery(query);
                        ResultSetMetaData rsmd = rs.getMetaData();
                        //System.out.println("Total
columns:"+rsmd.getColumnCount());
                        //System.out.println("Column Name of 1st
column:"+rsmd.getColumnName(1));
                        //System.out.println("Column Type Name of 1st
column:"+rsmd.getColumnTypeName(1));
                        int count = rsmd.getColumnCount();
                        while(rs.next()){
                                        for(int i=0;i<count;++i)

        System.out.print(rs.getString(i+1)+" ");

                                        System.out.println();
```

```
                        }

                        stmt1.close();
                }
                catch(SQLException e){
                        e.printStackTrace();
                }
                catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

```
E:\javaperformance\collage\SEM6>java Display_Table emp100
100 Ravi MANAGER 80000
101 Jatin SALESMAN 50000
102 Pratik SALESMAN 50000
103 Puja CLERK 65000
104 Janu ANALYST 70000
105 Garima SALESMAN 50000
```

Fig2.5 – Display Table Records

6. Write a Java application to invoke a stored procedure using a CallableStatement. For this a stored procedure called incrementSalary may be developed to increase all the employee's salary by a percentage specified in the parameter.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.CallableStatement;

import java.util.Scanner;
class Emp_Salary{
        public static void main(String []args){
                String url = "jdbc:oracle:thin:@localhost:1521:XE";
                String uname = "admin";
                String pass = "admin";
                String query="";
                try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
                catch(Exception e) {
                        e.printStackTrace();
                }
```

```
                try(Scanner scan = new Scanner(System.in);
                        Connection con =
DriverManager.getConnection(url,uname,pass);){
                        System.out.println("Incrementing Salary of Employees By
% Percentage");
                        System.out.println();
                        System.out.print("Enter value to increment Salary (E.g.
10): ");
                        int Percentage = scan.nextInt();
                        System.out.println();

                        query = "{ call incrementSalary(?,?) }";
                        CallableStatement cstmt = con.prepareCall(query);
                        cstmt.setInt(1,Percentage);
                        cstmt.registerOutParameter(2,java.sql.Types.INTEGER);
                        cstmt.execute();
                        int n = cstmt.getInt(2);
                        cstmt.close();
                        System.out.println(n+" row(s) Updated.");
                        System.out.println();
                        if(Percentage > -1)System.out.println("After
"+Percentage+"% increment in Salary of Employees");
                        else System.out.println("After "+(Percentage*-1)+"%
Decrement in Salary of Employees");
                        System.out.println();

                        query = "SELECT * FROM Testemp100";
                        Statement stmt = con.createStatement();
                        ResultSet rs = stmt.executeQuery(query);
                        while(rs.next())
                                System.out.println(rs.getInt(1)+"
"+rs.getString(2)+" "+rs.getString(3)+" "+rs.getFloat(4));

                        stmt.close();
                }
                catch(SQLException e){
                        e.printStackTrace();
                }
                catch(Exception e){
                        e.printStackTrace();
                }
        }
}
```

```
E:\javaperformance\collage\SEM6>java Emp_Salary
Incrementing Salary of Employees By % Percentage

Enter value to increment Salary (E.g. 10): 5

6 row(s) Updated.

After 5% increment in Salary of Employees

100 Ravi MANAGER 88516.56
101 Jatin SALESMAN 55322.87
102 Pratik SALESMAN 55322.87
103 Puja CLERK 71919.74
104 Janu ANALYST 77452.03
105 Garima SALESMAN 55322.87

E:\javaperformance\collage\SEM6>java Emp_Salary
Incrementing Salary of Employees By % Percentage

Enter value to increment Salary (E.g. 10): -5

6 row(s) Updated.

After 5% Decrement in Salary of Employees

100 Ravi MANAGER 84090.73
101 Jatin SALESMAN 52556.73
102 Pratik SALESMAN 52556.73
103 Puja CLERK 68323.75
104 Janu ANALYST 73579.43
105 Garima SALESMAN 52556.73
```

Fig2.6 – incrementSalary using a CallableStatement

Answer the following questions:

1. List and explain all four types of JDBC Drivers.

JDBC drivers are client-side adapters (installed on the client machine, not on the server) that convert requests from Java programs to a protocol that the DBMS can understand. There are 4 types of JDBC drivers:

1. Type-1 driver or JDBC-ODBC bridge driver
2. Type-2 driver or Native-API driver
3. Type-3 driver or Network Protocol driver
4. Type-4 driver or Thin driver

Type-1 driver

Type-1 driver or JDBC-ODBC bridge driver uses ODBC driver to connect to the database. The JDBC-ODBC bridge driver converts JDBC method calls into the ODBC function calls. Type-1 driver is also called Universal driver because it can be used to connect to any of the databases.

- As a common driver is used in order to interact with different databases, the data transferred through this driver is not so secured.
- The ODBC bridge driver is needed to be installed in individual client machines.
- Type-1 driver isn't written in java, that's why it isn't a portable driver.
- This driver software is built-in with JDK so no need to install separately.
- It is a database independent driver.

Type-2 driver

The Native API driver uses the client -side libraries of the database. This driver converts JDBC method calls into native calls of the database API. In order to interact with different database, this driver needs their local API, that's why data transfer is much more secure as compared to type-1 driver.

- Driver needs to be installed separately in individual client machines
- The Vendor client library needs to be installed on client machine.
- Type-2 driver isn't written in java, that's why it isn't a portable driver
- It is a database dependent driver.

Type-3 driver

The Network Protocol driver uses middleware (application server) that converts JDBC calls directly or indirectly into the vendor-specific database protocol. Here all the database

connectivity drivers are present in a single server, hence no need of individual client-side installation.

- Type-3 drivers are fully written in Java, hence they are portable drivers.
- No client side library is required because of application server that can perform many tasks like auditing, load balancing, logging etc.
- Network support is required on client machine.
- Maintenance of Network Protocol driver becomes costly because it requires database-specific coding to be done in the middle tier.
- Switch facility to switch over from one database to another database.

Type-4 driver

Type-4 driver is also called native protocol driver. This driver interact directly with database. It does not require any native database library that is why it is also known as Thin Driver.

- Does not require any native library and Middleware server, so no client-side or server-side installation.
- It is fully written in Java language, hence they are portable drivers.

Which Driver to use when?

- If you are accessing one type of database, such as Oracle, Sybase, or IBM, the preferred driver type is type-4.
- If your Java application is accessing multiple types of databases at the same time, type 3 is the preferred driver.
- Type 2 drivers are useful in situations, where a type 3 or type 4 driver is not available yet for your database.
- The type 1 driver is not considered a deployment-level driver, and is typically used for development and testing purposes only.

2. What is parameterised query? How it can be executed in java?

A parameterized query (also known as a prepared statement) is a means of pre-compiling a SQL statement so that all you need to supply are the "parameters" (think "variables") that need to be inserted into the statement for it to be executed. It's commonly used as a means of preventing SQL injection attacks.

In database management systems (DBMS), a prepared statement or parameterized statement is a feature used to execute the same or similar database statements repeatedly with high efficiency. Typically used with SQL statements such as queries or updates, the prepared statement takes the form of a template into which certain constant values are substituted during each execution.

The typical workflow of using a prepared statement is as follows:

i. Prepare: At first, the application creates the statement template and sends it to the DBMS. Certain values are left unspecified, called parameters, placeholders or bind variables (labelled "?" below):

```
INSERT INTO products (name, price) VALUES (?,?);
```

ii. Then, the DBMS compiles (parses, optimizes and translates) the statement template, and stores the result without executing it.

iii. Execute: At a later time, the application supplies (or binds) values for the parameters of the statement template, and the DBMS executes the statement (possibly returning a result). The application may execute the statement as many times as it wants with different values. In the above example, it might initially supply "bike" for the first parameter and "10900" for the second parameter, and then later supply "shoes" for the first parameter and "7400" for the second parameter.

3. Write a note on various APIs of java.sql package.

Provides the API for accessing and processing data stored in a data source (usually a relational database) using the Java™ programming language. This API includes a framework whereby different drivers can be installed dynamically to access different data sources. Although the JDBC™ API is mainly geared to passing SQL statements to a database, it provides for reading and writing data from any data source with a tabular format. The reader/writer facility, available through the `javax.sql.RowSet` group of interfaces, can be customized to use and update data from a spread sheet, flat file, or any other tabular data source.

What the JDBC™ 4.1 API Includes

The JDBC™ 4.1 API includes both the `java.sql` package, referred to as the JDBC core API, and the `javax.sql` package, referred to as the JDBC Optional Package API. This complete JDBC API is included in the Java™ Standard Edition (Java SE™), version 7. The `javax.sql` package extends the functionality of the JDBC API from a client-side API to a server-side API, and it is an essential part of the Java™ Enterprise Edition (Java EE™) technology.

Versions

The JDBC 4.1 API incorporates all of the previous JDBC API versions:

- The JDBC 4.0 API
- The JDBC 3.0 API
- The JDBC 2.1 core API
- The JDBC 2.0 Optional Package API
  (Note that the JDBC 2.1 core API and the JDBC 2.0 Optional Package API together are referred to as the JDBC 2.0 API.)
- The JDBC 1.2 API

- The JDBC 1.0 API

Classes, interfaces, methods, fields, constructors, and exceptions have the following "since" tags that indicate when they were introduced into the Java platform. When these "since" tags are used in Javadoc™ comments for the JDBC API, they indicate the following:

- Since 1.7 -- new in the JDBC 4.1 API and part of the Java SE platform, version 7
- Since 1.6 -- new in the JDBC 4.0 API and part of the Java SE platform, version 6
- Since 1.4 -- new in the JDBC 3.0 API and part of the J2SE platform, version 1.4
- Since 1.2 -- new in the JDBC 2.0 API and part of the J2SE platform, version 1.2
- Since 1.1 or no "since" tag -- in the original JDBC 1.0 API and part of the JDK™, version 1.1

NOTE: Many of the new features are optional; consequently, there is some variation in drivers and the features they support. Always check your driver's documentation to see whether it supports a feature before you try to use it.

NOTE: The class `SQLPermission` was added in the Java™ 2 SDK, Standard Edition, version 1.3 release. This class is used to prevent unauthorized access to the logging stream associated with the `DriverManager`, which may contain information such as table names, column data, and so on.

What the `java.sql` Package Contains

The `java.sql` package contains API for the following:

- Making a connection with a database via the `DriverManager` facility
  - `DriverManager` class -- makes a connection with a driver
  - `SQLPermission` class -- provides permission when code running within a Security Manager, such as an applet, attempts to set up a logging stream through the `DriverManager`
  - `Driver` interface -- provides the API for registering and connecting drivers based on JDBC technology ("JDBC drivers"); generally used only by the `DriverManager` class
  - `DriverPropertyInfo` class -- provides properties for a JDBC driver; not used by the general user
- Sending SQL statements to a database
  - `Statement` -- used to send basic SQL statements
  - `PreparedStatement` -- used to send prepared statements or basic SQL statements (derived from `Statement`)
  - `CallableStatement` -- used to call database stored procedures (derived from `PreparedStatement`)
  - `Connection` interface -- provides methods for creating statements and managing connections and their properties
  - `Savepoint` -- provides savepoints in a transaction
- Retrieving and updating the results of a query

    - `ResultSet` interface
- Standard mappings for SQL types to classes and interfaces in the Java programming language
    - `Array` interface -- mapping for SQL `ARRAY`
    - `Blob` interface -- mapping for SQL `BLOB`
    - `Clob` interface -- mapping for SQL `CLOB`
    - `Date` class -- mapping for SQL `DATE`
    - `NClob` interface -- mapping for SQL `NCLOB`
    - `Ref` interface -- mapping for SQL `REF`
    - `RowId` interface -- mapping for SQL `ROWID`
    - `Struct` interface -- mapping for SQL `STRUCT`
    - `SQLXML` interface -- mapping for SQL `XML`
    - `Time` class -- mapping for SQL `TIME`
    - `Timestamp` class -- mapping for SQL `TIMESTAMP`
    - `Types` class -- provides constants for SQL types
- Custom mapping an SQL user-defined type (UDT) to a class in the Java programming language
    - `SQLData` interface -- specifies the mapping of a UDT to an instance of this class
    - `SQLInput` interface -- provides methods for reading UDT attributes from a stream
    - `SQLOutput` interface -- provides methods for writing UDT attributes back to a stream
- Metadata
    - `DatabaseMetaData` interface -- provides information about the database
    - `ResultSetMetaData` interface -- provides information about the columns of a `ResultSet` object
    - `ParameterMetaData` interface -- provides information about the parameters to `PreparedStatement` commands
- Exceptions
    - `SQLException` -- thrown by most methods when there is a problem accessing data and by some methods for other reasons
    - `SQLWarning` -- thrown to indicate a warning
    - `DataTruncation` -- thrown to indicate that data may have been truncated
    - `BatchUpdateException` -- thrown to indicate that not all commands in a batch update executed successfully

Practical – 3 Servlet

1. A) Develop server side application using servlet for collecting employee information, validation the existing record of employee and retrieval of employee's records. Develop 3 separate servlets for collection, validation and for retrieval.

*****Servlet for collection*****

```
import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;

// First method of creating servlet
public class Collect_Info extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Collecting Employee information";
        }

        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                //String ppath = request.getContextPath();
                String fname ="",lname="",pass="",email="";
                if(request.getAttribute("fname_error")!=null){
                        fname=(String)request.getAttribute("fname_error");
                        request.removeAttribute("fname_error");
                }
                if(request.getAttribute("lname_error")!=null){
                        lname=(String)request.getAttribute("lname_error");
                        request.removeAttribute("lname_error");
                }
                if(request.getAttribute("email_error")!=null){
                        email=(String)request.getAttribute("email_error");
                        request.removeAttribute("email_error");
                }
                if(request.getAttribute("pass_error")!=null){
                        pass=(String)request.getAttribute("pass_error");
                        request.removeAttribute("pass_error");
                }

                out.println("<!DOCTYPE html>");
                out.println("<html>");

                out.println("<head>");
```

```
                        out.println("<title> Servlet By Sid </title>");
                        out.println("<meta charset=\"utf-8\">");
                        out.println("<meta name=\"author\"
content=\"SidPro\"/>");
                        out.println("<meta name=\"viewport\"
content=\"width=device-width,initial-scale=1\">");
                        out.println("<link type=\"text/css\" rel=\"stylesheet\"
href=\"Collect_Info.css\">");
                out.println("</head>");

                out.println("<body>");
                        out.println("<h2> "+message+"</h2>");
                        //out.println("<h3>Myservlet extends
HttpServlet</h3>");

                out.println("<form action=\"Validation_Info\" method=\"post\"
class=\"main_container\">");
                out.println("<div class=\"containerbase\">");
                        out.println("<lable for=\"firstname\">Enter First Name
:"+fname+"</lable><br>");
                        out.println("<input type=\"text\" name=\"fname\"
id=\"firstname\" placeholder=\"Enter your first name here\"><br>");
                        out.println("<lable for=\"lastname\">Enter Last Name
:"+lname+"</lable><br>");
                        out.println("<input type=\"text\" name=\"lname\"
id=\"lastname\" placeholder=\"Enter your last name here\"><br>");
                        out.println("<lable for=\"Email\">Enter
Email:"+email+"</lable><br>");
                        out.println("<input type=\"email\" name=\"email\"
id=\"Email\" placeholder=\"Enter your email address here\"><br>");
                        out.println("<lable for=\"exp\">Employee experience:
</lable>");
                        out.println("<select name=\"exp\" id=\"exp\">");
                                out.println("<option value=\"1\">1</option>");
                                out.println("<option value=\"2\">2</option>");
                                out.println("<option value=\"3\">3</option>");
                                out.println("<option value=\"4\"
selected>4</option>");
                                out.println("<option value=\"5\">5</option>");
                                out.println("<option value=\"6\">6</option>");
                                out.println("<option value=\"7\">7</option>");
                                out.println("<option value=\"8\">8</option>");
                        out.println("</select><br>");
                        out.println("<lable>Select Gender:</lable><br>");
                        out.println("<input type=\"radio\" name=\"gender\"
value=\"Male\" checked>Male");
                        out.println("<input type=\"radio\" name=\"gender\"
value=\"Female\">Female");
                        out.println("<input type=\"radio\" name=\"gender\"
value=\"other\">other<br>");

                        out.println("<lable for=\"Pass\">Enter
Password:"+pass+"</lable><br>");
                        out.println("<input type=\"password\" name=\"pass\"
id=\"Pass\" placeholder=\"Enter your password here\"><br>");
                        out.println("<div class=\"container\">");
```

```
                        out.println("<input type=\"submit\"
value=\"Submit\"><input type=\"reset\" value=\"Reset\">");
                        out.println("</div>");
                out.println("</div>");
                out.println("</form>");

                out.println("<section>");
                        out.println("<div class=\"wave wave1\"></div>");
                        out.println("<div class=\"wave wave2\"></div>");
                        out.println("<div class=\"wave wave3\"></div>");
                out.println("</section>");
                out.println("</body>");
                out.println("</html>");

        }

}
```

*****Servlet for validation*****

```
import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.ServletException;
import javax.servlet.ServletConfig;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import java.util.regex.Pattern;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;


// First method of creating servlet
public class Validation_Info extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Validating Employee information";
        }

        public void doPost(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                //String ppath = request.getContextPath();
                String fname =
((request.getParameter("fname")!=null)?request.getParameter("fname"):"");
```

```
                String lname =
(request.getParameter("lname")==null)?"":request.getParameter("lname");
                String email =
(request.getParameter("email")==null)?"":request.getParameter("email");
                String pass =
(request.getParameter("pass")==null)?"":request.getParameter("pass");
                String gender = request.getParameter("gender");
                int years = Integer.valueOf(request.getParameter("exp"));
                boolean go = true;

                        if(fname.equals("")){
                                go=false;
                                request.setAttribute("fname_error", "<span
style=\"color:red;\"> * First Name is Required!</span>");
                        }
                        else{
                                if(!Pattern.matches("^[a-zA-Z]*$",fname)){
                                        go=false;
                                        request.setAttribute("fname_error",
"<span style=\"color:red;\"> * Only letters and white space
allowed!</span>");
                                }
                        }
                        if(lname.equals("")){
                                go=false;
                                request.setAttribute("lname_error", "<span
style=\"color:red;\"> * Last Name is Required!</span>");
                        }
                        else{
                                if(!Pattern.matches("^[a-zA-Z]*$",lname)){
                                        go=false;
                                        request.setAttribute("lname_error",
"<span style=\"color:red;\"> * Only letters and white space
allowed!</span>");
                                }
                        }
                        if(pass.equals("")){
                                go=false;
                                request.setAttribute("pass_error", "<span
style=\"color:red;\"> * Password is Required!</span>");
                        }else{
                                if(pass.length()<8){
                                        go=false;
                                        request.setAttribute("pass_error",
"<span style=\"color:red;\"> * Password length must be 8 !</span>");
                                }
                        }
                        if(email.equals("")){
                                go=false;
                                request.setAttribute("email_error", "<span
style=\"color:red;\"> * Email is Required!</span>");
                        }
                        else{
                                Pattern ptr = Pattern.compile("^[A-Z0-9._%+-
]+@[A-Z0-9.-]+\\.[A-Z]{2,6}$",Pattern.CASE_INSENSITIVE);
                                if(!ptr.matcher(email).matches()){
```

```
                                        go=false;
                                        request.setAttribute("email_error",
"<span style=\"color:red;\"> * Invalid email format!</span>");
                                }
                        }

                        if(go){

                                ServletConfig config=getServletConfig();
                                String classname =
config.getInitParameter("classname");
                                String url =  config.getInitParameter("url");
                                String uname =
config.getInitParameter("username");
                                String password =
config.getInitParameter("password");

                                try {
                                        Class.forName(classname);
                                }catch(Exception e) {e.printStackTrace();}

                                try(Connection con =
DriverManager.getConnection(url,uname,password);){

                                        PreparedStatement stmt =
con.prepareStatement("INSERT INTO Emp10 VALUES(?,?,?,?,?,?)");
                                        stmt.setString(1,email);
                                        stmt.setString(2,lname);
                                        stmt.setString(3,fname);
                                        stmt.setString(4,gender);
                                        stmt.setInt(5,years);
                                        stmt.setString(6,pass);
                                        int n = stmt.executeUpdate();
                                        stmt.close();
                                        out.println("<p>"+n+" row(s)
inserted.</p>");
                                        response.sendRedirect("Display_Info");
                                        //System.out.println(n+" row(s)
inserted.");

                                }
                                catch(SQLException e){

                                        out.println("<p>"+e+"</p>");
                                }
                                catch(Exception e){

                                        out.println("<p>"+e+"</p>");
                                }


                        }else{
                                RequestDispatcher
rd=request.getRequestDispatcher("Collect_Info");
                                //out.println("<p>First name:
"+request.getParameter("fname")+"</p>");
```

```
                                rd.include(request, response);
                        }

        }

}
```

*****Servlet for retrieval*****

```
import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.ServletException;
import javax.servlet.ServletConfig;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import java.util.regex.Pattern;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

// First method of creating servlet
public class Display_Info extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Displaying Employee information";
        }

        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                //String ppath = request.getContextPath();


                out.println("<!DOCTYPE html>");
                out.println("<html>");

                out.println("<head>");
                        out.println("<title> Servlet By Sid </title>");
                        out.println("<meta charset=\"utf-8\">");
                        out.println("<meta name=\"author\"
content=\"SidPro\"/>");
                        out.println("<meta name=\"viewport\"
content=\"width=device-width,initial-scale=1\">");
```

```
                        out.println("<link type=\"text/css\" rel=\"stylesheet\"
href=\"Display_Info.css\">");
                out.println("</head>");

                out.println("<body>");
                        out.println("<h2> "+message+"<a
href=\"/Sidpro/Collect_Info\" target=\"_self\"> Register Here</a></h2>");
                        //out.println("<h3>Myservlet extends
HttpServlet</h3>");
                        ServletConfig config=getServletConfig();
                                String classname =
config.getInitParameter("classname");
                                String url =  config.getInitParameter("url");
                                String uname =
config.getInitParameter("username");
                                String password =
config.getInitParameter("password");

                                try {
                                        Class.forName(classname);
                                }catch(Exception e) {e.printStackTrace();}

                                try(Connection con =
DriverManager.getConnection(url,uname,password);){


                                        Statement stmt1 =
con.createStatement();
                                        ResultSet rs =
stmt1.executeQuery("SELECT * FROM emp10");
                                        out.println("<table>");

        out.println("<tr><td>EMAIL</th><th>LASTNAME</th><th>FIRSTNAME</th><th>
GENDER</td><td>EXPERIENCE</td><td>PASSWORD</td></tr>");
                                        while(rs.next()){
                                                out.println("<tr>");

        out.println("<td>"+rs.getString(1)+"</td>"+"<td>"+rs.getString(2)+"</t
d>"+"<td>"+rs.getString(3)+"</td>"+"<td>"+rs.getString(4)+"</td>"+"<td>"+rs.g
etString(5)+"</td>"+"<td>"+rs.getString(6)+"</td>");
                                                out.println("</tr>");
                                        }
                                        stmt1.close();
                                        out.println("</table>");
                                }
                                catch(SQLException e){

                                        out.println("<p>"+e+"</p>");
                                }
                                catch(Exception e){

                                        out.println("<p>"+e+"</p>");
                                }

                out.println("</body>");
                out.println("</html>");
```

```
        }

}
```

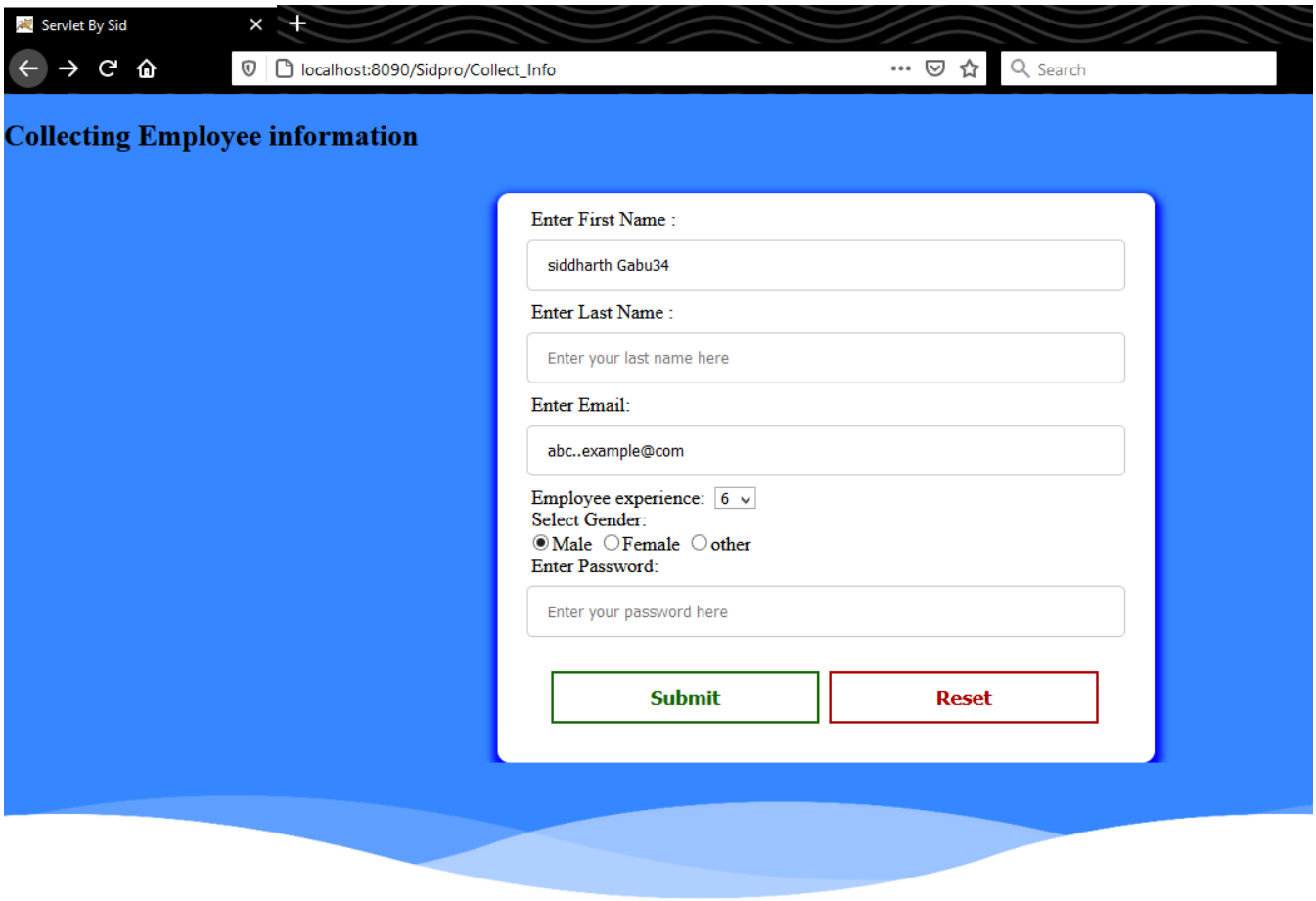


Fig3.1a – Collecting Employee Information

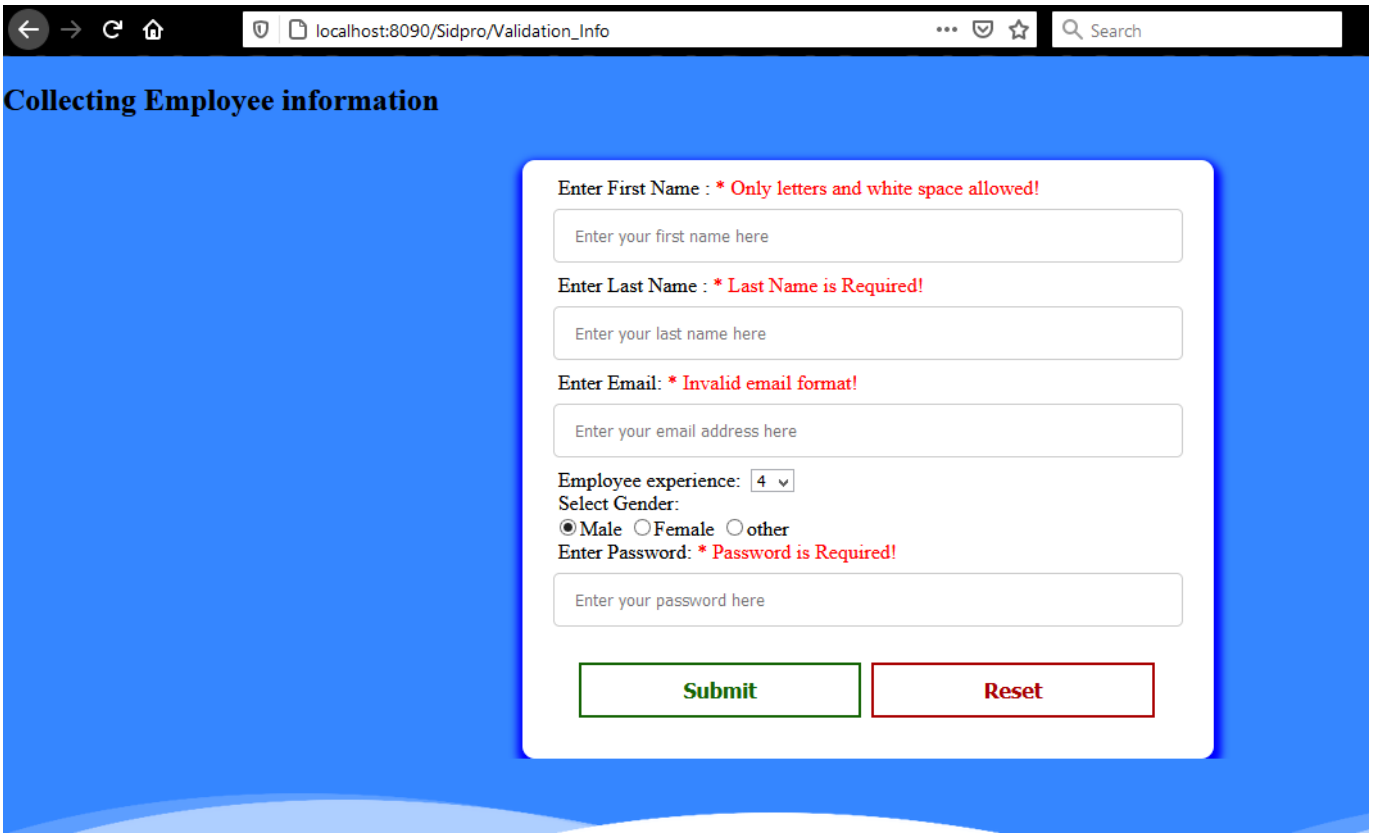
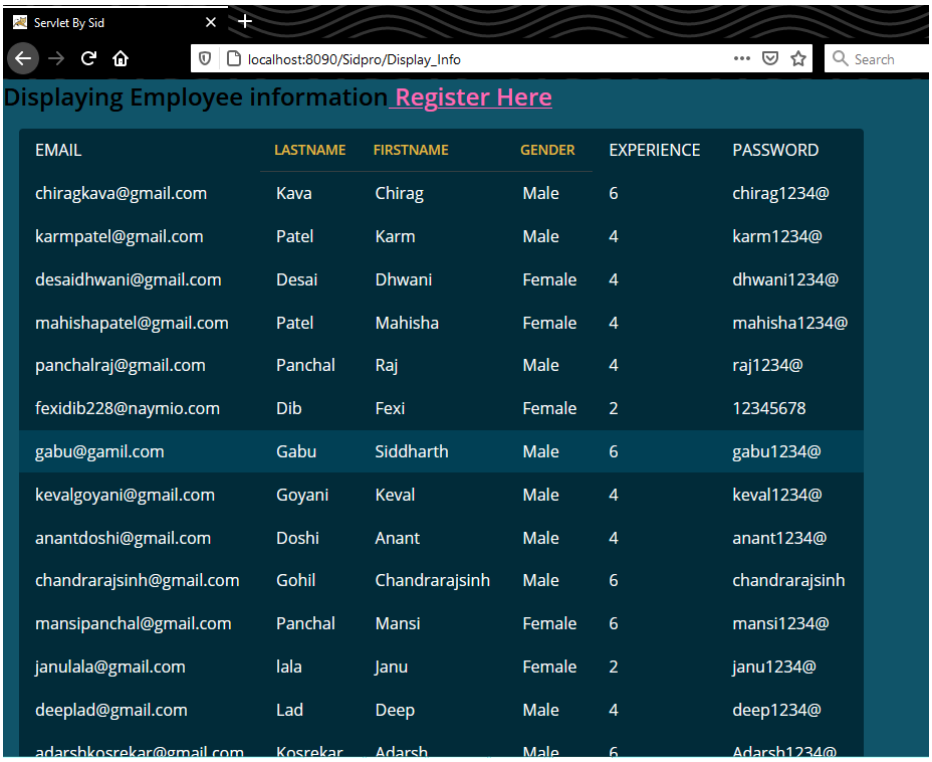


Fig3.1b – Validating Employee Information



EMAIL	LASTNAME	FIRSTNAME	GENDER	EXPERIENCE	PASSWORD
chiragkava@gmail.com	Kava	Chirag	Male	6	chirag1234@
karmpatel@gmail.com	Patel	Karm	Male	4	karm1234@
desaidhwani@gmail.com	Desai	Dhwani	Female	4	dhwani1234@
mahishapatel@gmail.com	Patel	Mahisha	Female	4	mahisha1234@
panchalraj@gmail.com	Panchal	Raj	Male	4	raj1234@
fexidib228@naymio.com	Dib	Fexi	Female	2	12345678
gabu@gamil.com	Gabu	Siddharth	Male	6	gabu1234@
kevalgoyani@gmail.com	Goyani	Keval	Male	4	keval1234@
anantdoshi@gmail.com	Doshi	Anant	Male	4	anant1234@
chandrarajsinh@gmail.com	Gohil	Chandrarajsinh	Male	6	chandrarajsinh
mansipanchal@gmail.com	Panchal	Mansi	Female	6	mansi1234@
janulala@gmail.com	lala	Janu	Female	2	janu1234@
deeplad@gmail.com	Lad	Deep	Male	4	deep1234@
adarshkosrekar@gmail.com	Kosrekar	Adarsh	Male	6	Adarsh1234@

Fig3.1c – Displaying Employee Information

1. B) Develop a client side web base form or applet for the server side employee information as in a).

*****Client side HTML for collection*****

```
<!DOCTYPE html>
<html>

<head>
        <title> Servlet By Sid </title>
        <meta charset="utf-8">
        <meta name="author" content="SidPro"/>
        <meta name="viewport" content="width=device-
width,initial-scale=1">
        <link type="text/css" rel="stylesheet"
href="Collect_Info.css">
        <style>
          input[type="button"]{
               padding: 10px 14px;
               margin: 8px 4px;
               display:inline-block;
               width:48%;
               font-size: 16px;
               color:white;
               font-weight: bold;
          }

          input[type="button"]{
               border:2px solid rgb(20, 100, 4);
               color:rgb(20, 100, 4);
               background-color: white;
          }
          input[type="button"]:hover{
               background-color: rgb(20, 100, 4);
               cursor: pointer;
               color:white;
          }
          @media only screen and (max-width:600px) {
               input[type="button"]{
                 width: 100%;
                 padding: 6px 8px;
                 margin: 6px 2px;
                 border-radius: 5px;
               }
          }
        </style>
</head>

<body>

       <h2>Collecting Client Side Employee information Of
Collegeek</h2>

<form method="post" class="main_container">
<div class="containerbase">
```

```
                        <lable id="F" for="firstname">Enter First Name
:</lable><br>
                        <input type="text" name="fname" id="firstname"
placeholder="Enter your first name here"><br>
                        <lable id="L" for="lastname">Enter Last Name
:</lable><br>
                        <input type="text" name="lname" id="lastname"
placeholder="Enter your last name here"><br>
                        <lable id="E" for="Email">Enter Email:</lable><br>
                        <input type="email" name="email" id="Email"
placeholder="Enter your email address here"><br>
                        <lable for="exp">Employee experience: </lable>
                        <select name="exp" id="exp">
                                <option value="1">1</option>
                                <option value="2">2</option>
                                <option value="3">3</option>
                                <option value="4" selected>4</option>
                                <option value="5">5</option>
                                <option value="6">6</option>
                                <option value="7">7</option>
                                <option value="8">8</option>
                        </select><br>
                        <lable>Select Gender:</lable><br>
                        <input type="radio" name="gender" value="Male"
checked>Male
                        <input type="radio" name="gender"
value="Female">Female
                        <input type="radio" name="gender"
value="other">other<br>

                        <lable id="P" for="Pass">Enter Password:</lable><br>
                        <input type="password" name="pass" id="Pass"
placeholder="Enter your password here"><br>
                        <div class="container">
                        <input type="button" value="Submit"
onclick="sendData()"><input type="reset" value="Reset">
                        </div>
                </div>
                </form>

                <section>
                        <div class="wave wave1"></div>
                        <div class="wave wave2"></div>
                        <div class="wave wave3"></div>
                </section>
                </body>
                <script>
                function sendData(){

                        var fname = document.getElementById('firstname').value;
                        var lname = document.getElementById('lastname').value;
                        var email = document.getElementById('Email').value;
                        var exp = document.getElementById('exp').value;
                        var gender =
document.querySelector('input[name=gender]:checked').value;
                        var pass = document.getElementById('Pass').value;
```

```
                        var Data = "?fname="+encodeURIComponent(fname)

        +"&lname="+encodeURIComponent(lname)

        +"&email="+encodeURIComponent(email)
                                                +"&exp="+encodeURIComponent(exp)

        +"&gender="+encodeURIComponent(gender)

        +"&pass="+encodeURIComponent(pass);

                        var xmlhttp = new XMLHttpRequest();
                        xmlhttp.responseType = 'json';
                        xmlhttp.onreadystatechange = function() {

                                if(this.readyState == 4 && this.status == 200)
{

                                        console.log(this.response);
                                        var arr = this.response;
                                        if(arr==null){
                                                console.log("heheh");
                                                window.location='Display_Info';
                                        }
                                        document.getElementById('F').innerHTML=
"Enter First Name :<span style='color:red'>"+arr[0].fname_error+"</span>";
                                        document.getElementById('L').innerHTML=
"Enter Last Name :<span style='color:red'>"+arr[1].lname_error+"</span>";
                                        document.getElementById('P').innerHTML=
"Enter Password:<span style='color:red'>"+arr[2].pass_error+"</span>";
                                        document.getElementById('E').innerHTML=
"Enter Email:<span style='color:red'>"+arr[3].email_error+"</span>";
                                }
                        };
                          xmlhttp.open("POST", "Client_side_validate"+Data,
true);
                          xmlhttp.setRequestHeader('Content-
Type','application/x-www-form-urlencoded' );
                          xmlhttp.send();
                }

                 </script>
                 </html>
```

*****Servlet for client side validation*****

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.ServletContext;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
import javax.servlet.RequestDispatcher;
import java.util.regex.Pattern;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;


// First method of creating servlet
public class Client_side_validate extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Validating Employee information";
        }

        public void doPost(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html;charset=UTF-8");


                PrintWriter out = response.getWriter();

                        //String ppath = request.getContextPath();
                        String fname =
(request.getParameter("fname")==null)?"":request.getParameter("fname");
                        String lname =
(request.getParameter("lname")==null)?"":request.getParameter("lname");
                        String email =
(request.getParameter("email")==null)?"":request.getParameter("email");
                        String pass =
(request.getParameter("pass")==null)?"":request.getParameter("pass");
                        String gender = request.getParameter("gender");
                        String
fname_error="",lname_error="",pass_error="",email_error="";
                        int years =
Integer.valueOf(request.getParameter("exp"));
                        boolean go = true;

                        if(fname.equals("")){
                                go=false;
                                fname_error=" * First Name is Required!";
                        }
                        else{
                                if(!Pattern.matches("^[a-zA-Z]*$",fname)){
                                        go=false;
                                        fname_error=" * Only letters and white
space allowed!";
                                }
                        }
                        if(lname.equals("")){
                                go=false;
                                lname_error=" * Last Name is Required! ";
```

```
                        }
                        else{
                                if(!Pattern.matches("^[a-zA-Z]*$",lname)){
                                        go=false;
                                        lname_error=" * Only letters and white
space allowed!";
                                }
                        }
                        if(pass.equals("")){
                                go=false;
                                pass_error=" * Password is Required!";
                        }else{
                                if(pass.length()<8){
                                        go=false;
                                        pass_error=" * Password length must be
8 !";
                                }
                        }
                        if(email.equals("")){
                                go=false;
                                email_error=" * Email is Required!";
                        }
                        else{
                                Pattern ptr = Pattern.compile("^[A-Z0-9._%+-
]+@[A-Z0-9.-]+\\.[A-Z]{2,6}$",Pattern.CASE_INSENSITIVE);
                                if(!ptr.matcher(email).matches()){
                                        go=false;
                                        email_error=" * Invalid email format!";
                                }
                        }

                        if(go){

                                ServletContext config=getServletContext();
                                String classname =
config.getInitParameter("classname");
                                String url =  config.getInitParameter("url");
                                String uname =
config.getInitParameter("username");
                                String password =
config.getInitParameter("password");

                                try {
                                        Class.forName(classname);
                                }catch(Exception e) {e.printStackTrace();}

                                try(Connection con =
DriverManager.getConnection(url,uname,password);){

                                        PreparedStatement stmt =
con.prepareStatement("INSERT INTO Emp10 VALUES(?,?,?,?,?,?)");
                                        stmt.setString(1,email);
                                        stmt.setString(2,lname);
                                        stmt.setString(3,fname);
                                        stmt.setString(4,gender);
                                        stmt.setInt(5,years);
```

```
                                        stmt.setString(6,pass);
                                        int n = stmt.executeUpdate();
                                        stmt.close();
                                        //out.println("<p>"+n+" row(s)
inserted.</p>");

                                        //System.out.println(n+" row(s)
inserted.");

                                }
                                catch(SQLException e){

                                        out.println("<p>"+e+"</p>");
                                }
                                catch(Exception e){

                                        out.println("<p>"+e+"</p>");
                                }


                        }else{
                                out.println("["
                                +"{\"fname_error\":\""+fname_error+"\"},"
                                +"{\"lname_error\":\""+lname_error+"\"},"
                                +"{\"pass_error\":\""+pass_error+"\"},"
                                +"{\"email_error\":\""+email_error+"\"}]");

                        }



        }

}
```

*****Servlet for retrieval*****

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.ServletConfig;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import java.util.regex.Pattern;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

// First method of creating servlet
public class Display_Info extends HttpServlet{
```

```
        private String message;

        public void init() throws ServletException{
                message = "Displaying Employee information";
        }
        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                //String ppath = request.getContextPath();
                out.println("<!DOCTYPE html>");
                out.println("<html>");

                out.println("<head>");
                        out.println("<title> Servlet By Sid </title>");
                        out.println("<meta charset=\"utf-8\">");
                        out.println("<meta name=\"author\"
content=\"SidPro\"/>");
                        out.println("<meta name=\"viewport\"
content=\"width=device-width,initial-scale=1\">");
                        out.println("<link type=\"text/css\" rel=\"stylesheet\"
href=\"Display_Info.css\">");
                out.println("</head>");
                out.println("<body>");
                        out.println("<h2> "+message+"<a
href=\"/Sidpro/Collect_Info\" target=\"_self\"> Register Here</a></h2>");
                        //out.println("<h3>Myservlet extends
HttpServlet</h3>");
                        ServletConfig config=getServletConfig();
                                String classname =
config.getInitParameter("classname");
                                String url =  config.getInitParameter("url");
                                String uname =
config.getInitParameter("username");
                                String password =
config.getInitParameter("password");
                                try {
                                        Class.forName(classname);
                                }catch(Exception e) {e.printStackTrace();}

                                try(Connection con =
DriverManager.getConnection(url,uname,password);){
                                        Statement stmt1 =
con.createStatement();
                                        ResultSet rs =
stmt1.executeQuery("SELECT * FROM emp10");
                                        out.println("<table>");

        out.println("<tr><td>EMAIL</th><th>LASTNAME</th><th>FIRSTNAME</th><th>
GENDER</td><td>EXPERIENCE</td><td>PASSWORD</td></tr>");
                                        while(rs.next()){
                                                out.println("<tr>");

        out.println("<td>"+rs.getString(1)+"</td>"+"<td>"+rs.getString(2)+"</t
d>"+"<td>"+rs.getString(3)+"</td>"+"<td>"+rs.getString(4)+"</td>"+"<td>"+rs.g
etString(5)+"</td>"+"<td>"+rs.getString(6)+"</td>");
```

```
                                        out.println("</tr>");
                                }
                                stmt1.close();
                                out.println("</table>");
                        }
                        catch(SQLException e){
                                out.println("<p>"+e+"</p>");
                        }
                        catch(Exception e){
                                out.println("<p>"+e+"</p>");
                        }

                out.println("</body>");
                out.println("</html>");
        }
}
```

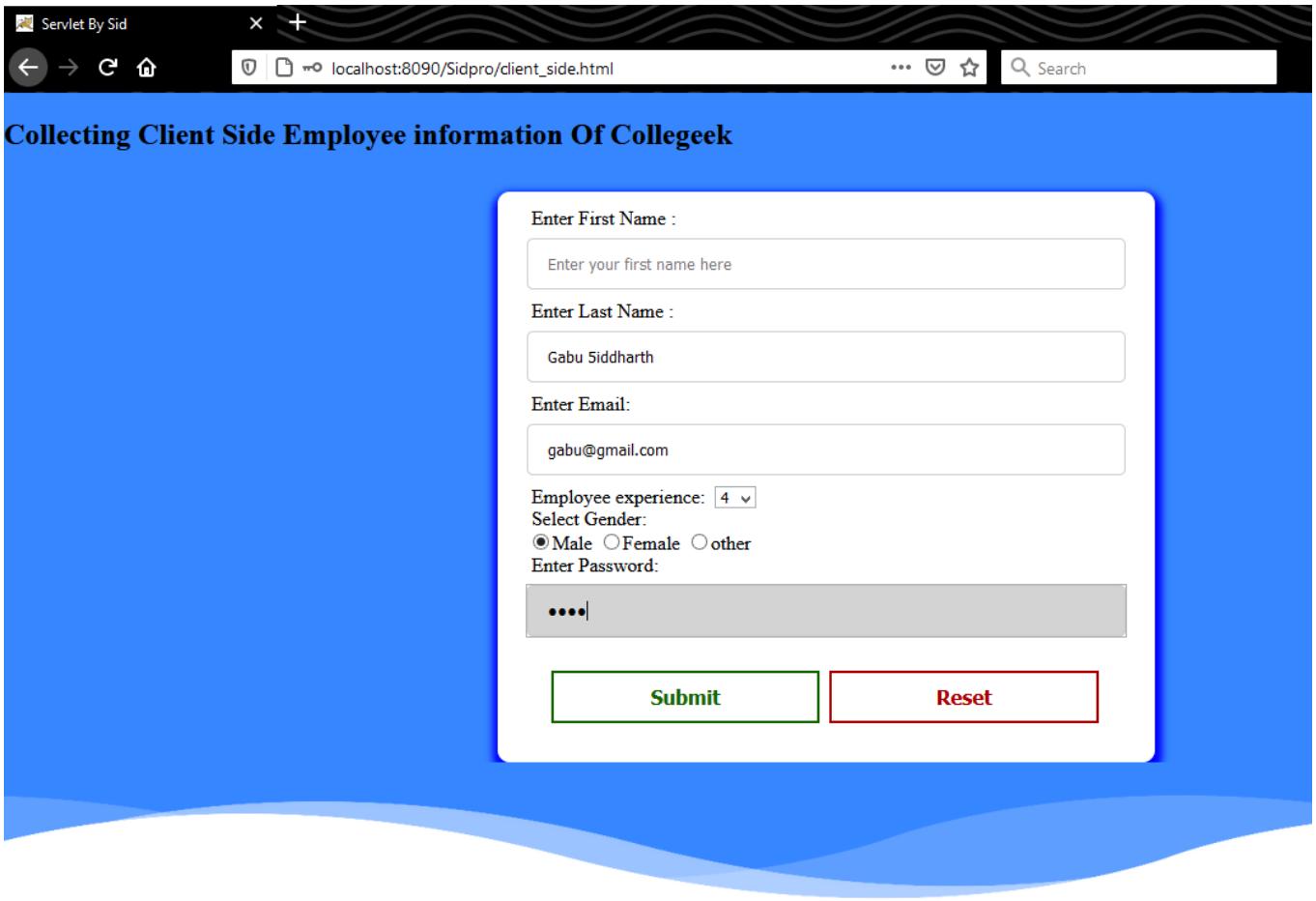


Fig3.2a – Collecting Employee information (Client side form)

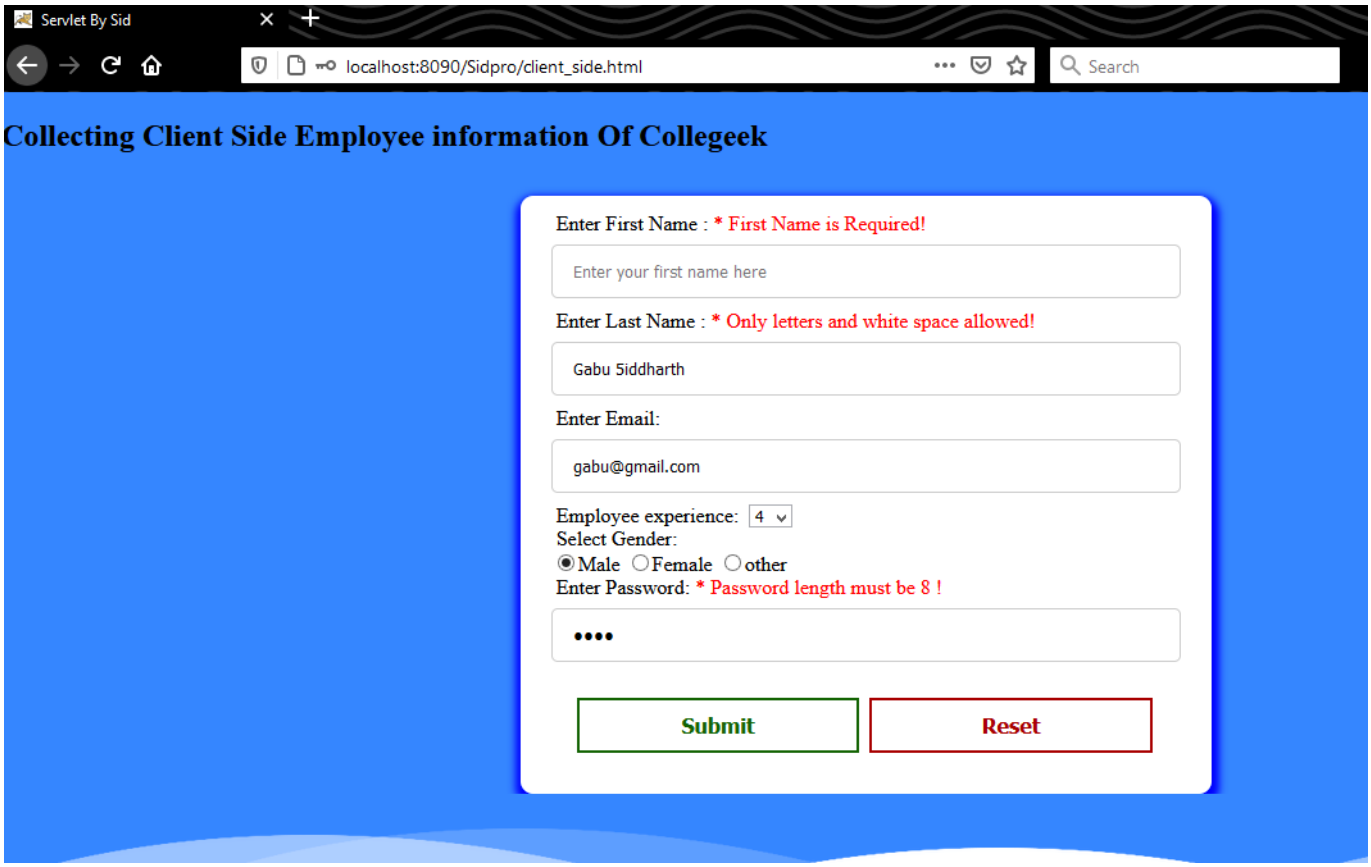
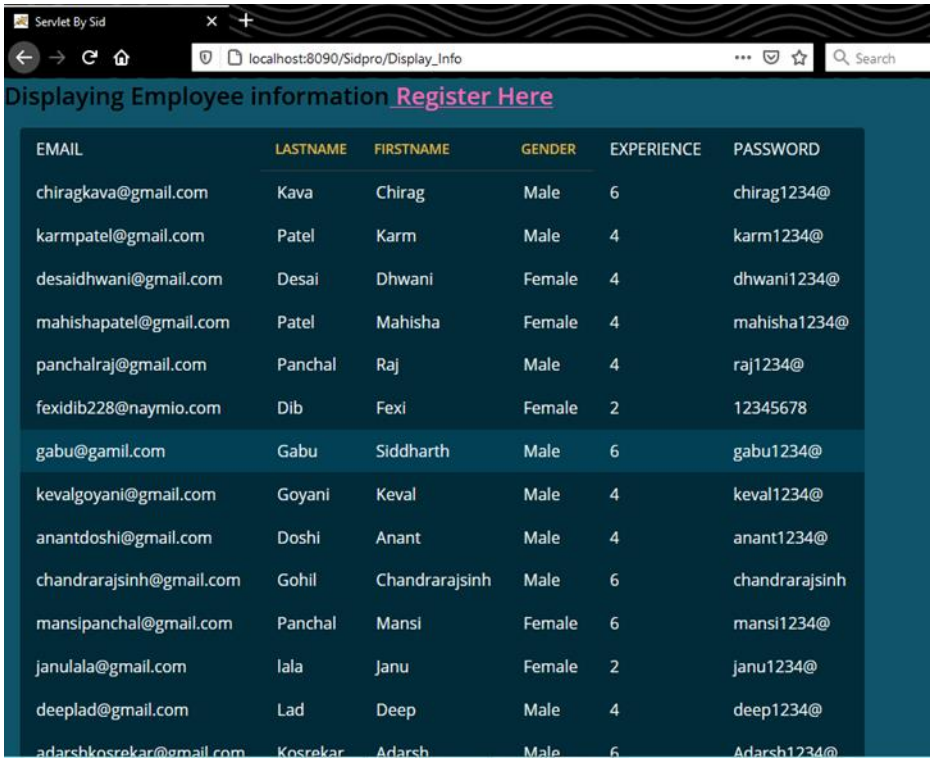


Fig3.2b – Validating Employee Information (Client side form)



EMAIL	LASTNAME	FIRSTNAME	GENDER	EXPERIENCE	PASSWORD
chiragkava@gmail.com	Kava	Chirag	Male	6	chirag1234@
karmpatel@gmail.com	Patel	Karm	Male	4	karm1234@
desaidhwani@gmail.com	Desai	Dhwani	Female	4	dhwani1234@
mahishapatel@gmail.com	Patel	Mahisha	Female	4	mahisha1234@
panchalraj@gmail.com	Panchal	Raj	Male	4	raj1234@
fexidib228@naymio.com	Dib	Fexi	Female	2	12345678
gabu@gamil.com	Gabu	Siddharth	Male	6	gabu1234@
kevalgoyani@gmail.com	Goyani	Keval	Male	4	keval1234@
anantdoshi@gmail.com	Doshi	Anant	Male	4	anant1234@
chandrarajsinh@gmail.com	Gohil	Chandrarajsinh	Male	6	chandrarajsinh
mansipanchal@gmail.com	Panchal	Mansi	Female	6	mansi1234@
janulala@gmail.com	lala	Janu	Female	2	janu1234@
deeplad@gmail.com	Lad	Deep	Male	4	deep1234@
adarshkosrekar@gmail.com	Kosrekar	Adarsh	Male	6	Adarsh1234@

Fig3.2c – Displaying Employee Information

2. For above application implement a login page. A user can use above pages only after login using correct user ID and password. Implement logout facility also. Do session tracking using any one of the four methods discussed in class.

*****Servlet for Two Links*****

```
import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;

// First method of creating servlet
public class Index_Select extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Collecting Employee information";
        }

        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                String username = "";
                HttpSession session=request.getSession(false);
                if(session==null){
                        response.sendRedirect("Login");
                }else{

        username=(String)(session.getAttribute("username")==null?"":session.ge
tAttribute("username"));
                        if(username.equals("")){
                                        response.sendRedirect("Login");
                        }
                }

                out.println("<!DOCTYPE html>");
                out.println("<html>");

                out.println("<head>");
                        out.println("<title> Servlet By Sid </title>");
                        out.println("<meta charset=\"utf-8\">");
                        out.println("<meta name=\"author\"
content=\"SidPro\"/>");
                        out.println("<meta name=\"viewport\"
content=\"width=device-width,initial-scale=1\">");
```

```
                        out.println("<link type=\"text/css\" rel=\"stylesheet\"
href=\"index_select.css\">");
                        out.println("<script src=\"jquery-
3.5.1.min.js\"></script>");
                        out.println("<script
src=\"index_select.js\"></script>");
                        /*out.println("<script>");
                        out.println("function CloseS(){");
                        out.println("window.location.href=\"Logout\";}");
                        out.println("</script>");*/
                out.println("</head>");

                out.println("<body>");
                out.println("<div class=\"topnav\">");
                        out.println("<a class=\"active\"
href=\"#home\">Home</a>");
                        out.println("<a href=\"#About\">About</a>");
                        out.println("<a
href=\"Collect_Info\">Registration</a>");
                        out.println("<a href=\"Logout\">Logout</a>");
                out.println("</div>");
                out.println("<h2 style=\"text-align:center\">"+username+"
Click One of Below Given Link</h2>");
                out.println("<p style=\"text-align:center\"><a
href=\"Collect_Info\">Employee Registration</a></p>");
                out.println("<p style=\"text-align:center\"><a
href=\"Display_Info\">View Existing Records of Employee</a></p>");
                out.println("<button id=\"button4\">Stop</button><br>");
                out.println("<button id=\"button1\">hide links</button><br>");
                out.println("<button id=\"button2\">Play</button><br>");
                out.println("<button id=\"button3\">Boomerang</button>");
                out.println("<div id=\"div1\"
style=\"width:80px;height:80px;display:none;background-
color:yellow\"></div><br>");
                out.println("<div id=\"div2\"
style=\"width:80px;height:80px;display:none;background-
color:red\"></div><br>");
                out.println("<div id=\"div3\"
style=\"width:80px;height:80px;display:none;background-
color:blue;\"></div>");
                out.println("<div id=\"div4\"
style=\"width:80px;height:80px;background-
color:#ccc;position:absolute;\"></div>");

                out.println("</body>");
                out.println("</html>");

        }

}
```

*****Servlet for Displaying Employee Info*****

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.ServletConfig;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import java.util.regex.Pattern;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;

// First method of creating servlet
public class Display_Info extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Displaying Employee information";
        }

        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                //String ppath = request.getContextPath();
                String username = "";
                HttpSession session=request.getSession(false);
                if(session==null){
                        response.sendRedirect("Login");
                }else{


        username=(String)(session.getAttribute("username")==null?"":session.ge
tAttribute("username"));
                        if(username.equals("")){
                                        response.sendRedirect("Login");
                        }
                }

                out.println("<!DOCTYPE html>");
                out.println("<html>");

                out.println("<head>");
                        out.println("<title> Servlet By Sid </title>");
                        out.println("<meta charset=\"utf-8\">");
                        out.println("<meta name=\"author\"
content=\"SidPro\"/>");
                        out.println("<meta name=\"viewport\"
content=\"width=device-width,initial-scale=1\">");
```

```
                        out.println("<link type=\"text/css\" rel=\"stylesheet\"
href=\"Display_Info.css\">");
                        /*out.println("<script>");
                        out.println("function CloseS(){");
                        out.println("window.location.href=\"Logout\";}");
                        out.println("</script>");*/
                out.println("</head>");

                out.println("<body>");
                        out.println("<h2> "+message+" | <a
href=\"Collect_Info\" target=\"_self\"> Register Here </a> | <a
href=\"Logout\"> Logout </a></h2>");
                        out.println("<p>UserName :"+username+"</p>");
                        ServletConfig config=getServletConfig();
                                String classname =
config.getInitParameter("classname");
                                String url =  config.getInitParameter("url");
                                String uname =
config.getInitParameter("username");
                                String password =
config.getInitParameter("password");

                                try {
                                        Class.forName(classname);
                                }catch(Exception e) {e.printStackTrace();}

                                try(Connection con =
DriverManager.getConnection(url,uname,password);){


                                        Statement stmt1 =
con.createStatement();
                                        ResultSet rs =
stmt1.executeQuery("SELECT * FROM emp10");
                                        out.println("<table>");

        out.println("<tr><th>EMAIL</th><th>LASTNAME</th><th>FIRSTNAME</th><th>
GENDER</th><th>EXPERIENCE</th><th>PASSWORD</th></tr>");
                                        while(rs.next()){


        out.println("<tr><td>"+rs.getString(1)+"</td>"+"<td>"+rs.getString(2)+
"</td>"+"<td>"+rs.getString(3)+"</td>"+"<td>"+rs.getString(4)+"</td>"+"<td>"+
rs.getString(5)+"</td>"+"<td>"+rs.getString(6)+"</td></tr>");

                                        }
                                        stmt1.close();
                                        out.println("</table>");
                                }
                                catch(SQLException e){

                                        out.println("<p>"+e+"</p>");
                                }
                                catch(Exception e){

                                        out.println("<p>"+e+"</p>");
```

```
                                }

                out.println("</body>");
                out.println("</html>");

        }

}
```

*****Servlet for Logout*****

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;

// First method of creating servlet
public class Logout extends HttpServlet{
        private String message;

        public void init() throws ServletException{
                message = "Logout Employee";
        }

        public void service(HttpServletRequest request,HttpServletResponse
response) throws ServletException,IOException{
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                HttpSession session=request.getSession(false);
                if(session==null){
                        response.sendRedirect("Login");
                }else{
                        session.invalidate();
                }

                        //RequestDispatcher
rd=request.getRequestDispatcher("Login");
                        //rd.forward(request,response);
                        response.sendRedirect("Login");

        }
}
```

Rest all files are same as above P1

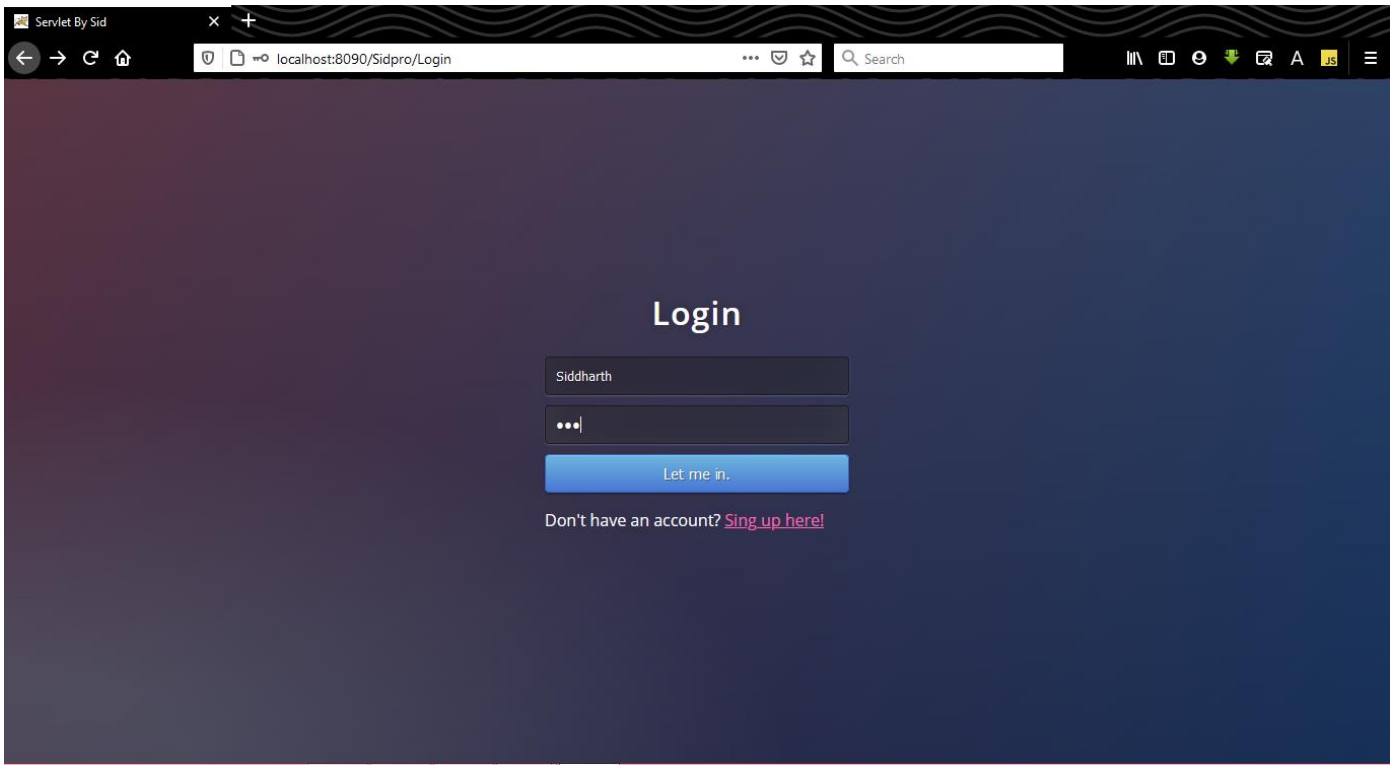


Fig3.3a – Try to Login with Wrong password

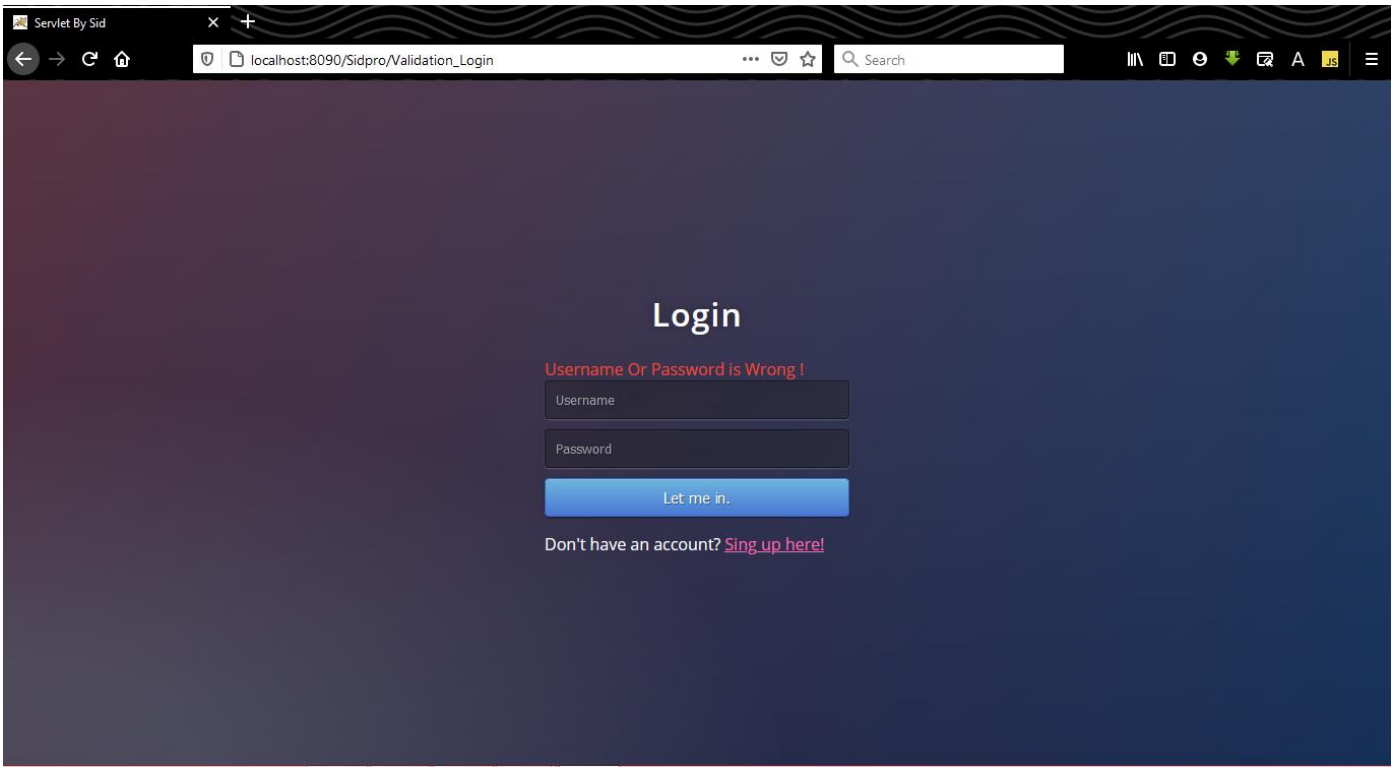


Fig3.3b – For Wrong Username or Password got Error Message

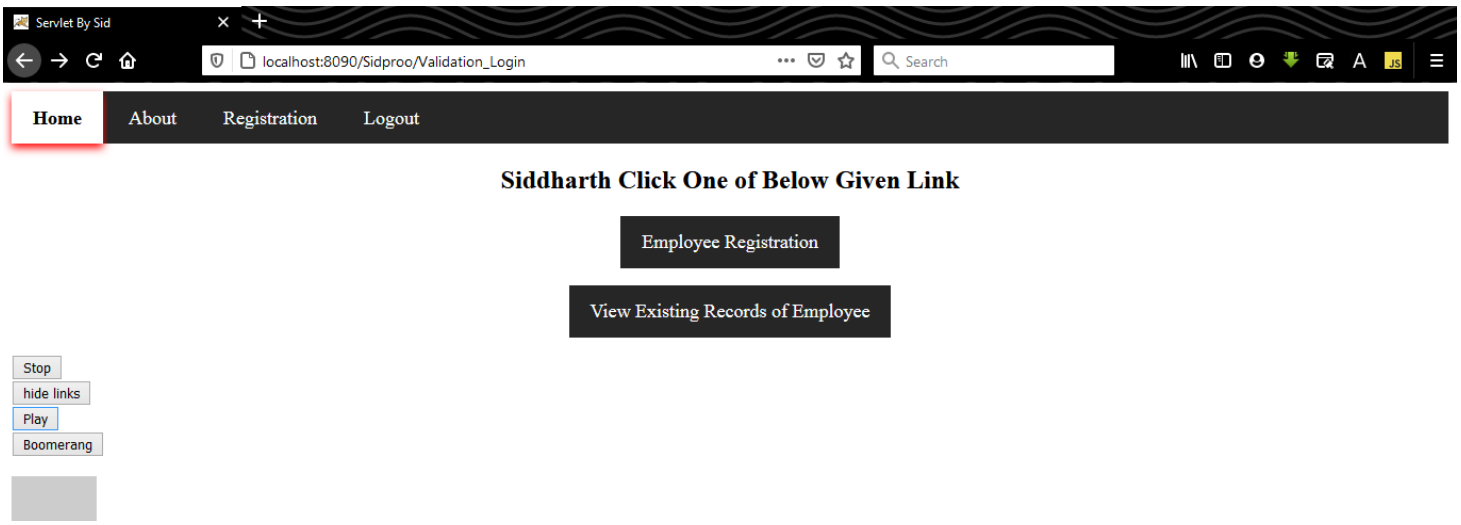
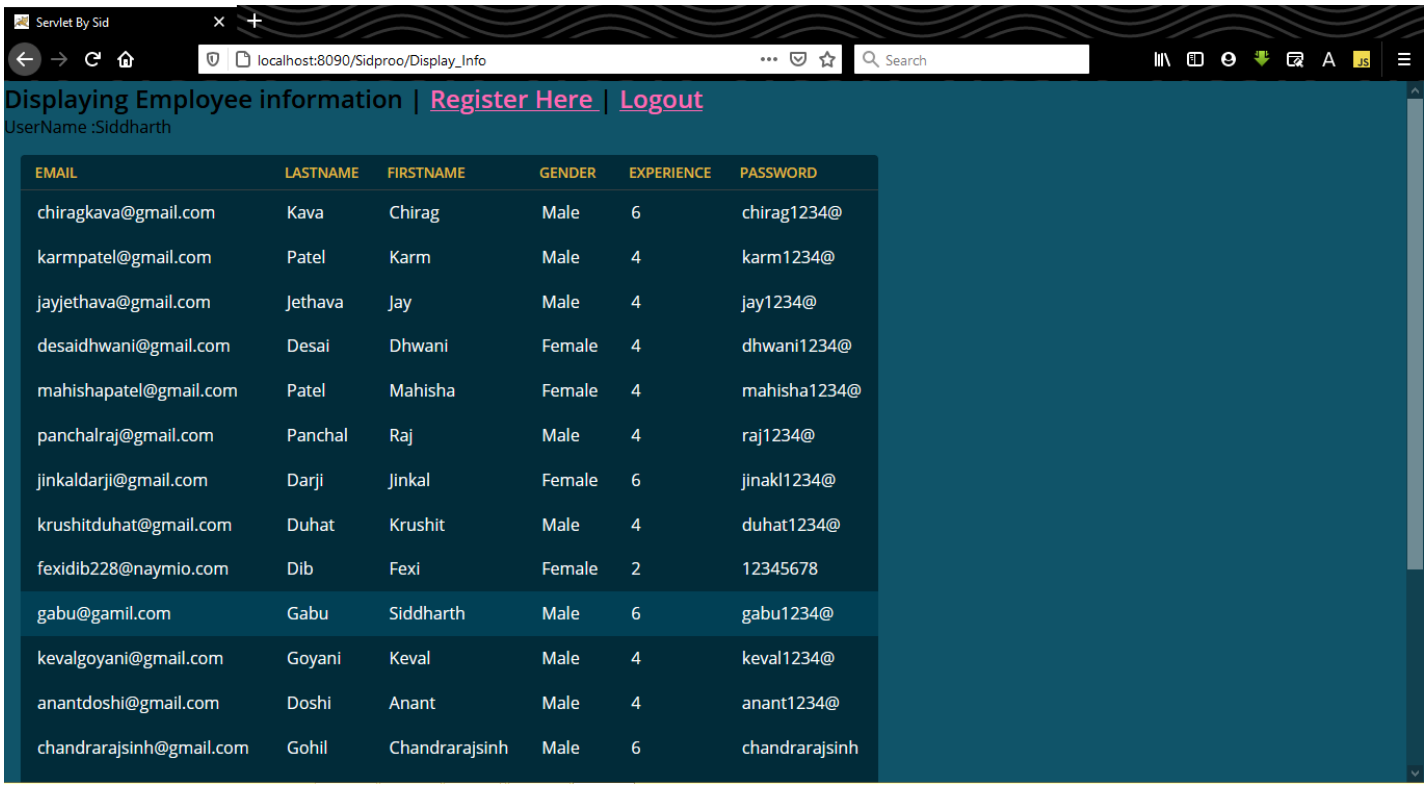


Fig3.3c – After successful login



EMAIL	LASTNAME	FIRSTNAME	GENDER	EXPERIENCE	PASSWORD
chiragkava@gmail.com	Kava	Chirag	Male	6	chirag1234@
karmpatel@gmail.com	Patel	Karm	Male	4	karm1234@
jayjethava@gmail.com	Jethava	Jay	Male	4	jay1234@
desaidhwani@gmail.com	Desai	Dhwani	Female	4	dhwani1234@
mahishapatel@gmail.com	Patel	Mahisha	Female	4	mahisha1234@
panchalraj@gmail.com	Panchal	Raj	Male	4	raj1234@
jinkaldarji@gmail.com	Darji	Jinkal	Female	6	jinakl1234@
krushitduhat@gmail.com	Duhat	Krushit	Male	4	duhat1234@
fexidib228@naymio.com	Dib	Fexi	Female	2	12345678
gabu@gamil.com	Gabu	Siddharth	Male	6	gabu1234@
kevalgoyani@gmail.com	Goyani	Keval	Male	4	keval1234@
anantdoshi@gmail.com	Doshi	Anant	Male	4	anant1234@
chandrarajsinh@gmail.com	Gohil	Chandrarajsinh	Male	6	chandrarajsinh

Fig3.3d – Logged in user can see Info of Employees

Answer the followings:

1) List various characteristics of Enterprise application.

Given below are few common characteristics of Enterprise Applications. If any software product has the following characteristics, we can identify it as an Enterprise Application. These ware originally documented by “Martin Fowler”, in his book “Patterns of Enterprise Application Architecture”.

- **Persistent Data -** Enterprise applications usually involve persistent data. The data is persistent because it needs to be around between multiple runs of the program—indeed, it usually needs to persist for several years. Also during this time there will be many changes in the programs that use it.
- **Lot of Data -** There's usually a lot of data, a moderate system will have over 1 GB of data organized in tens of millions of records—so much that managing it is a major part of the system.
- **Access Data Concurrently -** Usually many people access data concurrently. For many systems this may be less than a hundred people, but for Web-based systems that talk over the Internet this goes up by orders of magnitude.
- **Lot of User Interface Screens -** With so much data, there's usually a lot of user interface screens to handle it. It's not unusual to have hundreds of distinct screens.
- **Integrate with other Enterprise Applications -** Enterprise applications rarely live on an island. Usually they need to integrate with other enterprise applications scattered around the enterprise. The various systems are built at different times with different technologies, and even the collaboration mechanisms will be different.

2) Draw and explain servlet life cycle.

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

1. Servlet class is loaded.
2. Servlet instance is created.
3. init method is invoked.
4. service method is invoked.
5. destroy method is invoked.

As displayed in the below diagram, there are three states of a servlet: new, ready and end.
The servlet is in new state if servlet instance is created.
After invoking the init() method, Servlet comes in the ready state. In the ready state, servlet performs all the tasks.
When the web container invokes the destroy() method, it shifts to the end state.

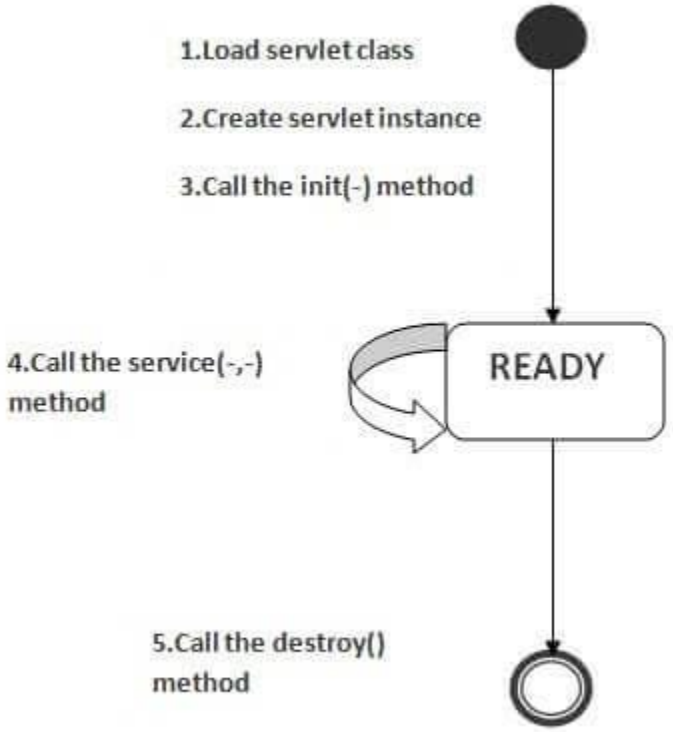


Fig1 – servlet life cycle

3) Compare servlet with JSP.

A servlet is a Java class which is used to extend the capabilities of servers that host applications accessed by means of a request-response model. Servlets are mainly used to extend the applications hosted by webs servers, however, they can respond to other types of requests too. For such applications, HTTP-specific servlet classes are defined by Java Servlet technology.

A JSP is a text document which contains two types of text: static data and dynamic data. The static data can be expressed in any text-based format (like HTML, XML, SVG and WML), and the dynamic content can be expressed by JSP elements.

Servlet	JSP
Servlet is a java code.	JSP is a html based code.
Writing code for servlet is harder than JSP as it is html in java.	JSP is easy to code as it is java in html.
Servlet plays a controller role in MVC approach.	JSP is the view in MVC approach for showing output.
Servlet is faster than JSP.	JSP is slower than Servlet because the first step in JSP lifecycle is the translation of JSP to java code and then compile.
Servlet can accept all protocol requests.	JSP only accept http requests.
In Servlet, we can override the service() method.	In JSP, we cannot override its service() method.
In Servlet by default session management is not enabled, user have to enable it explicitly.	In JSP session management is automatically enabled.
In Servlet we have to implement everything like business logic and presentation logic in just one servlet file.	In JSP business logic is separated from presentation logic by using javaBeans.
Modification in Servlet is a time consuming task because it includes reloading, recompiling and restarting the server.	JSP modification is fast, just need to click the refresh button.

4) What is session tracking? How can we implement it in Servlet?

Session simply means a particular interval of time.

Session Tracking is a way to maintain state (data) of an user. It is also known as **session management** in servlet.

Http protocol is a stateless so we need to maintain state using session tracking techniques. Each time user requests to the server, server treats the request as the new request. So we need to maintain the state of an user to recognize to particular user.

HTTP is stateless that means each request is considered as the new request.

There are four techniques used in Session tracking:

1. **Cookies**
2. **Hidden Form Field**
3. **URL Rewriting**
4. **HttpSession**

HttpSession, In such case, container creates a session id for each user. The container uses this id to identify the particular user. An object of HttpSession can be used to perform two tasks:

1. bind objects
2. view and manipulate information about a session, such as the session identifier, creation time, and last accessed time

The HttpServletRequest interface provides two methods to get the object of HttpSession:

1. **public HttpSession getSession():**Returns the current session associated with this request, or if the request does not have a session, creates one.
2. **public HttpSession getSession(boolean create):**Returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session.

In this **above Practical example**, we are setting the attribute in the session scope in one servlet and getting that value from the session scope in another servlet. To set the attribute in the session scope, we have used the setAttribute() method of HttpSession interface and to get the attribute, we have used the getAttribute method.

Practical – 4

1. Write a simple JSP program for user Registration & then control will be transfer it into login page. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database.

***** register_user.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Collect_Info.css">
 </head>

 <body>
    <%
        String fname ="",lname="",pass="",enNo="";
        if(request.getAttribute("fname_error")!=null){
            fname=(String)request.getAttribute("fname_error");
            request.removeAttribute("fname_error");
        }
        if(request.getAttribute("lname_error")!=null){
            lname=(String)request.getAttribute("lname_error");
            request.removeAttribute("lname_error");
        }
        if(request.getAttribute("enNo_error")!=null){
            enNo=(String)request.getAttribute("enNo_error");
            request.removeAttribute("enNo_error");
        }
        if(request.getAttribute("pass_error")!=null){
            pass=(String)request.getAttribute("pass_error");
            request.removeAttribute("pass_error");
        }
    %>
    <h2>Collecting Employee information Of Collegeek</h2>

    <form method="post" action="validate_user" class="main_container">
    <div class="containerbase">
        <lable id="F" for="firstname">Enter First Name :<%= fname
%></lable><br>
        <input type="text" name="firstName" id="firstname" placeholder="Enter
your first name here"><br>
        <lable id="L" for="lastname">Enter Last Name :<%= lname
%></lable><br>
        <input type="text" name="lastName" id="lastname" placeholder="Enter
your last name here"><br>
        <lable id="E" for="enNo">Enter Enrollment no :<%= enNo %></lable><br>
        <input type="text" name="enNo" id="enNo" placeholder="Enter your
Enrollment no here"><br>
```

```
        <lable>Select Gender:</lable><br>
        <input type="radio" name="gender" value="Male" checked>Male
        <input type="radio" name="gender" value="Female">Female
        <input type="radio" name="gender" value="other">other<br>

        <lable id="P" for="Pass">Enter Password:<%= pass %></lable><br>
        <input type="password" name="password" id="Pass" placeholder="Enter
your password here"><br>
        <div class="container">
        <input type="submit" value="Submit"><input type="reset"
value="Reset">
        </div>
    </div>
    </form>

    <section>
        <div class="wave wave1"></div>
        <div class="wave wave2"></div>
        <div class="wave wave3"></div>
    </section>
</body>
</html>
```

***** validate_user.jsp *****

```
<%@ page import="java.util.regex.Pattern" %>
<%@ page import="sidpro.Register_user" %>
<%
        String fname =
((request.getParameter("firstName")!=null)?request.getParameter("firstName"):
"");
        String lname =
(request.getParameter("lastName")==null)?"":request.getParameter("lastName");
        String enNo =
(request.getParameter("enNo")==null)?"":request.getParameter("enNo");
        String pass =
(request.getParameter("password")==null)?"":request.getParameter("password");
        String gender = request.getParameter("gender");

        boolean go = true;
            if(fname.equals("")){
                go=false;
                request.setAttribute("fname_error", "<span
style=\"color:red;\"> * First Name is Required!</span>");
            }
            else{
                if(!Pattern.matches("^[a-zA-Z]*$",fname)){
                    go=false;
                    request.setAttribute("fname_error", "<span
style=\"color:red;\"> * Only letters and white space allowed!</span>");
                }
            }

            if(lname.equals("")){
                go=false;
                request.setAttribute("lname_error", "<span
style=\"color:red;\"> * Last Name is Required!</span>");
```

```
            }
            else{
                if(!Pattern.matches("^[a-zA-Z]*$",lname)){
                    go=false;
                    request.setAttribute("lname_error", "<span
style=\"color:red;\"> * Only letters and white space allowed!</span>");
                }
            }

            if(pass.equals("")){
                go=false;
                request.setAttribute("pass_error", "<span
style=\"color:red;\"> * Password is Required!</span>");
            }else{
                if(pass.length()<8){
                    go=false;
                    request.setAttribute("pass_error", "<span
style=\"color:red;\"> * Password length must be 8 !</span>");
                }
            }

            if(enNo.equals("")){
                go=false;
                request.setAttribute("enNo_error", "<span
style=\"color:red;\"> * Enrollment no is Required!</span>");
            }
            else{
                if(!Pattern.matches("^[0-9]{12}$",enNo)){
                    go=false;
                    request.setAttribute("enNo_error", "<span
style=\"color:red;\"> * Invalid Enrollment no format!</span>");
                }
            }

            if(go){ %>
                        <jsp:useBean id="userBean"
class="sidpro.StudentBean">
                        <jsp:setProperty name="userBean" property="*"/>
                        </jsp:useBean>
<%
                int status=Register_user.register(userBean);
                if(status>0)  out.println("You are successfully registered");
                response.sendRedirect("Login.jsp");
            }else{  %>
                <jsp:include page="register_user.jsp"/>
<%          }
%>
```

***** Logout.jsp *****

```
<%
       session.invalidate();
    response.sendRedirect("Login.jsp");
%>
```

***** five_marks.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Collect_Info.css">
    <style>
      /* Chrome, Safari, Edge, Opera */
input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
  -webkit-appearance: none;
  margin: 0;
}

/* Firefox */
input[type=number] {
  -moz-appearance: textfield;
}
input[type=number]{
    padding: 12px 16px;
    margin: 8px 0px;
    display:inline-block;
    border-radius: 5px;
    border:1px solid #ccc;
    box-sizing: border-box;
    width:100%;
  }
    input[type=number]:hover,input[type=number]:focus{
    background-color: lightgray;
    border-color:2px solid gray;
  }
    </style>
 </head>

 <body>

    <h2>Collecting Marks of students </h2>

 <form method="post" action="grade.jsp" class="main_container">
 <div class="containerbase">
     <lable id="F" for="ADJ">Enter Mark of Subject 1 :</lable><br>
     <input type="number" name="ADJ" id="ADJ" placeholder="Enter Mark of
Advance Java Programming" min="0" max="100" required><br>
     <lable id="L" for="DM">Enter Mark of Subject 2 :</lable><br>
     <input type="number" name="DM" id="DM" placeholder="Enter Mark of Data
mining" min="0" max="100" required><br>
     <lable id="F" for="TOC">Enter Mark of Subject 3 :</lable><br>
     <input type="number" name="TOC" id="TOC" placeholder="Enter Mark of
Theory of Computation" min="0" max="100" required><br>
     <lable id="L" for="MPI">Enter Mark of Subject 4 :</lable><br>
     <input type="number" name="MPI" id="MPI" placeholder="Enter Mark of
Microprocessor and Interfacing" min="0" max="100" required><br>
```

```
     <lable id="F" for="DV">Enter Mark of Subject 5 :</lable><br>
     <input type="number" name="DV" id="DV" placeholder="Enter Mark of Data
visualization" min="0" max="100" required><br>

     <div class="container">
     <input type="submit" value="Submit"><input type="reset" value="Reset">
     </div>
 </div>
 </form>

 </body>

 </html>
```

***** Login.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Login.css">
 </head>

 <body>
     <div class="login">
     <h1>Login</h1>
     <% String fname ="";
        if(request.getAttribute("fname_error")!=null){
            fname=(String)request.getAttribute("fname_error");
            request.removeAttribute("fname_error");
        } %>
     <span style="color:#f24835"><%= fname %></span>
     <form action="validate_Login" method="post">
     <input type="text" name="uname" placeholder="Username"
required="required" />
     <input type="password" name="pass" placeholder="Password"
required="required" />
     <button type="submit" class="btn btn-primary btn-block btn-large">Let me
in.</button>
     <p>Don't have an account? <a href="register_user.jsp"
target="_self">Sing up here!</a></p>
     </form>
     </div>

 </body>
 </html>
```

***** grade.jsp *****

```
<!DOCTYPE html>
 <html>
 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Collect_Info.css">
     <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.cs
s"
     integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
 </head>

 <body>
    <h2>Grades </h2>

    <%
        int s[] = new int[5];
        String name[] = {"ADJ","DM","TOC","MPI","DV"};
        s[0] = Integer.valueOf(request.getParameter("ADJ"));
        s[1] = Integer.valueOf(request.getParameter("DM"));
        s[2] = Integer.valueOf(request.getParameter("TOC"));
        s[3] = Integer.valueOf(request.getParameter("MPI"));
        s[4] = Integer.valueOf(request.getParameter("DV"));

        out.println("<table class=\"table table-dark\">");
        out.print("<tr>");
        out.print("<th>Subject</th><th>Grades</th>");
        out.println("</tr>");

        for(int i=0;i<5;++i){
            String l = "";
            if(s[i]>=85)l="AA";
            else if(s[i]>=75)l="AB";
            else if(s[i]>=65)l="BB";
            else if(s[i]>=55)l="BC";
            else if(s[i]>=45)l="CC";
            else if(s[i]>=40)l="CD";
            else if(s[i]>=35)l="DD";
            else l = "FF";
            out.print("<tr>");
            out.print("<td>"+name[i]+"</td><td>"+l+"</td>");
            out.println("</tr>");
        }
        out.println("<tr><td>GTU</td><td>sucess is inevitable
ha..ha..ha..!!!<td></tr>");
        out.println("</table>");
    %>
 </body>

 </html>
```

***** validate_login.jsp *****

```
<%@ page import="java.sql.*" %>
<%
        String username = request.getParameter("uname");
        String pass = request.getParameter("pass");
        boolean go = true;

                String classname =
application.getInitParameter("classname");
                String url =  application.getInitParameter("url");
                String uname = application.getInitParameter("username");
                String password = application.getInitParameter("password");

                try {
                    Class.forName(classname);
                }catch(Exception e) {e.printStackTrace();}

                try{
                    Connection con =
DriverManager.getConnection(url,uname,password);
                    Statement stmt = con.createStatement();
                    ResultSet rs = stmt.executeQuery("SELECT * FROM user_jsp
WHERE firstname='"+username+"' AND ROWNUM <= 1");
                    while(rs.next()){
                        if(rs.getString(5).equals(pass)){
                            go=false;

                                session.setAttribute("username",username);

                        }
                    }
                    stmt.close();
                    con.close();
                }
                catch(SQLException e){

                    out.println("<p>"+e+"</p>");
                }
                catch(Exception e){

                    out.println("<p>"+e+"</p>");
                }

            if(go){

                request.setAttribute("fname_error", " Username Or Password is
Wrong !");
                //out.println("<p>First name:
"+request.getParameter("fname")+"</p>");  %>
                <jsp:include page="Login.jsp"/>
<%          }else{

                response.sendRedirect("five_marks.jsp");

            }
%>
```

***** StudetnBean.java *****

```
package sidpro;

import java.io.Serializable;

public class StudentBean implements Serializable {
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private String firstName;   // Student first name
    private String lastName;    // Student last name
    private String password;        // Student password
    private String gender;      // Student gender
    private String enNo;            // Student Enrollment no
    /**
     *
     */
    public StudentBean() {
    }
    /**
     * @param firstName
     * @param lastName
     * @param password
     * @param gender
     * @param enNo
     */
    public StudentBean(String firstName, String lastName, String password,
String gender, String enNo) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.password = password;
        this.gender = gender;
        this.enNo = enNo;
    }
    /**
     * @return the firstName
     */
    public String getFirstName() {
        return firstName;
    }
    /**
     * @param firstName the firstName to set
     */
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    /**
     * @return the lastName
     */
    public String getLastName() {
        return lastName;
    }
    /**
     * @param lastName the lastName to set
     */
```

```
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    /**
     * @return the city
     */
    public String getPassword() {
        return password;
    }
    /**
     * @param city the city to set
     */
    public void setPassword(String password) {
        this.password = password;
    }
    /**
     * @return the gender
     */
    public String getGender() {
        return gender;
    }
    /**
     * @param gender the gender to set
     */
    public void setGender(String gender) {
        this.gender = gender;
    }
    /**
     * @return the enNo
     */
    public String getEnNo() {
        return enNo;
    }
    /**
     * @param enNo the enNo to set
     */
    public void setEnNo(String enNo) {
        this.enNo = enNo;
    }

}
```

***** Register_user.java *****

```
package sidpro;

import java.sql.*;
import static database.Database.*;

public class Register_user {

    public static int register(StudentBean u){
        int status=0;

                long eno = Long.valueOf(u.getEnNo());
                try {
                    Class.forName(CLASSNAME);
```

```
                }catch(Exception e) {e.printStackTrace();}

                try(Connection con =
DriverManager.getConnection(URL,UNAME,PASSWORD);){

                    PreparedStatement stmt = con.prepareStatement("INSERT
INTO user_jsp VALUES(?,?,?,?,?)");
                    stmt.setLong(1,eno);
                    stmt.setString(2,u.getLastName());
                    stmt.setString(3,u.getFirstName());
                    stmt.setString(4,u.getGender());
                    stmt.setString(5,u.getPassword());
                    int n = stmt.executeUpdate();
                    stmt.close();
                    //out.println("<p>"+n+" row(s) inserted.</p>");
                    //response.sendRedirect("Display_Info");
                    //System.out.println(n+" row(s) inserted.");

                }
                catch(SQLException e){

                    e.printStackTrace();
                }
                catch(Exception e){
                    e.printStackTrace();
                    //out.println("<p>"+e+"</p>");
                }

        return status;
    }

}
```

***** Database.java *****

```
package database;

public interface Database {
String CLASSNAME="oracle.jdbc.driver.OracleDriver";
String URL="jdbc:oracle:thin:@localhost:1521:XE";
String UNAME="admin";
String PASSWORD="admin";

}
```

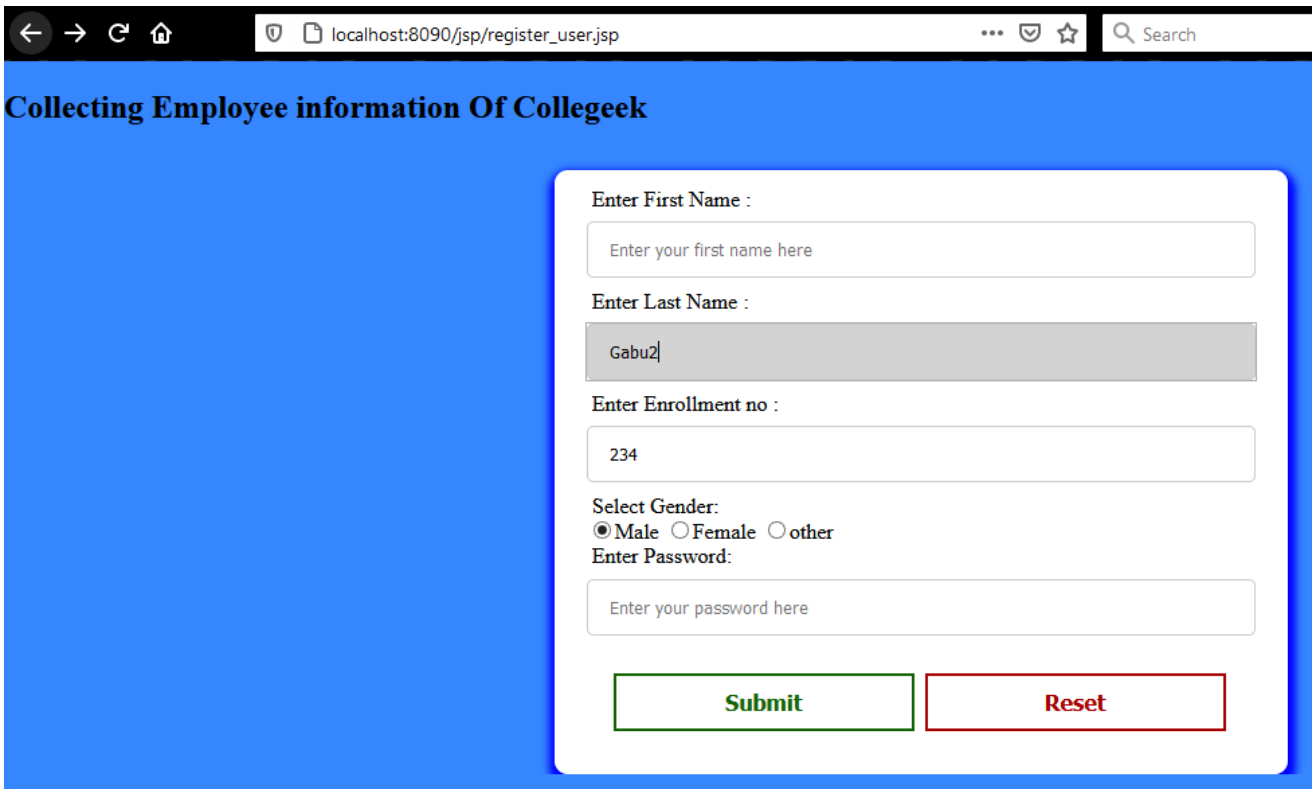


Fig4.1 – Registration form

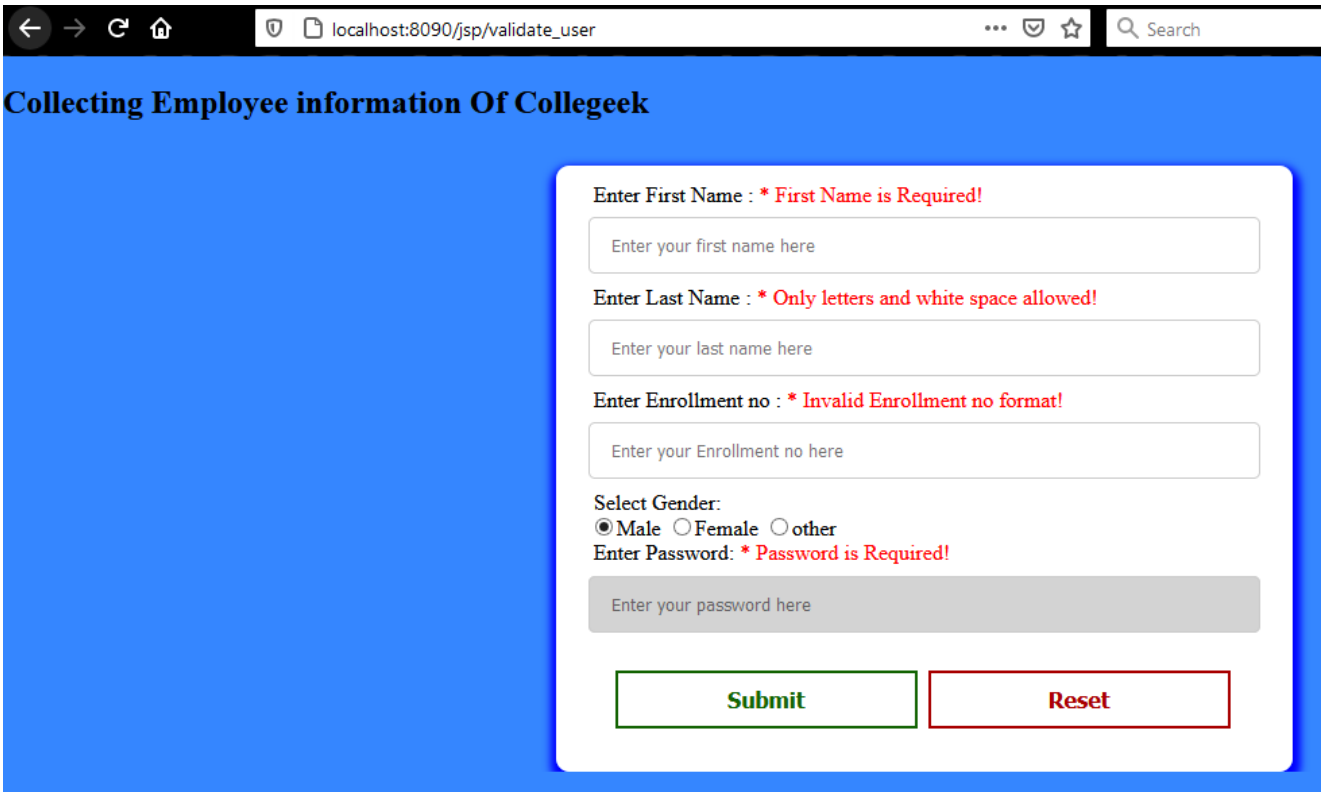


Fig4.2 – Registration form validation

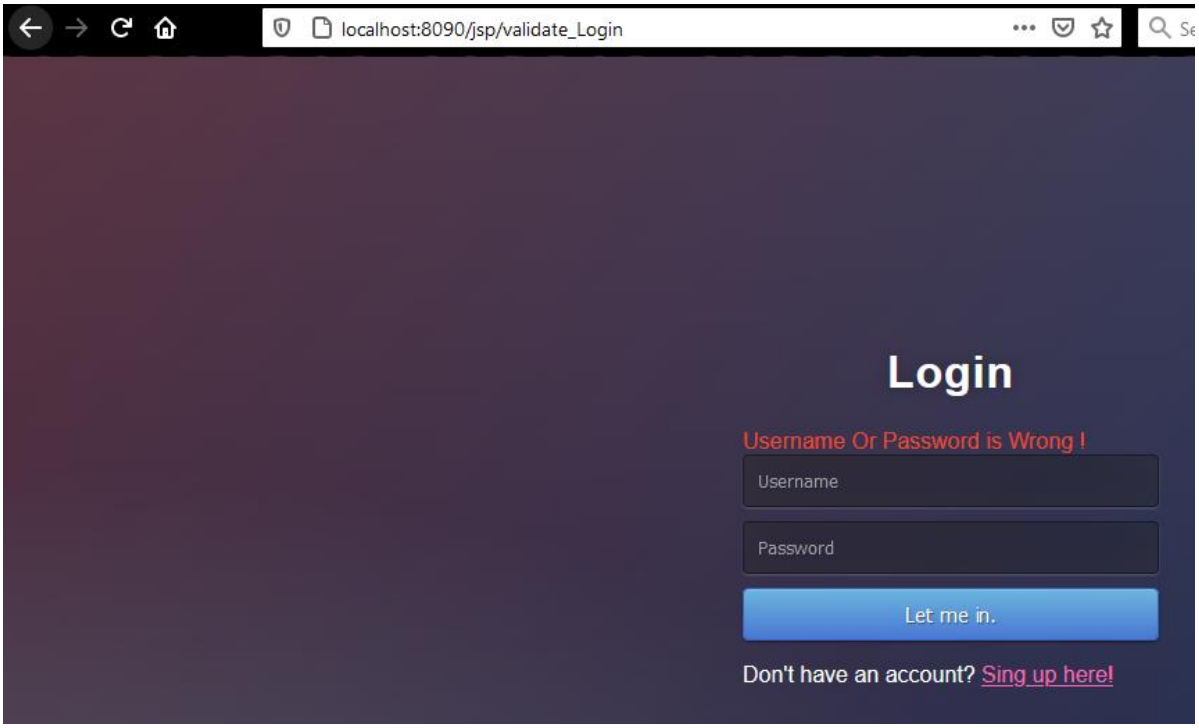


Fig4.3 – Invalid Login

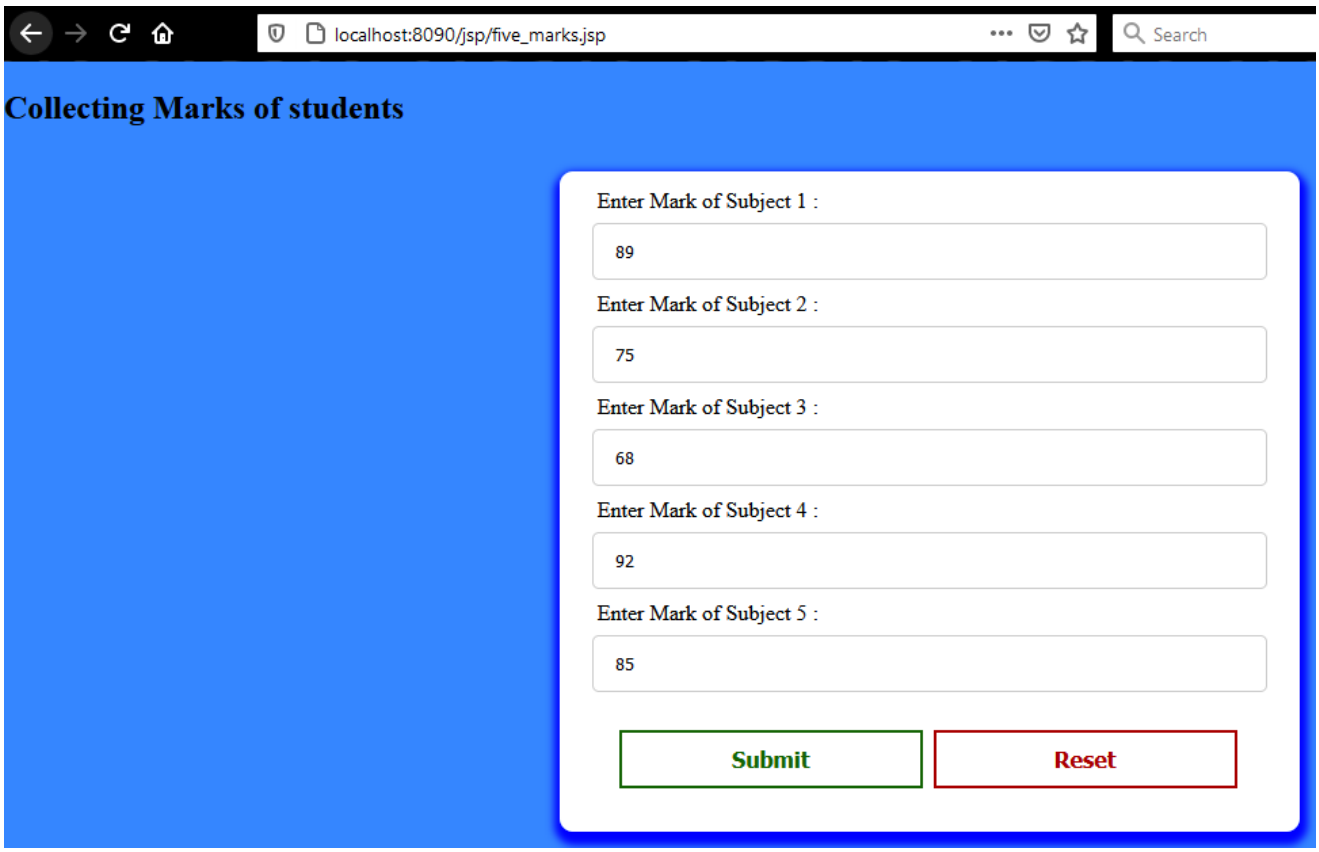


Fig4.4 - Collecting Marks

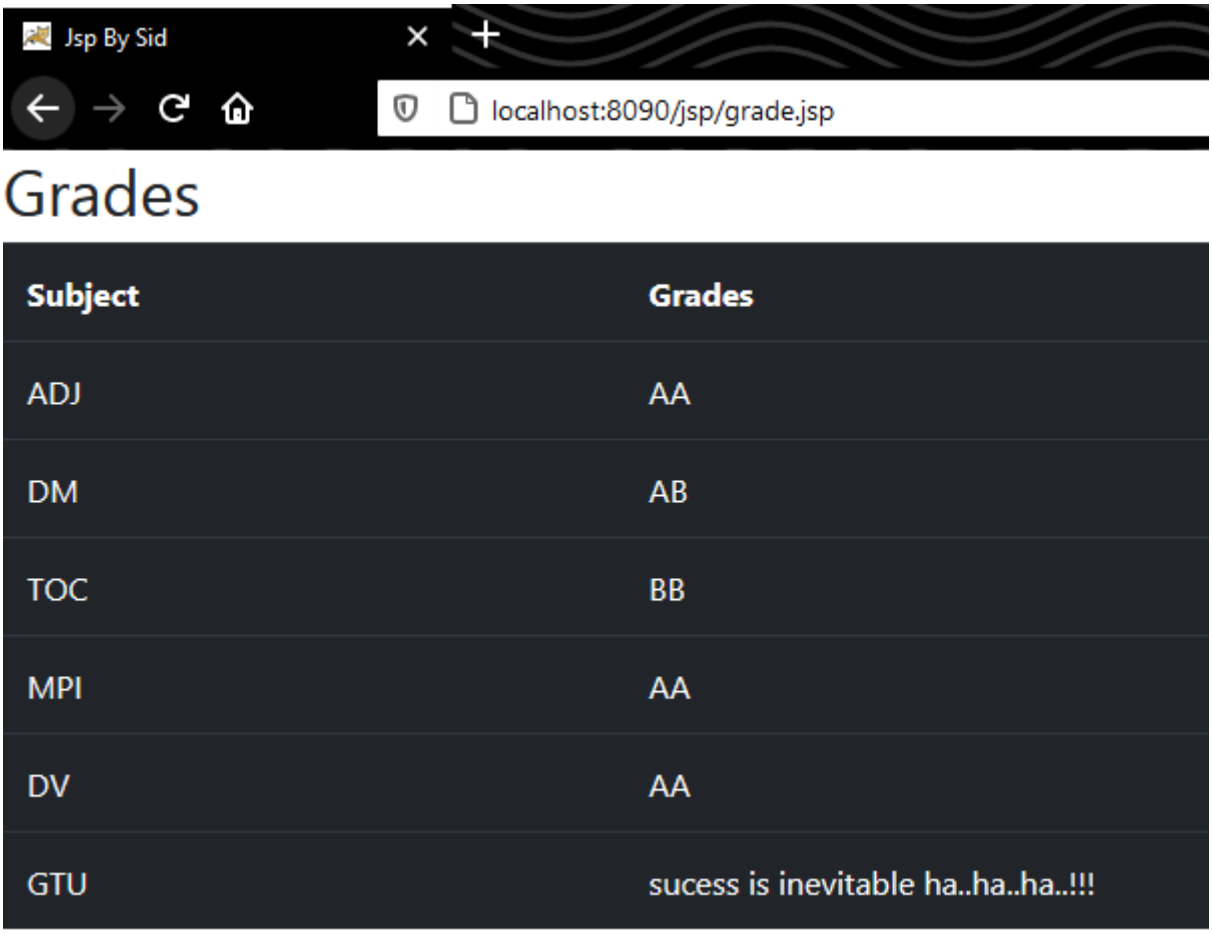


Fig4.5 – Grade of Student

2. In above practical perform session tracking.

***** grader.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Collect_Info.css">
     <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.cs
s"
     integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
 </head>
```

```
 <body>

    <h2>Grades <br><a href="register_user.jsp" target="_self"> Register Here
</a> | <a href="Logout.jsp"> Logout </a></h2>
    <%
            String
username=(String)(session.getAttribute("username")==null?"":session.getAttrib
ute("username"));
            if(username.equals("")){
                    response.sendRedirect("Login.jsp");
            }else{
                    out.println("<h3>UserName: "+username+"</h3>");
            }

        int s[] = new int[5];
        String name[] = {"ADJ","DM","TOC","MPI","DV"};
        s[0] = Integer.valueOf(
(request.getParameter("ADJ")==null?"0":request.getParameter("ADJ")) );
        s[1] = Integer.valueOf(
(request.getParameter("DM")==null?"0":request.getParameter("DM")) );
        s[2] = Integer.valueOf(
(request.getParameter("TOC")==null?"0":request.getParameter("TOC")) );
        s[3] = Integer.valueOf(
(request.getParameter("MPI")==null?"0":request.getParameter("MPI")) );
        s[4] = Integer.valueOf(
(request.getParameter("DV")==null?"0":request.getParameter("DV")) );

        out.println("<table class=\"table table-dark\">");
        out.print("<tr>");
        out.print("<th>Subject</th><th>Grades</th>");
        out.println("</tr>");

        for(int i=0;i<5;++i){
            String l = "";
            if(s[i]>=85)l="AA";
            else if(s[i]>=75)l="AB";
            else if(s[i]>=65)l="BB";
            else if(s[i]>=55)l="BC";
            else if(s[i]>=45)l="CC";
            else if(s[i]>=40)l="CD";
            else if(s[i]>=35)l="DD";
            else l = "FF";
            out.print("<tr>");
            out.print("<td>"+name[i]+"</td><td>"+l+"</td>");
            out.println("</tr>");
        }
        out.println("<tr><td>GTU</td><td>sucess is inevitable
ha..ha..ha..!!!<td></tr>");
        out.println("</table>");
    %>
 </body>

 </html>
```

***** five_marks.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
     <title> Jsp By Sid </title>
     <meta charset="utf-8">
     <meta name="author" content="SidPro"/>
     <meta name="viewport" content="width=device-width,initial-scale=1">
     <link type="text/css" rel="stylesheet" href="Collect_Info.css">
    <style>
      /* Chrome, Safari, Edge, Opera */
input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
  -webkit-appearance: none;
  margin: 0;
}

/* Firefox */
input[type=number] {
  -moz-appearance: textfield;
}
input[type=number]{
    padding: 12px 16px;
    margin: 8px 0px;
    display:inline-block;
    border-radius: 5px;
    border:1px solid #ccc;
    box-sizing: border-box;
    width:100%;
  }
    input[type=number]:hover,input[type=number]:focus{
    background-color: lightgray;
    border-color:2px solid gray;
  }
    </style>
 </head>

 <body>

    <h2>Collecting Marks of students <br><a href="register_user.jsp"
target="_self"> Register Here </a> | <a href="Logout.jsp"> Logout </a></h2>
<%

            String
username=(String)(session.getAttribute("username")==null?"":session.getAttrib
ute("username"));
            if(username.equals("")){
                    response.sendRedirect("Login.jsp");
            }else{
                    out.println("<h3>UserName: "+username+"</h3>");
            }
%>
 <form method="post" action="grade.jsp" class="main_container">
 <div class="containerbase">
     <lable id="F" for="ADJ">Enter Mark of Subject 1 :</lable><br>
```

```
     <input type="number" name="ADJ" id="ADJ" placeholder="Enter Mark of
Advance Java Programming" min="0" max="100" required><br>
     <lable id="L" for="DM">Enter Mark of Subject 2 :</lable><br>
     <input type="number" name="DM" id="DM" placeholder="Enter Mark of Data
mining" min="0" max="100" required><br>
     <lable id="F" for="TOC">Enter Mark of Subject 3 :</lable><br>
     <input type="number" name="TOC" id="TOC" placeholder="Enter Mark of
Theory of Computation" min="0" max="100" required><br>
     <lable id="L" for="MPI">Enter Mark of Subject 4 :</lable><br>
     <input type="number" name="MPI" id="MPI" placeholder="Enter Mark of
Microprocessor and Interfacing" min="0" max="100" required><br>
     <lable id="F" for="DV">Enter Mark of Subject 5 :</lable><br>
     <input type="number" name="DV" id="DV" placeholder="Enter Mark of Data
visualization" min="0" max="100" required><br>

     <div class="container">
     <input type="submit" value="Submit"><input type="reset" value="Reset">
     </div>
 </div>
 </form>

 </body>

 </html>
```

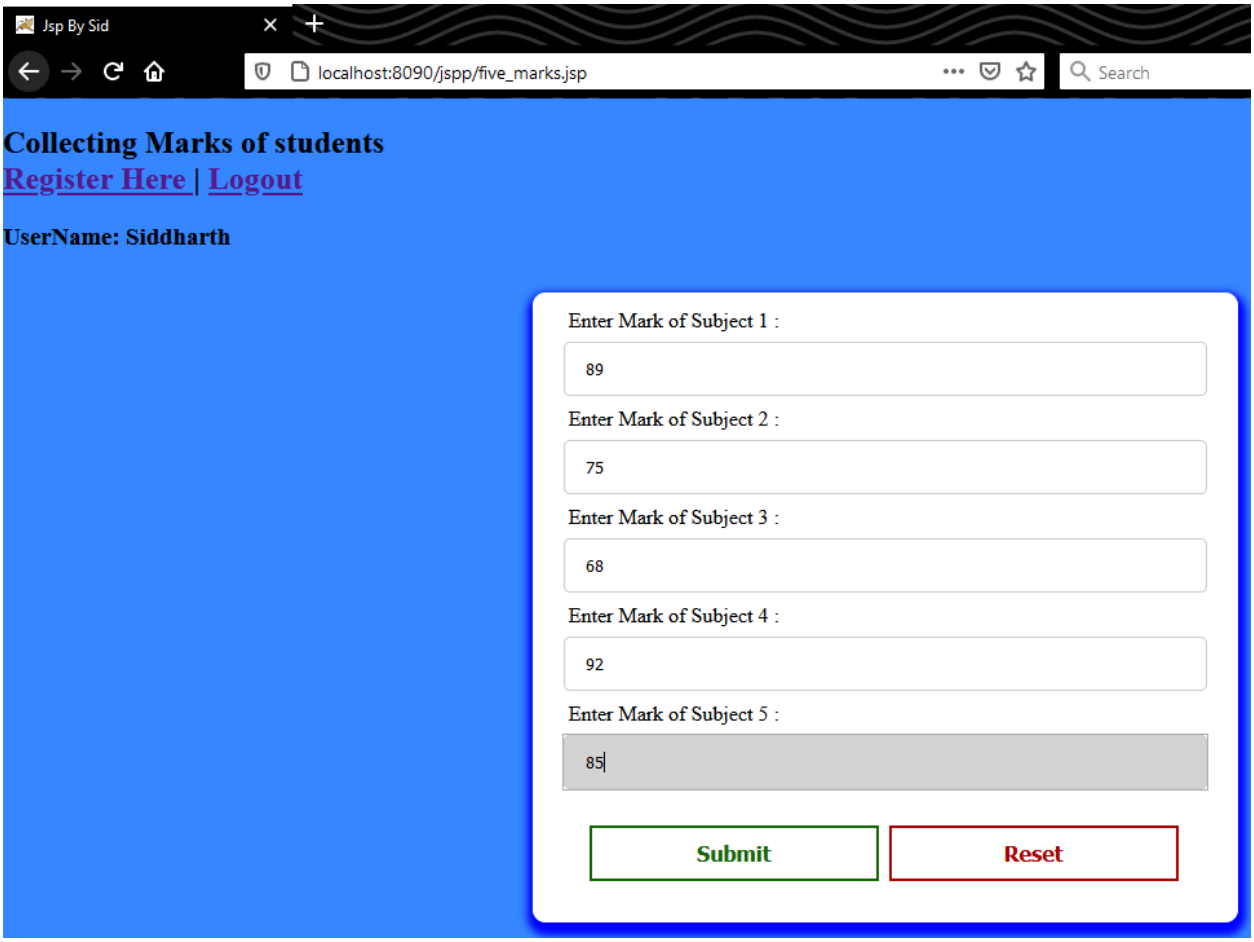


Fig4.6 – After Successful Login

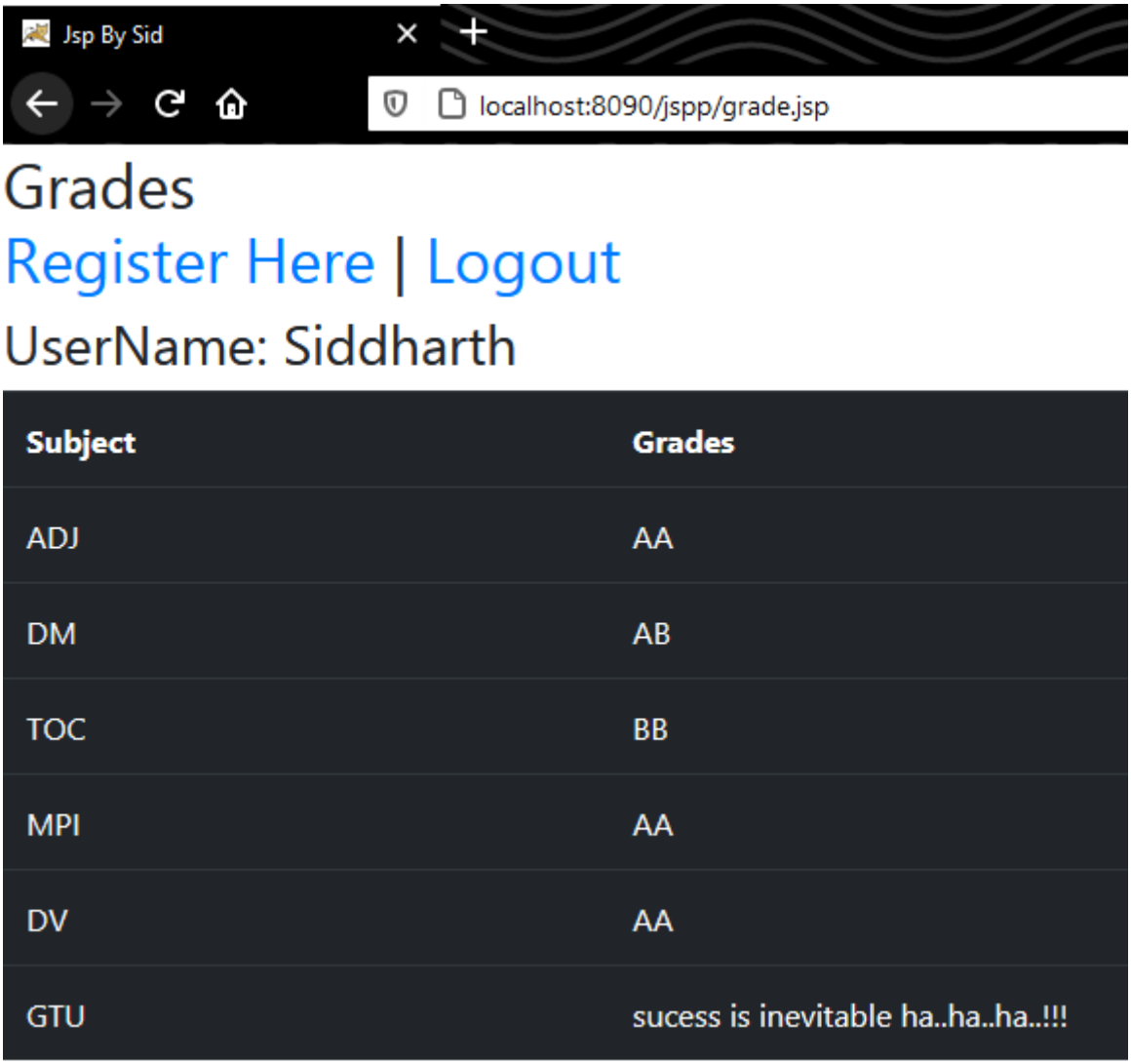


Fig4.7 – Grade in session tracking

Practical – 5

AIM: Implement a simple hello world web application using JSF.

***** helloworld.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough"
      xmlns:f = "http://java.sun.com/jsf/core">
<head>
  <title>Register - JSF By SidPro</title>
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Collect_Info.css" rel="stylesheet" />
  <style>
      .error{color:red}
  </style>
</head>

<body>
<h2 style="text-align:center">Registration form</h2>
      <h:form method="post"  class="main_container">
            <div class="containerbase">

                    <h:outputLabel for="firstname">Enter First Name
:</h:outputLabel>
                    <h:message for = "firstname" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.firstName}"
name="firstName" id="firstname" a:placeholder="Enter your first name here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="lastname">Enter Last Name
:</h:outputLabel>
                    <h:message for = "lastname" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.lastName}"
name="lastName" id="lastname" a:placeholder="Enter your last name here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="enNo">Enter Enrollment no
:</h:outputLabel>
                    <h:message for = "enNo" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.enNo}"
name="enNo" id="enNo" a:placeholder="Enter your Enrollment no here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel>Select Gender:</h:outputLabel><br/>
                    <h:selectOneRadio value="#{student.gender}">
                      <f:selectItem itemValue = "Male" itemLabel = "Male"
checked="true" />
                      <f:selectItem itemValue = "Female" itemLabel = "Female"
/>
                      <f:selectItem itemValue = "Other" itemLabel = "Other" />
                   </h:selectOneRadio> <br/>

                    <h:outputLabel for="Pass">Enter Password:</h:outputLabel>
                    <h:message for = "Pass" style = "color:red" /><br/>
```

```
                    <h:inputSecret type="password" value="#{student.password}"
name="password" id="Pass" a:placeholder="Enter your password here"
required="true" requiredMessage="This field is required">
                         <f:validateLength minimum = "8" maximum = "14" />
                    </h:inputSecret> <br/>
                    <div class="container">
                         <h:commandButton action="#{student.submit()}"
type="submit" value="Submit" />
                         <h:commandButton type="reset" value="Reset" />
                    </div>
               </div>
      </h:form>
</body>

</html>
```

***** myresponse.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough">
<h:head>
  <title>Hello World - Welcome</title>
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Collect_Info.css" rel="stylesheet" />
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
  </script>
  <style>
  .topnav{
    display:block;
    background-color: #262626;
  }
  a{
    text-decoration:none;
    display: inline-block;
    padding: 14px 20px;
    font-size: 18px;
    color:white;
    background-color: #262626;
  }
  .active{
    background-color: white;
    color: black;
    font-weight: bold;
    box-shadow:0px 4px 8px 0 red;
  }
  .vide{
    position: absolute;
    top : 100px;
    right : 20px;
    outline: 2px solid gray;
    background-color: rgba(255, 0, 0, 0.5);
```

```
    z-index: auto;
  }
  </style>
<script>
  window.oncontextmenu = (e) => {e.preventDefault();}
$(document).ready(function(){
  $("#button4").click(function(){
    $("#div1,#div2,#div3").stop();
  });
  $("#button1").click(function(){
    $("p").toggle();
  });
  $("#button2").click(function(){
    $("#div1").fadeToggle();
    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(2000);
  });
  $("#button3").click(function(){
    $("#div4").animate({left:'250px'},"slow");
    $("#div4").animate({left:'10px'},"slow");
  });
  $("a").click(function(){
    $(this).addClass("active").siblings(this).removeClass("active")});
});
</script>
</h:head>

<h:body>
<div class="topnav">
  <a class="active" href="#home">Home</a>
  <a href="#About">About</a>
  <a href="#Registration">Registration</a>
  <a href="#Login">Login</a>
</div>
      <h1 style="font-size:48px;text-align:center;color:red">Hello,
#{theUserName}</h1>
<p>This is JSF Hello World Program!.</p>
<p>JSF = JavaServer Faces </p>
<button id="button4">Stop</button><br/>
<button id="button1">Hide</button><br/>
<button id="button2">Show</button><br/>
<button id="button3">Boomerang</button>
<div id="div1" style="width:80px;height:80px;display:none;background-
color:yellow">
</div><br/>
<div id="div2" style="width:80px;height:80px;display:none;background-
color:red">
</div><br/>
<div id="div3" style="width:80px;height:80px;display:none;background-
color:blue;">
</div>
<div id="div4" style="width:80px;height:80px;background-
color:#ccc;position:absolute;">
</div>
</h:body>

</html>
```

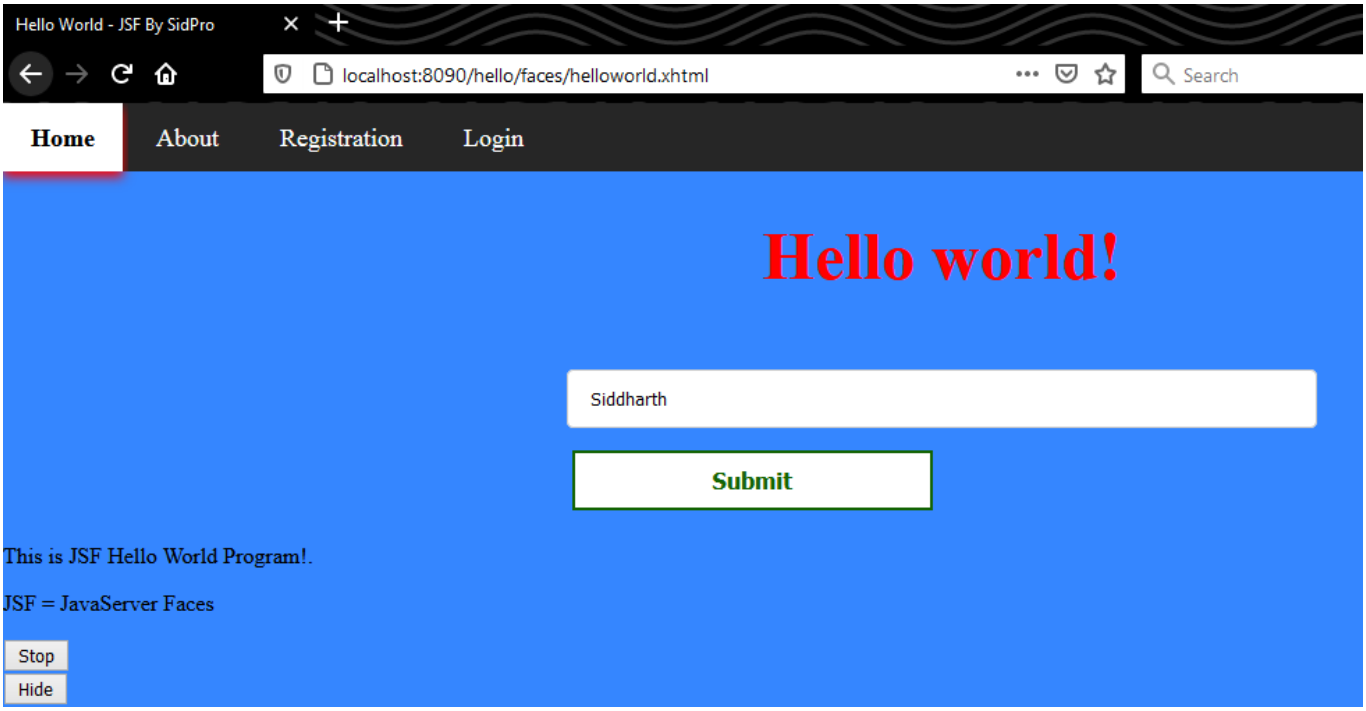


Fig5.1 – hellowrold.xhtml

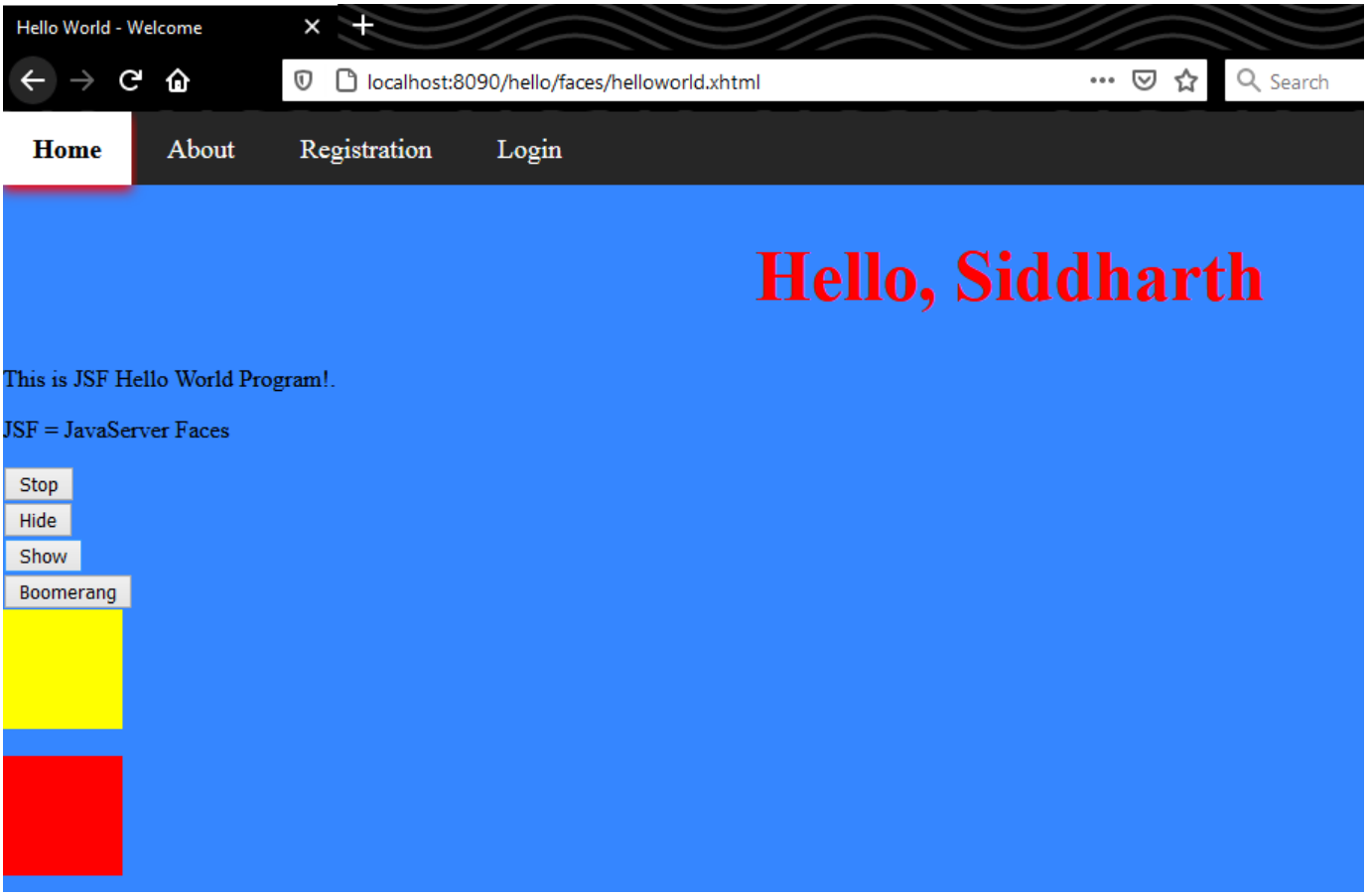


Fig5.2 – myresponse.xhtml

Practical – 6

AIM: Write a JSF application for user Registration which forward to login page if successful registration. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database.

***** register.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough"
      xmlns:f = "http://java.sun.com/jsf/core">
<head>
  <title>Register - JSF By SidPro</title>
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Collect_Info.css" rel="stylesheet" />
  <style>
      .error{color:red}
  </style>
</head>

<body>
<h2 style="text-align:center">Registration form</h2>
      <h:form method="post"  class="main_container">
            <div class="containerbase">

                    <h:outputLabel for="firstname">Enter First Name
:</h:outputLabel>
                    <h:message for = "firstname" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.firstName}"
name="firstName" id="firstname" a:placeholder="Enter your first name here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="lastname">Enter Last Name
:</h:outputLabel>
                    <h:message for = "lastname" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.lastName}"
name="lastName" id="lastname" a:placeholder="Enter your last name here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="enNo">Enter Enrollment no
:</h:outputLabel>
                    <h:message for = "enNo" style = "color:red" /><br/>
                    <h:inputText type="text" value="#{student.enNo}"
name="enNo" id="enNo" a:placeholder="Enter your Enrollment no here"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel>Select Gender:</h:outputLabel><br/>
                    <h:selectOneRadio value="#{student.gender}">
                      <f:selectItem itemValue = "Male" itemLabel = "Male"
checked="true" />
                      <f:selectItem itemValue = "Female" itemLabel = "Female"
/>
                      <f:selectItem itemValue = "Other" itemLabel = "Other" />
```

```
                    </h:selectOneRadio> <br/>

                      <h:outputLabel for="Pass">Enter Password:</h:outputLabel>
                      <h:message for = "Pass" style = "color:red" /><br/>
                      <h:inputSecret type="password" value="#{student.password}"
name="password" id="Pass" a:placeholder="Enter your password here"
required="true" requiredMessage="This field is required">
                            <f:validateLength minimum = "8" maximum = "14" />
                      </h:inputSecret> <br/>
                      <div class="container">
                            <h:commandButton action="#{student.submit()}"
type="submit" value="Submit" />
                            <h:commandButton type="reset" value="Reset" />
                      </div>
                </div>
        </h:form>
</body>

</html>
```

***** Student.java ManagedBean *****

```
package com.sidpro;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;

import javax.faces.bean.ManagedBean;

@ManagedBean
public class Student {
    private String firstName;   // Student first name
    private String lastName;    // Student last name
    private String password;        // Student password
    private String gender;      // Student gender
    private String enNo;            // Student Enrollment no
    /**
     *
     */
    public Student() {
    }
    /**
     * @return the firstName
     */
    public String getFirstName() {
        return firstName;
    }
    /**
     * @param firstName the firstName to set
     */
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    /**
```

```
     * @return the lastName
     */
    public String getLastName() {
        return lastName;
    }
    /**
     * @param lastName the lastName to set
     */
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    /**
     * @return the password
     */
    public String getPassword() {
        return password;
    }
    /**
     * @param password the password to set
     */
    public void setPassword(String password) {
        this.password = password;
    }
    /**
     * @return the gender
     */
    public String getGender() {
        return gender;
    }
    /**
     * @param gender the gender to set
     */
    public void setGender(String gender) {
        this.gender = gender;
    }
    /**
     * @return the enNo
     */
    public String getEnNo() {
        return enNo;
    }
    /**
     * @param enNo the enNo to set
     */
    public void setEnNo(String enNo) {
        this.enNo = enNo;
    }

    public boolean save(){
        int result = 0;
        try{
            long eno = Long.valueOf(this.getEnNo());
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection(
"jdbc:oracle:thin:@localhost:1521:XE","admin","admin");
            PreparedStatement stmt = con.prepareStatement("INSERT INTO
user_jsp VALUES(?,?,?,?,?)");
```

```
            stmt.setLong(1,eno);
            stmt.setString(2,this.getLastName());
            stmt.setString(3,this.getFirstName());
            stmt.setString(4,this.getGender());
            stmt.setString(5,this.getPassword());
            result = stmt.executeUpdate();
            stmt.close();
        }catch(Exception e){
            System.out.println(e);
        }
        if(result == 1){
            return true;
        }else return false;
    }

    public String submit(){
        if(this.save()){
            return "login.xhtml";
        }else return "register.xhtml";
    }

    public String check(){
        boolean go =true;
        try{

            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection(
"jdbc:oracle:thin:@localhost:1521:XE","admin","admin");
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM user_jsp WHERE
firstname='"+this.getFirstName()+"' AND ROWNUM <= 1");
            while(rs.next()){
                if(rs.getString(5).equals(this.getPassword())){
                    go=false;
                }
            }
            stmt.close();
            con.close();
        }catch(Exception e){
            System.out.println(e);
        }
        if(go)return "login.xhtml";
        return "marks.xhtml";
    }
}
```

***** login.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough"
      xmlns:f = "http://java.sun.com/jsf/core">
<head>
  <title>Login - JSF By SidPro</title>
```

```
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Login.css" rel="stylesheet" />
</head>

<body>
<div class="login">
       <h1>Login</h1>
       <h:form method="post">
            <h:message for = "Pass" style="color:#f24835" /><br/>
            <h:inputText type="text" value="#{student.firstName}"
name="firstName" id="firstname" a:placeholder="Username" required="true"
requiredMessage="Username Or Password is Wrong !"/><br/>
            <h:inputSecret type="password" value="#{student.password}"
name="password" id="Pass" a:placeholder="Password" required="true"
requiredMessage="Username Or Password is Wrong !"/><br/>
             <h:commandButton action="#{student.check()}" type="submit"
value="Let me in." class="btn btn-primary btn-block btn-large"/>
             <p>Don't have an account? <a href="register.xhtml"
target="_self">Sing up here!</a></p>
       </h:form>
</div>
</body>

</html>
```

***** marks.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough"
      xmlns:f = "http://java.sun.com/jsf/core">
<head>
  <title>Marks - JSF By SidPro</title>
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Collect_Info.css" rel="stylesheet" />
</head>

<body>
<h2 style="text-align:center">Marks Collection</h2>
<h:form method="post"  class="main_container">
            <div class="containerbase">

                    <h:outputLabel for="ADJ">Enter Mark of Subject 1
:</h:outputLabel>
                    <h:message for = "ADJ" style = "color:red" /><br/>
                    <h:inputText type="number" value="#{marks.ADJ}" name="ADJ"
id="ADJ" a:placeholder="Enter Mark of Advance Java Programming" min="0"
max="100" required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="DM">Enter Mark of Subject 2
:</h:outputLabel>
                    <h:message for = "DM" style = "color:red" /><br/>
```

```
                    <h:inputText type="number" value="#{marks.DM}" name="DM"
id="DM" a:placeholder="Enter Mark of Data mining" min="0" max="100"
required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="TOC">Enter Mark of Subject 3
:</h:outputLabel>
                    <h:message for = "TOC" style = "color:red" /><br/>
                    <h:inputText type="number" value="#{marks.TOC}" name="TOC"
id="TOC" a:placeholder="Enter Mark of Theory of Computation" min="0"
max="100" required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="MPI">Enter Mark of Subject 4
:</h:outputLabel>
                    <h:message for = "MPI" style = "color:red" /><br/>
                    <h:inputText type="number" value="#{marks.MPI}" name="MPI"
id="MPI" a:placeholder="Enter Mark of Microprocessor and Interfacing" min="0"
max="100" required="true" requiredMessage="This field is required"/><br/>
                    <h:outputLabel for="DV">Enter Mark of Subject 5
:</h:outputLabel>
                    <h:message for = "DV" style = "color:red" /><br/>
                    <h:inputText type="number" value="#{marks.DV}" name="DV"
id="DV" a:placeholder="Enter Mark of Data visualization" min="0" max="100"
required="true" requiredMessage="This field is required"/><br/>

                    <h:inputHidden value="#{marks.marDM}" type="text"
name="MarDM" id="MarDM" />
                    <h:inputHidden value="#{marks.marDV}" type="text"
name="MarDV" id="MarDV"/>
                    <h:inputHidden value="#{marks.marTOC}" type="text"
name="MarTOC" id="MarTOC"/>
                    <h:inputHidden value="#{marks.marMPI}" type="text"
name="MarMPI" id="MarMPI"/>
                    <h:inputHidden value="#{marks.marADJ}" type="text"
name="MarADJ" id="MarADJ"/>

                    <div class="container">
                         <h:commandButton action="grade" type="submit"
value="Submit" />
                         <h:commandButton type="reset" value="Reset" />
                    </div>
               </div>
      </h:form>

</body>

</html>
```

***** Marks.java ManagedBean *****

```
/**
 *
 */
package com.sidpro;

import javax.faces.bean.ManagedBean;

/**
 * @author MG
 *
```

```
 */
@ManagedBean
public class Marks {
    private int DM,ADJ,DV,TOC,MPI;
    private String
MarDM="DM",MarADJ="ADJ",MarDV="DV",MarTOC="TOC",MarMPI="MPI";
    /**
     *
     */
    public Marks() {
    }

    /**
     * @return the dM
     */
    public int getDM() {
        return DM;
    }

    /**
     * @param dM the dM to set
     */
    public void setDM(int dM) {
        DM = dM;
    }

    /**
     * @return the aDJ
     */
    public int getADJ() {
        return ADJ;
    }

    /**
     * @param aDJ the aDJ to set
     */
    public void setADJ(int aDJ) {
        ADJ = aDJ;
    }

    /**
     * @return the dV
     */
    public int getDV() {
        return DV;
    }

    /**
     * @param dV the dV to set
     */
    public void setDV(int dV) {
        DV = dV;
    }

    /**
     * @return the tOC
     */
```

```
    public int getTOC() {
        return TOC;
    }

    /**
     * @param tOC the tOC to set
     */
    public void setTOC(int tOC) {
        TOC = tOC;
    }

    /**
     * @return the mPI
     */
    public int getMPI() {
        return MPI;
    }

    /**
     * @param mPI the mPI to set
     */
    public void setMPI(int mPI) {
        MPI = mPI;
    }



    public String Grade(int s){
        if(s>=85)return "AA";
        else if(s>=75)return "AB";
        else if(s>=65)return "BB";
        else if(s>=55)return "BC";
        else if(s>=45)return "CC";
        else if(s>=40)return "CD";
        else if(s>=35)return "DD";
        return "FF";
    }

    /**
     * @return the marDM
     */

    /**
     * @param marDM the marDM to set
     */
    public void setMarDM(String marDM) {
        this.MarDM = marDM;
    }

    /**
     * @param marADJ the marADJ to set
     */
    public void setMarADJ(String marADJ) {
        MarADJ = marADJ;
    }

    /**
```

```
     * @param marDV the marDV to set
     */
    public void setMarDV(String marDV) {
        MarDV = marDV;
    }

    /**
     * @param marTOC the marTOC to set
     */
    public void setMarTOC(String marTOC) {
        MarTOC = marTOC;
    }

    /**
     * @param marMPI the marMPI to set
     */
    public void setMarMPI(String marMPI) {
        MarMPI = marMPI;
    }
    public String getMarADJ() {
        MarADJ = this.Grade(ADJ);
        return MarADJ;
    }




    /**
     * @return the marDM
     */


    /**
     * @return the marADJ
     */
    public String getMarDM() {
        MarDM = this.Grade(DM);
        return MarDM;
    }

    /**
     * @return the marDV
     */
    public String getMarDV() {
        MarDV = this.Grade(DV);
        return MarDV;
    }

    /**
     * @return the marTOC
     */
    public String getMarTOC() {
        MarTOC = this.Grade(TOC);
        return MarTOC;
    }

    /**
```

```
     * @return the marMPI
     */
    public String getMarMPI() {
        MarMPI = this.Grade(MPI);
        return MarMPI;
    }
}
```

***** grade.xhtml *****

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:a="http://xmlns.jcp.org/jsf/passthrough"
      xmlns:f = "http://java.sun.com/jsf/core">
<head>
  <title>Grades - JSF By SidPro</title>
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link href="/hello/resources/css/Collect_Info.css" rel="stylesheet" />
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.cs
s"
       integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous" />
</head>

<body>
      <h2>Grades </h2>
      #{Marks.DM}
      <table class="table table-dark">
            <tr>
                  <th>Subject</th>
                  <th>Grades</th>
            </tr>
            <tr>
                  <td>DM</td>
                  <td>#{marks.marDM}</td>
            </tr>
            <tr>
                  <td>DV</td>
                  <td>#{marks.marDV}</td>
            </tr>
            <tr>
                  <td>ADJ</td>
                  <td>#{marks.marADJ}</td>
            </tr>
            <tr>
                  <td>TOC</td>
                  <td>#{marks.marTOC}</td>
            </tr>
            <tr>
                  <td>MPI</td>
                  <td>#{marks.marMPI}</td>
```

```
            </tr>
            <tr>
                  <td>GTU</td>
                  <td>sucess is inevitable ha..ha..ha..!!!</td>
            </tr>
      </table>
</body>

</html>
```

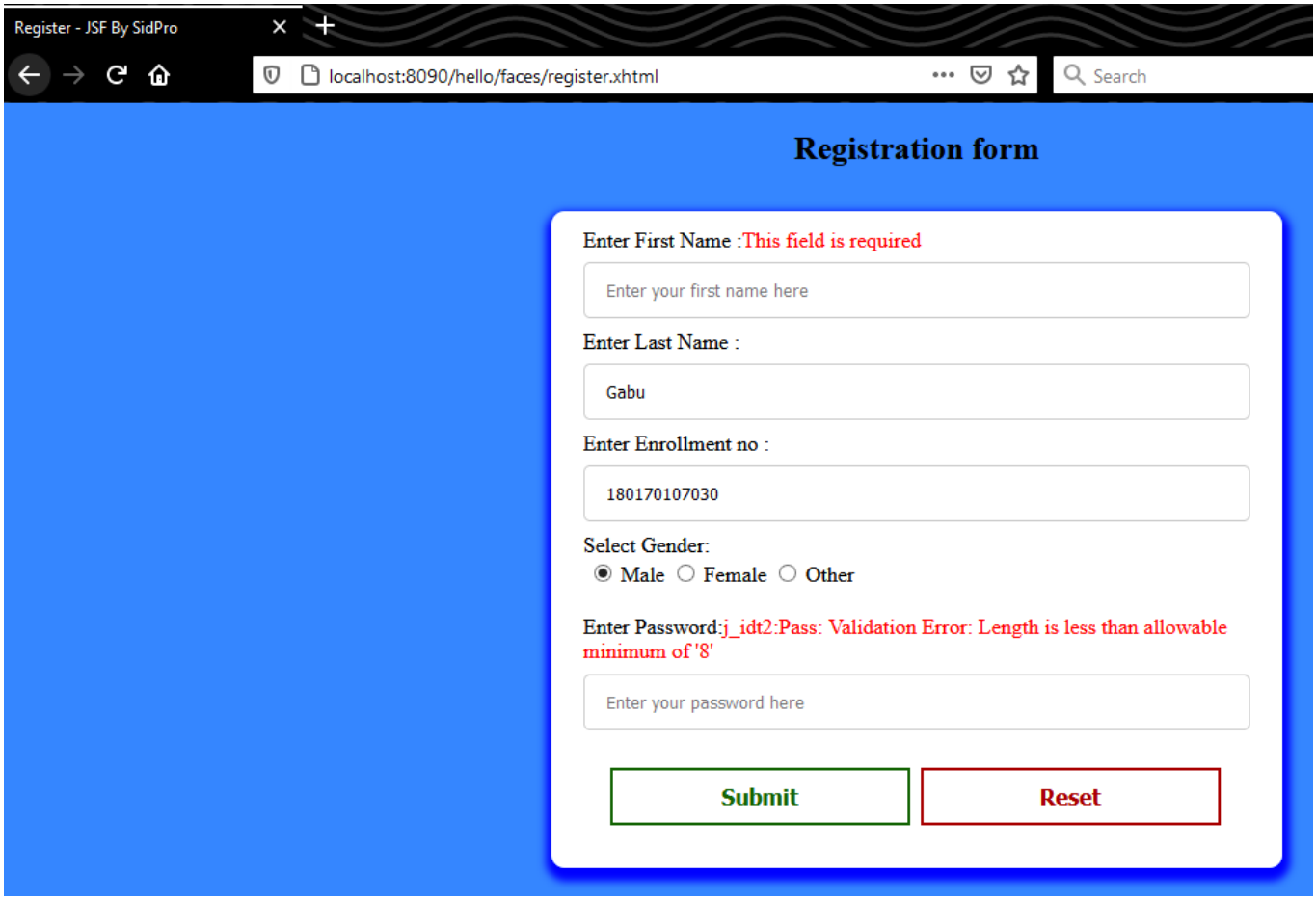


Fig6.1 – Registration validation

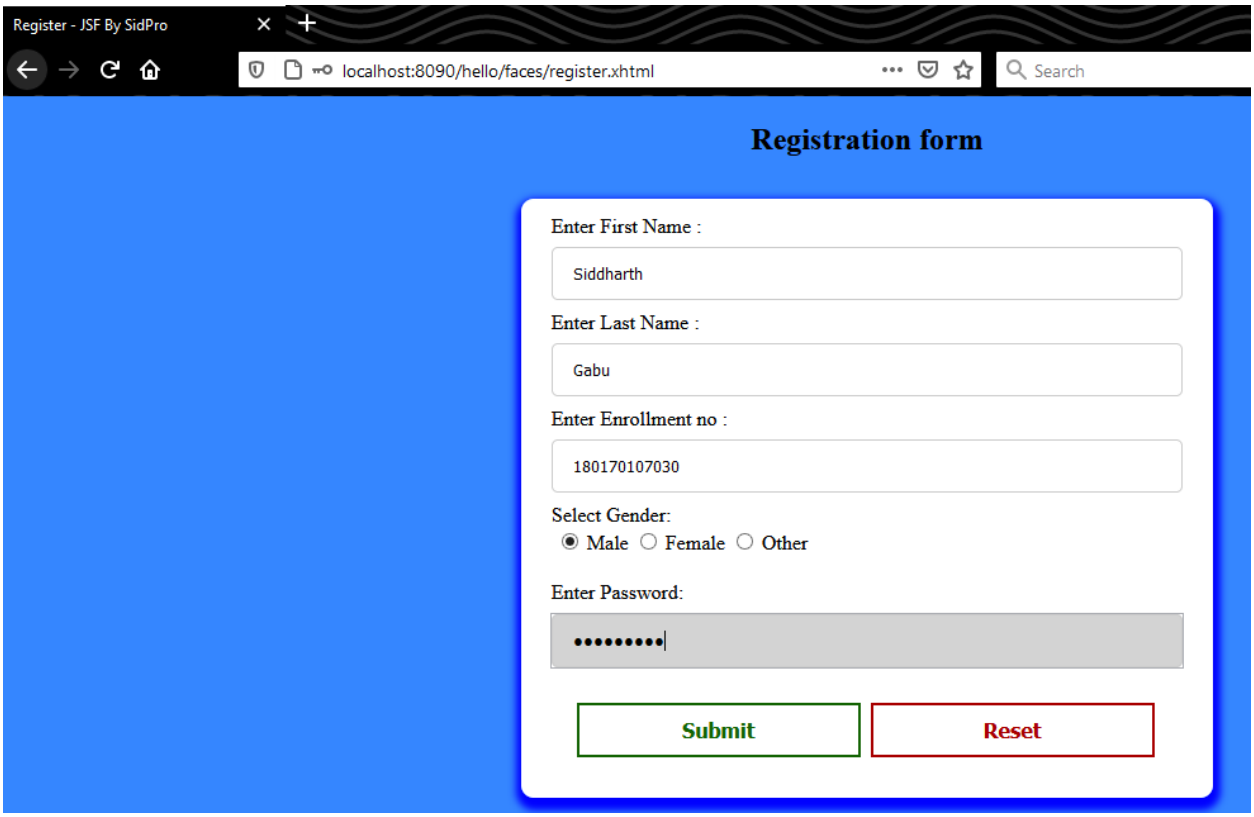


Fig6.2 - Registration

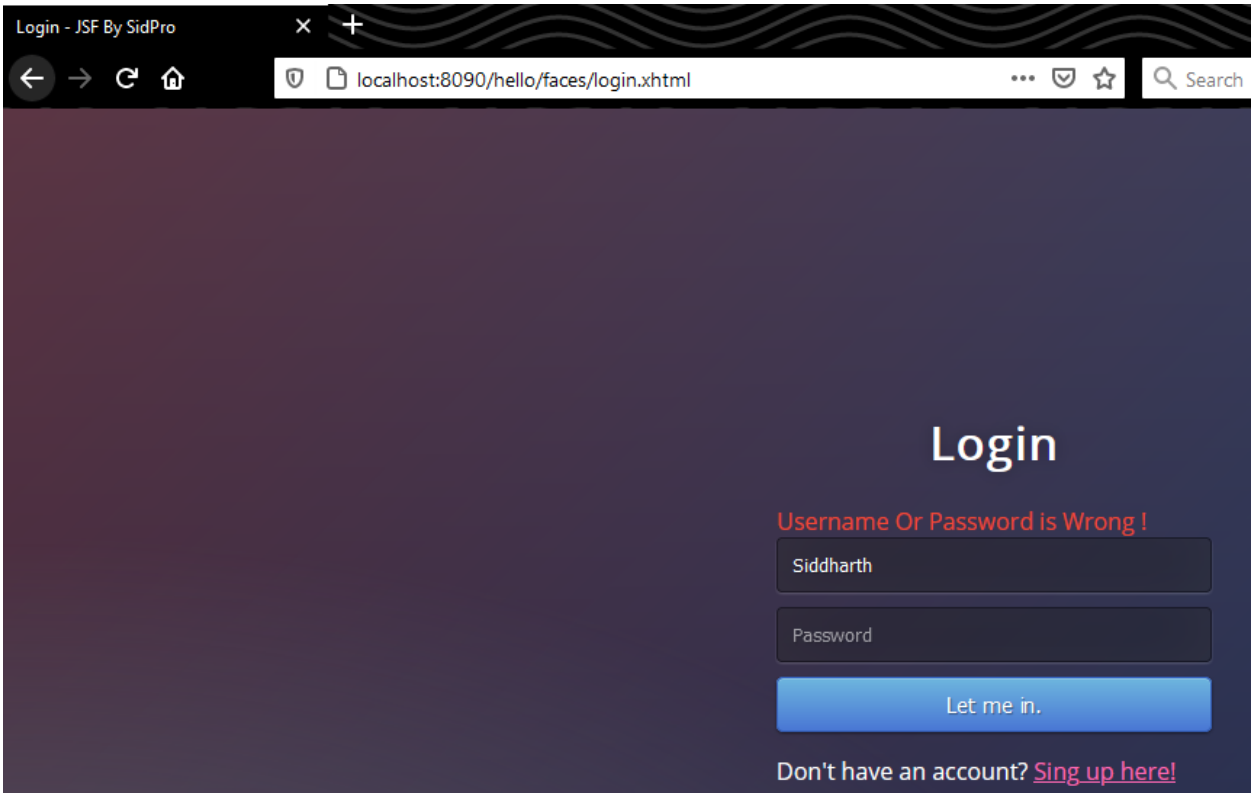


Fig6.3 – Login

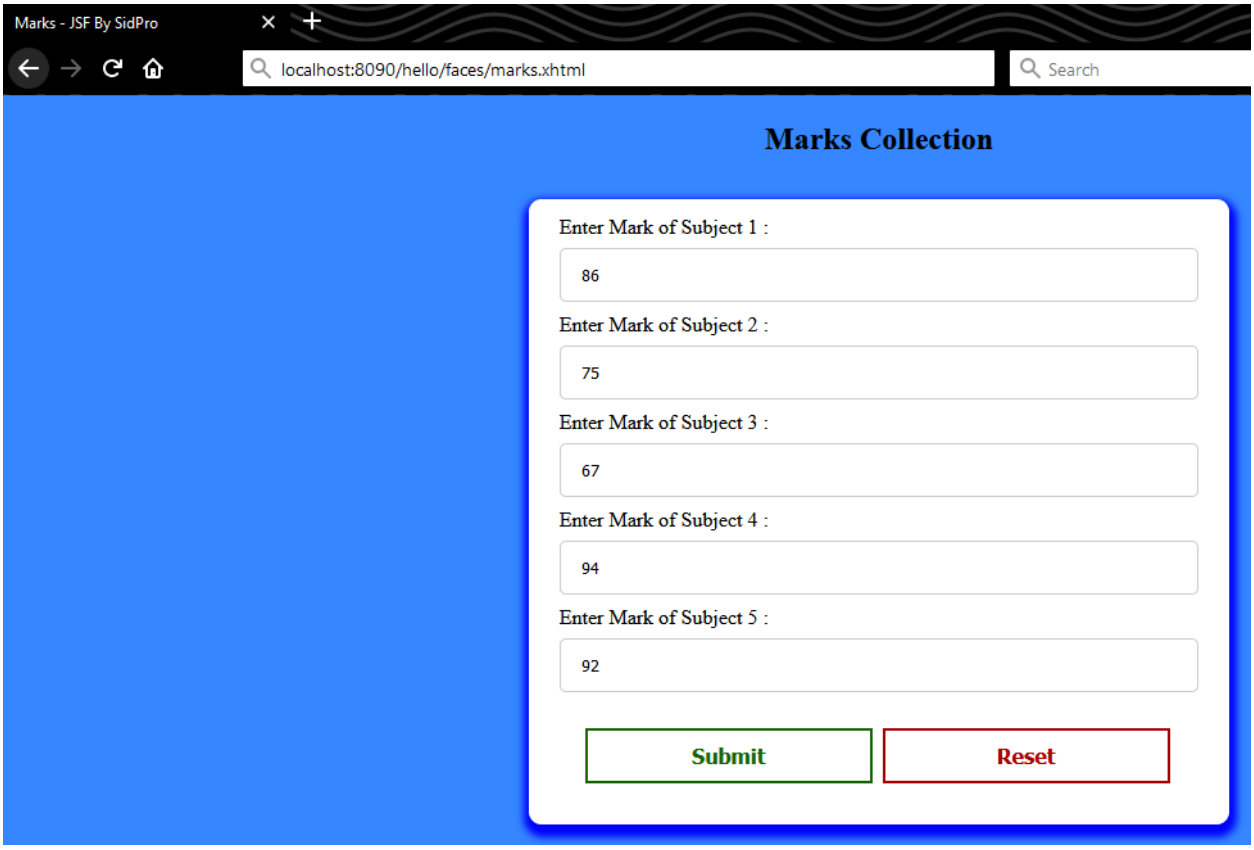
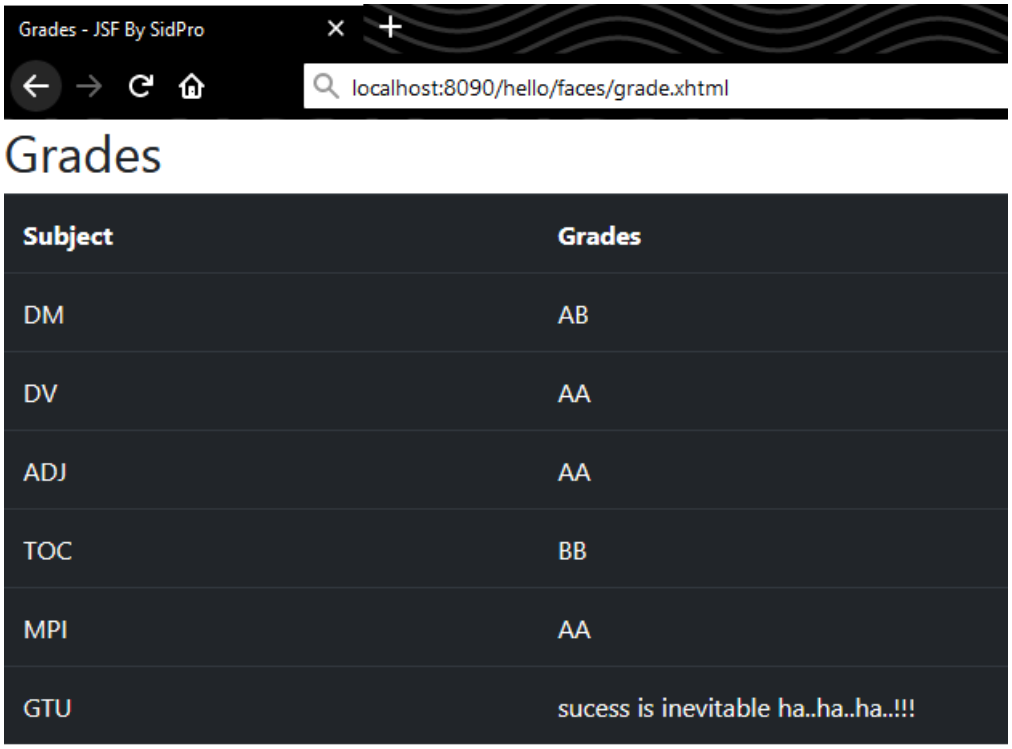


Fig6.4 – Mark Collection



Subject	Grades
DM	AB
DV	AA
ADJ	AA
TOC	BB
MPI	AA
GTU	sucess is inevitable ha..ha..ha..!!!

Fig 6.6 - Grades

Practical – 7

AIM: Create a java application using hibernate to use persistence object. Consider emp1000 table.

***** Persistence object Employee.java *****

```
/**
 *
 */
package javasem6;

/**
 * CREATE TABLE emp10000 (
 *  emp_id int PRIMARY KEY,
 *  emp_name varchar(255),
 *  job_name varchar(255),
 *  salary Number(15,2)
 * );
 *
 * @author Sidpro
 *
 */
public class Employee {
    private int emp_id;
    private String emp_name;
    private String job_name;
    private int salary;
    /**
     *
     */
    public Employee() {
    }
    /**
     * @return the emp_id
     */
    public int getEmp_id() {
        return emp_id;
    }
    /**
     * @param emp_id the emp_id to set
     */
    public void setEmp_id(int emp_id) {
        this.emp_id = emp_id;
    }
    /**
     * @return the emp_name
     */
    public String getEmp_name() {
        return emp_name;
    }
    /**
     * @param emp_name the emp_name to set
     */
    public void setEmp_name(String emp_name) {
        this.emp_name = emp_name;
```

```
    }
    /**
     * @return the job_name
     */
    public String getJob_name() {
        return job_name;
    }
    /**
     * @param job_name the job_name to set
     */
    public void setJob_name(String job_name) {
        this.job_name = job_name;
    }
    /**
     * @return the salary
     */
    public int getSalary() {
        return salary;
    }
    /**
     * @param salary the salary to set
     */
    public void setSalary(int salary) {
        this.salary = salary;
    }

}
```

***** StoreData.java *****

```
/**
 *
 */
package javasem6;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;
import org.hibernate.query.Query;

import java.util.List;
import java.util.Scanner;
/**
 * CREATE TABLE emp10000 (
 *  emp_id int PRIMARY KEY,
 *  emp_name varchar(255),
 *  job_name varchar(255),
 *  salary Number(15,2)
 * );
 *
 * @author Sidpro
 *
 */
public class StoreData {
    public static void main (String[] args) {
```

```
        String job_name="",emp_name="";
        int emp_id=106;
        int salary=50000;

        Scanner scan = new Scanner(System.in);

        System.out.print("Enter FirstName: ");
        emp_name = scan.nextLine();

        System.out.println("1 MANAGER");
        System.out.println("2 SALESMAN");
        System.out.println("3 CLERK");
        System.out.println("4 ANALYST");
        System.out.print("Select your Job 1/2/3/4 ?:");
        int option = scan.nextInt();
        switch(option){
            case 1:job_name="MANAGER";break;
            case 2:job_name="SALESMAN";break;
            case 3:job_name="CLERK";break;
            case 4:job_name="ANALYST";break;
            default:
                job_name="SALESMAN";
        }
        switch(option){
            case 1:salary=80000;break;
            case 2:salary=50000;break;
            case 3:salary=65000;break;
            case 4:salary=70000;break;
            default:
                salary=50000;
        }
        scan.close();


        // creating configuration object

        Configuration cfg = new Configuration();

        cfg.configure("hibernate.cfg.xml");

        try (SessionFactory factory = cfg.buildSessionFactory()) {
            Session session = factory.openSession();

            @SuppressWarnings("rawtypes")
            Query q = session.createQuery("SELECT emp_id FROM Employee ORDER
BY emp_id DESC");
            @SuppressWarnings("unchecked")
            List<Integer> list = q.list();
            emp_id = list.get(0);
            emp_id = emp_id + 1;
            System.out.println("emp_id "+emp_id);
            Transaction t = session.beginTransaction();
            Employee emp = new Employee();
            emp.setEmp_id(emp_id);
            emp.setEmp_name(emp_name);
            emp.setJob_name(job_name);
            emp.setSalary(salary);
```

```
            session.save(emp);
            t.commit();
            session.close();

            System.out.println("Succesfully saved");
        }
    }
}
```

***** hibernate.cfg.xml *****

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
          "-//Hibernate/Hibernate Configuration DTD 5.3//EN"
          "http://hibernate.sourceforge.net/hibernate-configuration-5.3.dtd">

    <hibernate-configuration>

        <session-factory>
            <!-- Update the exisiting tables -->
            <!-- prperty valuse can be: create,create-drop,update,validate --
>
            <property name="hbm2ddl.auto">update</property>
            <!-- Oracle dialect -->
            <property
name="dialect">org.hibernate.dialect.Oracle10gDialect</property>
            <!-- Database connection settings -->
            <property
name="connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
            <property
name="connection.url">jdbc:oracle:thin:@localhost:1521:XE</property>
            <property name="connection.username">admin</property>
            <property name="connection.password">admin</property>

            <!-- Echo all executed SQL to stdout -->
            <property name="show_sql">false</property>

            <mapping resource="employee.hbm.xml"/>
        </session-factory>

    </hibernate-configuration>
```

***** Employee.hbm.xml *****

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
5.3//EN" "http://hibernate.sourceforge.net/hibernate-mapping-5.3.dtd">
<hibernate-mapping>
  <class name="javasem6.Employee" table="emp10000">
    <id name="emp_id" column="emp_id">
     <generator class="assigned"></generator>
    </id>
     <!-- In the absence of a column attribute, Hibernate uses the property
name as the column name. -->
    <property name="emp_name"></property>
    <property name="job_name"></property>
    <property name="salary"></property>
```

```
  </class>
</hibernate-mapping>
```

```
Enter FirstName: Gabu
1 MANAGER
2 SALESMAN
3 CLERK
4 ANALYST
Select your Job 1/2/3/4 ?:1
Apr 16, 2021 8:53:02 AM org.hibernate.Version logVersion
INFO: HHH000412: Hibernate ORM core version [WORKING]
Apr 16, 2021 8:53:03 AM org.hibernate.boot.jaxb.internal.stax.LocalXmlResourceResolver resolveEntity
WARN: HHH90000012: Recognized obsolete hibernate namespace http://hibernate.sourceforge.net/hibernate-configuratic
Apr 16, 2021 8:53:03 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
Apr 16, 2021 8:53:04 AM org.hibernate.boot.jaxb.internal.stax.LocalXmlResourceResolver resolveEntity
WARN: HHH90000012: Recognized obsolete hibernate namespace http://hibernate.sourceforge.net/hibernate-mapping. Use
Apr 16, 2021 8:53:05 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl configu
WARN: HHH10001002: Using Hibernate built-in connection pool (not for production use!)
Apr 16, 2021 8:53:05 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCr
INFO: HHH10001005: using driver [oracle.jdbc.driver.OracleDriver] at URL [jdbc:oracle:thin:@localhost:1521:XE]
Apr 16, 2021 8:53:05 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCr
INFO: HHH10001001: Connection properties: {password=****, user=admin}
Apr 16, 2021 8:53:05 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCr
INFO: HHH10001003: Autocommit mode: false
Apr 16, 2021 8:53:05 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledC
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Apr 16, 2021 8:53:06 AM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.Oracle10gDialect
Apr 16, 2021 8:53:07 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvir
Apr 16, 2021 8:53:07 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateServic
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPl
emp_id 109
Succesfully saved
```

Fig7.1 – Store Data

Practical – 8

AIM: Write a java program which uses HQL to access records from emp1000 table.

***** Persistence object Employee.java *****

```
/**
 *
 */
package javasem6;

/**
 * CREATE TABLE emp10000 (
 *  emp_id int PRIMARY KEY,
 *  emp_name varchar(255),
 *  job_name varchar(255),
 *  salary Number(15,2)
 * );
 *
 * @author Sidpro
 *
 */
public class Employee {
    private int emp_id;
    private String emp_name;
    private String job_name;
    private int salary;
    /**
     *
     */
    public Employee() {
    }
    /**
     * @return the emp_id
     */
    public int getEmp_id() {
        return emp_id;
    }
    /**
     * @param emp_id the emp_id to set
     */
    public void setEmp_id(int emp_id) {
        this.emp_id = emp_id;
    }
    /**
     * @return the emp_name
     */
    public String getEmp_name() {
        return emp_name;
    }
    /**
     * @param emp_name the emp_name to set
     */
    public void setEmp_name(String emp_name) {
        this.emp_name = emp_name;
```

```
    }
    /**
     * @return the job_name
     */
    public String getJob_name() {
        return job_name;
    }
    /**
     * @param job_name the job_name to set
     */
    public void setJob_name(String job_name) {
        this.job_name = job_name;
    }
    /**
     * @return the salary
     */
    public int getSalary() {
        return salary;
    }
    /**
     * @param salary the salary to set
     */
    public void setSalary(int salary) {
        this.salary = salary;
    }

}
```

***** DisplayData.java *****

```
/**
 *
 */
package javasem6;

import java.util.Iterator;
import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

/**
 * CREATE TABLE emp10000 (
 *  emp_id int PRIMARY KEY,
 *  emp_name varchar(255),
 *  job_name varchar(255),
 *  salary Number(15,2)
 * );
 *
 * @author Sidpro
 *
 */
public class DisplayData {

    /**
     * @param args
```

```
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        // creating configuration object

                Configuration cfg = new Configuration();

                cfg.configure("hibernate.cfg.xml");

                try (SessionFactory factory = cfg.buildSessionFactory()) {
                    Session session = factory.openSession();


                    @SuppressWarnings("rawtypes")
                    List employees = session.createQuery("FROM Employee ORDER
BY emp_id ASC").list();

                     for (@SuppressWarnings("rawtypes")
                    Iterator iterator = employees.iterator();
iterator.hasNext();){
                        Employee employee = (Employee) iterator.next();
                        System.out.print("Employee Id: " +
employee.getEmp_id());
                        System.out.print("   First Name: " +
employee.getEmp_name());
                        System.out.print("   Job Name: " +
employee.getJob_name());
                        System.out.println("   Salary: " +
employee.getSalary());
                     }
                    session.close();
                }
    }

}
```

***** hibernate.cfg.xml *****

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
          "-//Hibernate/Hibernate Configuration DTD 5.3//EN"
          "http://hibernate.sourceforge.net/hibernate-configuration-5.3.dtd">

    <hibernate-configuration>

        <session-factory>
            <!-- Update the exisiting tables -->
            <!-- prperty valuse can be: create,create-drop,update,validate --
>
            <property name="hbm2ddl.auto">update</property>
            <!-- Oracle dialect -->
            <property
name="dialect">org.hibernate.dialect.Oracle10gDialect</property>
            <!-- Database connection settings -->
            <property
name="connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
```

```
            <property
name="connection.url">jdbc:oracle:thin:@localhost:1521:XE</property>
            <property name="connection.username">admin</property>
            <property name="connection.password">admin</property>

            <!-- Echo all executed SQL to stdout -->
            <property name="show_sql">false</property>

            <mapping resource="employee.hbm.xml"/>
        </session-factory>

    </hibernate-configuration>
```

***** Employee.hbm.xml *****

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
5.3//EN" "http://hibernate.sourceforge.net/hibernate-mapping-5.3.dtd">
<hibernate-mapping>
  <class name="javasem6.Employee" table="emp10000">
    <id name="emp_id" column="emp_id">
     <generator class="assigned"></generator>
    </id>
     <!-- In the absence of a column attribute, Hibernate uses the property
name as the column name. -->
    <property name="emp_name"></property>
    <property name="job_name"></property>
    <property name="salary"></property>
  </class>
</hibernate-mapping>
```

```
Employee Id: 100   First Name: Ravi   Job Name: MANAGER   Salary: 80000
Employee Id: 101   First Name: Jatin   Job Name: SALESMAN   Salary: 50000
Employee Id: 102   First Name: Pratik   Job Name: SALESMAN   Salary: 50000
Employee Id: 103   First Name: Puja   Job Name: CLERK   Salary: 65000
Employee Id: 104   First Name: Janu   Job Name: ANALYST   Salary: 70000
Employee Id: 105   First Name: Garima   Job Name: SALESMAN   Salary: 50000
Employee Id: 106   First Name: Bolt   Job Name: SALESMAN   Salary: 50000
Employee Id: 107   First Name: Husen   Job Name: ANALYST   Salary: 70000
Employee Id: 108   First Name: Chiku   Job Name: ANALYST   Salary: 70000
```

Fig8.1 – Display Data

Practical – 9

AIM: Implement a simple hello world web application using spring.

***** app-servlet.xml *****

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:ctx="http://www.springframework.org/schema/context"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:mvc="http://www.springframework.org/schema/mvc"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
      http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
      http://www.springframework.org/schema/mvc
      http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd
      http://www.springframework.org/schema/context
      http://www.springframework.org/schema/context/spring-context-2.5.xsd">
      <!-- Provide support for component scanning -->
      <ctx:component-scan base-package="com.sidpro" />
      <!--Provide resources mapping -->

      <bean id="viewResolver" class =
"org.springframework.web.servlet.view.InternalResourceViewResolver" >
        <property name="prefix">
            <value>/WEB-INF/</value>
        </property>
        <property name ="suffix">
            <value>.jsp</value>
        </property>
    </bean>
    <mvc:annotation-driven/>
    <mvc:resources mapping="/resources/**" location="/WEB-INF/resources/"/>

    <!-- Define Spring MVC view resolver -->

</beans>
```

***** index.jsp *****

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
      <title> Spring MVC By Sid </title>
      <meta charset="ISO-8859-1">
      <meta name="author" content="SidPro"/>
      <meta name="viewport" content="width=device-width,initial-scale=1" />
      <link rel="stylesheet" href="resources/Collect_Info.css"
type="text/css" />
      <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
  </script>
  <style>


  .topnav{
```

```
    display:block;
    background-color: #262626;
  }
  a{
    text-decoration:none;
    display: inline-block;
    padding: 14px 20px;
    font-size: 18px;
    color:white;
    background-color: #262626;
  }
  .active{
    background-color: white;
    color: black;
    font-weight: bold;
    box-shadow:0px 4px 8px 0 red;
  }
  .vide{
    position: absolute;
    top : 100px;
    right : 20px;
    outline: 2px solid gray;
    background-color: rgba(255, 0, 0, 0.5);
    z-index: auto;
  }
  </style>
<script>

$(document).ready(function(){
  $("#button4").click(function(){
    $("#div1,#div2,#div3").stop();
  });
  $("#button1").click(function(){
    $("p").toggle();
  });
  $("#button2").click(function(){
    $("#div1").fadeToggle();
    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(2000);
  });
  $("#button3").click(function(){
    $("#div4").animate({left:'250px'},"slow");
    $("#div4").animate({left:'10px'},"slow");
  });
  $("a").click(function(){
    $(this).addClass("active").siblings(this).removeClass("active")});
});
</script>
</head>

<body>

<div class="topnav">
  <a class="active" href="#home">Home</a>
  <a href="#About">About</a>
  <a href="#Registration">Registration</a>
  <a href="#Login">Login</a>
```

```
</div>
      <h1 style="font-size:48px;text-align:center;color:red">Hello
world!</h1>
      <form method="post" action="hello" class="main_container">
      <label for="firstname">Enter FirstName: </label>
            <input type="text" id="firstname" name="firstname"
placeholder="What's your name?"/>
            <input type="submit" value="Submit"/>
      </form>
<p>This is Spring MVC Hello World Program!.</p>
<p>POJO = Plain Old Java Object </p>
<button id="button4">Stop</button><br/>
<button id="button1">Hide</button><br/>
<button id="button2">Show</button><br/>
<button id="button3">Boomerang</button>
<div id="div1" style="width:80px;height:80px;display:none;background-
color:yellow">
</div><br/>
<div id="div2" style="width:80px;height:80px;display:none;background-
color:red">
</div><br/>
<div id="div3" style="width:80px;height:80px;display:none;background-
color:blue;">
</div>
<div id="div4" style="width:80px;height:80px;background-
color:#ccc;position:absolute;">
</div>


</body>

</html>
```

***** helloworld.jsp *****

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
      <title> Spring MVC By Sid </title>
      <meta charset="ISO-8859-1">
      <meta name="author" content="SidPro"/>
      <meta name="viewport" content="width=device-width,initial-scale=1">
      <link type="text/css" rel="stylesheet"
href="resources/Collect_Info.css">
      <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
  </script>
  <style>

  .topnav{
    display:block;
    background-color: #262626;
  }
  a{
    text-decoration:none;
```

```
    display: inline-block;
    padding: 14px 20px;
    font-size: 18px;
    color:white;
    background-color: #262626;
  }
  .active{
    background-color: white;
    color: black;
    font-weight: bold;
    box-shadow:0px 4px 8px 0 red;
  }
  .vide{
    position: absolute;
    top : 100px;
    right : 20px;
    outline: 2px solid gray;
    background-color: rgba(255, 0, 0, 0.5);
    z-index: auto;
  }
  </style>
<script>

$(document).ready(function(){
  $("#button4").click(function(){
    $("#div1,#div2,#div3").stop();
  });
  $("#button1").click(function(){
    $("p").toggle();
  });
  $("#button2").click(function(){
    $("#div1").fadeToggle();
    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(2000);
  });
  $("#button3").click(function(){
    $("#div4").animate({left:'250px'},"slow");
    $("#div4").animate({left:'10px'},"slow");
  });
  $("a").click(function(){
    $(this).addClass("active").siblings(this).removeClass("active")});
});
</script>
</head>

<body>

<div class="topnav">
  <a class="active" href="#home">Home</a>
  <a href="#About">About</a>
  <a href="#Registration">Registration</a>
  <a href="#Login">Login</a>
</div>
      <h1 style="font-size:48px;text-align:center;color:red">Hello, <%=
request.getAttribute("Name") %></h1>
<p>This is Spring MVC Hello World Program!.</p>
<p>POJO = Plain Old Java Object </p>
```

```
<button id="button4">Stop</button><br/>
<button id="button1">Hide</button><br/>
<button id="button2">Show</button><br/>
<button id="button3">Boomerang</button>
<div id="div1" style="width:80px;height:80px;display:none;background-
color:yellow">
</div><br/>
<div id="div2" style="width:80px;height:80px;display:none;background-
color:red">
</div><br/>
<div id="div3" style="width:80px;height:80px;display:none;background-
color:blue;">
</div>
<div id="div4" style="width:80px;height:80px;background-
color:#ccc;position:absolute;">
</div>


</body>

</html>
```

***** AddController.java *****

```
/**
 *
 */
package com.sidpro;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
/**
 * @author MG
 *
 */
@RequestMapping("/hellow")
@Controller
public class AddController {
    @RequestMapping("/hello")
    public String hello() {
        //System.out.println("Hello World!");
        return "helloworld.jsp";
    }
}
```

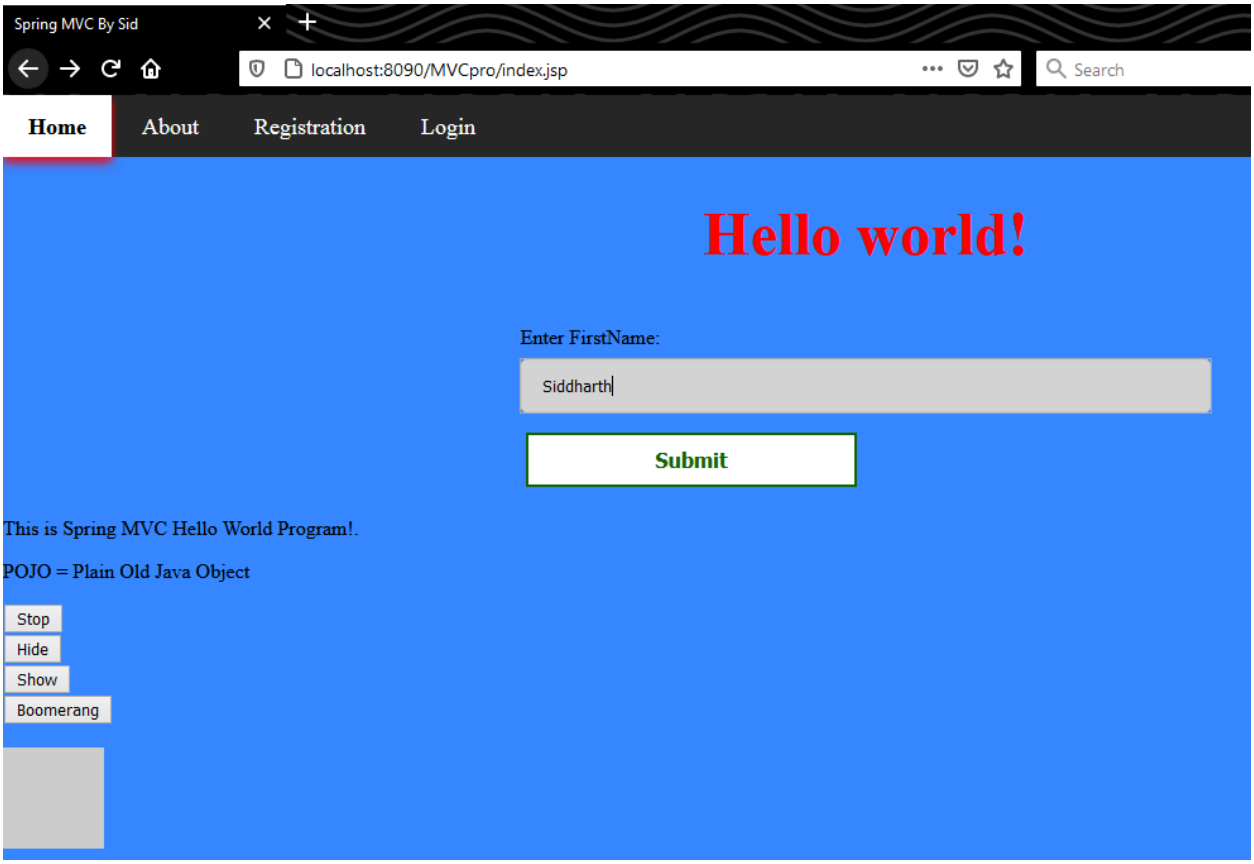
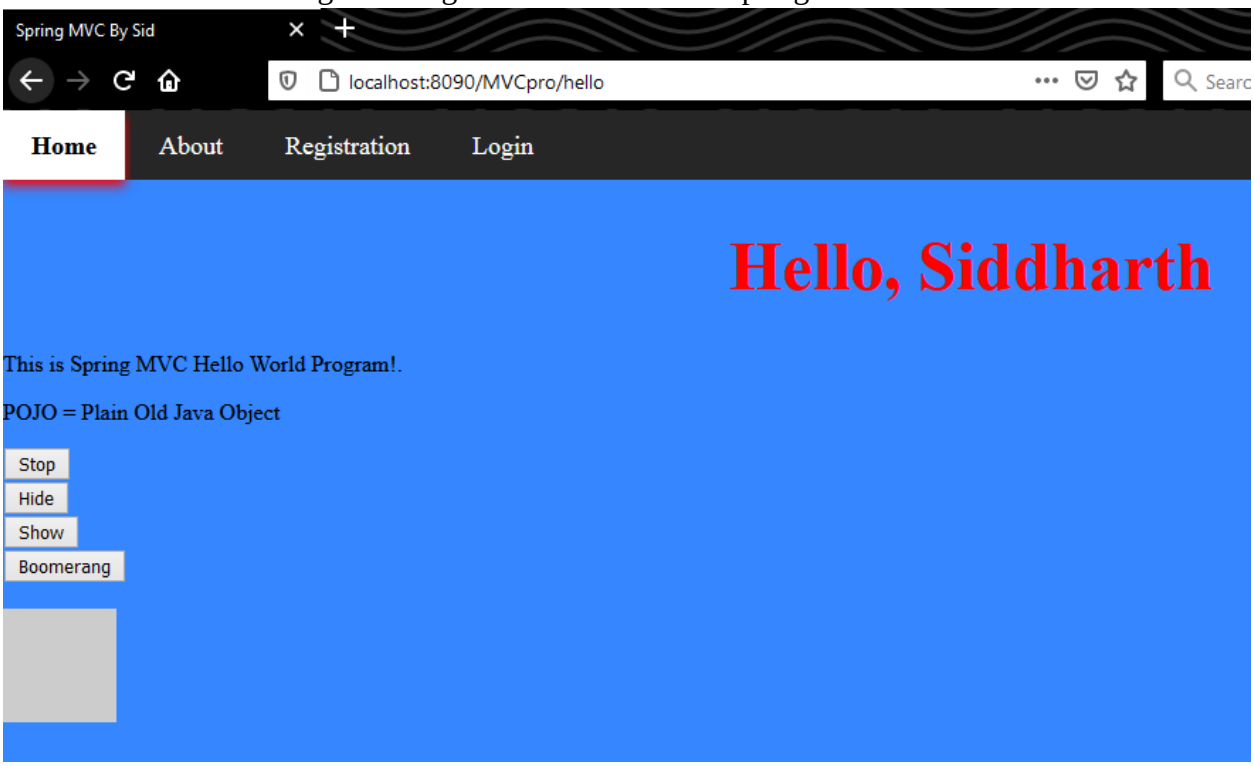


Fig9.1 & Fig9.2 – Hello world in Spring MVC



Practical – 10

AIM: Write a spring application for user Registration which forward to login page if successful registration. Create a login page to login using registered user credentials. If valid user accept the marks of five subjects and then print the grade of student. The registered information must be stored in database.

***** register_user.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
       <title> Jsp By Sid </title>
       <meta charset="utf-8">
       <meta name="author" content="SidPro"/>
       <meta name="viewport" content="width=device-width,initial-scale=1">
       <link type="text/css" rel="stylesheet"
href="resources/Collect_Info.css">
 </head>

 <body>
      <%
            String fname ="",lname="",pass="",enNo="";
            if(request.getAttribute("fname_error")!=null){
                  fname=(String)request.getAttribute("fname_error");
                  request.removeAttribute("fname_error");
            }
            if(request.getAttribute("lname_error")!=null){
                  lname=(String)request.getAttribute("lname_error");
                  request.removeAttribute("lname_error");
            }
            if(request.getAttribute("enNo_error")!=null){
                  enNo=(String)request.getAttribute("enNo_error");
                  request.removeAttribute("enNo_error");
            }
            if(request.getAttribute("pass_error")!=null){
                  pass=(String)request.getAttribute("pass_error");
                  request.removeAttribute("pass_error");
            }
      %>
      <h2>Collecting Employee information Of Collegeek</h2>

      <form method="post" action="validate_user" class="main_container">
      <div class="containerbase">
            <lable id="F" for="firstname">Enter First Name :<%= fname
%></lable><br>
            <input type="text" name="firstName" id="firstname"
placeholder="Enter your first name here"><br>
            <lable id="L" for="lastname">Enter Last Name :<%= lname
%></lable><br>
            <input type="text" name="lastName" id="lastname"
placeholder="Enter your last name here"><br>
            <lable id="E" for="enNo">Enter Enrollment no :<%= enNo
%></lable><br>
```

```
            <input type="text" name="enNo" id="enNo" placeholder="Enter your
Enrollment no here"><br>
            <lable>Select Gender:</lable><br>
            <input type="radio" name="gender" value="Male" checked>Male
            <input type="radio" name="gender" value="Female">Female
            <input type="radio" name="gender" value="other">other<br>

            <lable id="P" for="Pass">Enter Password:<%= pass %></lable><br>
            <input type="password" name="password" id="Pass"
placeholder="Enter your password here"><br>
            <div class="container">
            <input type="submit" value="Submit"><input type="reset"
value="Reset">
            </div>
      </div>
      </form>

      <section>
            <div class="wave wave1"></div>
            <div class="wave wave2"></div>
            <div class="wave wave3"></div>
      </section>
</body>
</html>
```

***** RegisterData.java *****

```
package com.sidpro;

import java.sql.*;

public class RegisterData {
    private String Email,Name,Password;

    public RegisterData(String Email,String Name,String Password)
    {
        this.Email=Email;
        this.Name=Name;
        this.Password=Password;
    }
    public RegisterData(String Email,String Password)
    {
        this.Email=Email;
        this.Password=Password;
    }

    public int send_data()
    {
        try {
            Connection con = DatabaseConnection.initializeDatabase();
            PreparedStatement st = con
                    .prepareStatement("insert into student_data values(?, ?,
?, ?)");
            st.setString(1,Email);
            st.setString(2,Name);
            st.setString(3, Password);
            st.setInt(4,0);
```

```
            st.executeUpdate();
            st.close();
            con.close();

            return 1;

        } catch (ClassNotFoundException | SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            if(e.getMessage().startsWith("Duplicate entry"))
            {
                return 2;
            }
            else
            {
                return 3;
            }

        }
    }


    public String Check_Pass()
    {
        try {
            Connection con = DatabaseConnection.initializeDatabase();
            PreparedStatement ps=con.prepareStatement("select Pass,S_name
from student_data where Email_Id=?");
            ps.setString(1,Email);
            ResultSet rs=ps.executeQuery();
            if(rs.next())
            {
                if(Password.equals(rs.getString(1)))
                {
                    String Name=rs.getString(2);
                    ps.close();
                    con.close();
                    return Name;

                }else
                {
                    ps.close();
                    con.close();
                    return "2";

                }
            }else
            {
                ps.close();
                con.close();
                return "3";
            }

//          response.getWriter().print(rs.getString(1));
```

```
        } catch (ClassNotFoundException | SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            return "4";
        }

    }
}
```

***** RegisterController.java *****

```
package com.sidpro;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;

import com.sidpro.RegisterData;

@Controller
public class RegistarionController {

    @RequestMapping("/registration")
    public String handler(HttpServletRequest request,HttpServletResponse
response)
    {
        String Name =request.getParameter("name");
        String Email=request.getParameter("email");
        String Pass=request.getParameter("password");
        RegisterData data=new RegisterData(Email,Name,Pass);
        int Result=data.send_data();
        HttpSession session = request.getSession();
        if(Result==1)
        {
            return "login";
        }else if(Result==2)
        {
            session.setAttribute("error", Email+" Id is already Exist");
            return "error";
        }else
        {
            session.setAttribute("error", "Some thing Went's Wrong");
            return "error";
        }

    }

    @RequestMapping("/login_data_check")
    public String login_data(HttpServletRequest request,HttpServletResponse
response)
    {
        String Email=request.getParameter("email");
        String Pass=request.getParameter("password");
```

```
        RegisterData data=new RegisterData(Email,Pass);
        HttpSession session = request.getSession();
        String Result=data.Check_Pass();
        if(Result.equals("2"))
        {
            session.setAttribute("error", "Your Enter Password is Wrong");
            return "login";
        }else if(Result.equals("3"))
        {
            session.setAttribute("error", "Your Enter Email Id is not
Registered");
            return "login";
        }else if(Result.equals("4"))
        {
            session.setAttribute("error", "Something Went's Wrong");
            return "error";
        }else
        {
            session.setAttribute("Username", Email);
            session.setAttribute("name", Result);
            return "welcome";
        }

    }

    @RequestMapping("/calculate_Result")
    public String calculateResult(HttpServletRequest
request,HttpServletResponse response)
    {
        int Total, GET;
        float pr;
        int T1 = Integer.parseInt(request.getParameter("T1"));
        int T2 = Integer.parseInt(request.getParameter("T2"));
        int T3 = Integer.parseInt(request.getParameter("T3"));
        int T4 = Integer.parseInt(request.getParameter("T4"));
        int T5 = Integer.parseInt(request.getParameter("T5"));

        int G1 = Integer.parseInt(request.getParameter("G1"));
        int G2 = Integer.parseInt(request.getParameter("G2"));
        int G3 = Integer.parseInt(request.getParameter("G3"));
        int G4 = Integer.parseInt(request.getParameter("G4"));
        int G5 = Integer.parseInt(request.getParameter("G5"));

        if(T1<=0 && T2<=0 && T3<=0 && T4<=0 && T5<0)
        {
            HttpSession session = request.getSession();
            session.setAttribute("error", "Totla Masrk Must be non Zero
positive Integr");
            return "error";
        }else
        {
            Total= T1+T2+T3+T4+T5;
            GET= G1+G2+G3+G4+G5;
            pr=(float)GET/Total*100;
            HttpSession session = request.getSession();
            session.setAttribute("total", String.valueOf(Total));
            session.setAttribute("get", String.valueOf(GET));
```

```
            session.setAttribute("per", String.valueOf(pr));
            return "result";
        }



    }



    @RequestMapping("/login")
    public String LoginIn(Model model)
    {
        return "login";
    }

    @RequestMapping("/welcome")
    public String welcome(Model model)
    {
        return "welcome";
    }

    @RequestMapping("/logOut")
    public String LogOut(HttpServletRequest request,HttpServletResponse
response)
    {
        HttpSession session = request.getSession();
        session = request.getSession();
        session.removeAttribute("Username");
        session.invalidate();
        return "login";
    }

}
```

***** DatabaseConnection.java *****

```
package com.sidpro;


import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Connection;

public class DatabaseConnection {
    public static Connection initializeDatabase()
        throws SQLException, ClassNotFoundException
    {
        // Initialize all the information regarding
        // Database Connection
        String dbDriver = "oracle.jdbc.driver.OracleDriver";
        String dbURL = "jdbc:oracle:thin:@localhost:1521:XE";
        // Database name to access
        String dbName = "student";
        String dbUsername = "admin";
        String dbPassword = "admin";
```

```
        Class.forName(dbDriver);
        Connection con = DriverManager.getConnection(dbURL + dbName,
                                                     dbUsername,
                                                     dbPassword);
        return con;
    }
}
```

***** HelloController.java *****

```
package net.javaguides.springmvc.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.view.InternalResourceViewResolver;
import org.springframework.web.servlet.view.JstlView;

@Configuration
@EnableWebMvc
@ComponentScan(basePackages  = {"net.javaguides.springmvc"})
public class AppConfig {
    @Bean
    public InternalResourceViewResolver resolver()
    {
        InternalResourceViewResolver resolver=new
InternalResourceViewResolver();
        resolver.setViewClass(JstlView.class);
        resolver.setPrefix("/WEB-INF/views/");
        resolver.setSuffix(".jsp");
        return resolver;
    }

}
```

***** Login.jsp *****

```
<!DOCTYPE html>
 <html>

 <head>
       <title> Jsp By Sid </title>
       <meta charset="utf-8">
       <meta name="author" content="SidPro"/>
       <meta name="viewport" content="width=device-width,initial-scale=1">
       <link type="text/css" rel="stylesheet" href="resources/Login.css">
 </head>

 <body>
       <div class="login">
       <h1>Login</h1>
       <% String fname ="";
            if(request.getAttribute("fname_error")!=null){
                  fname=(String)request.getAttribute("fname_error");
                  request.removeAttribute("fname_error");
            } %>
```

```
        <span style="color:#f24835"><%= fname %></span>
        <form action="validate_Login" method="post">
        <input type="text" name="uname" placeholder="Username"
required="required" />
        <input type="password" name="pass" placeholder="Password"
required="required" />
        <button type="submit" class="btn btn-primary btn-block btn-large">Let
me in.</button>
        <p>Don't have an account? <a href="register_user.jsp"
target="_self">Sing up here!</a></p>
        </form>
        </div>

 </body>
 </html>
```

***** pom.xml *****

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.sidpro</groupId>
  <artifactId>MVCpro</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

  <name>MVCpro Maven Webapp</name>
  <!-- FIXME change it to the project's website -->
  <url>http://www.example.com</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.7</maven.compiler.source>
    <maven.compiler.target>1.7</maven.compiler.target>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-
context -->
      <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-context</artifactId>
          <version>5.3.6</version>
      </dependency>
```

```
    <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc
-->
      <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-webmvc</artifactId>
          <version>5.3.6</version>
      </dependency>
    <!-- https://mvnrepository.com/artifact/com.oracle.database.jdbc/ojdbc6 -
->
      <dependency>
          <groupId>com.oracle.database.jdbc</groupId>
          <artifactId>ojdbc6</artifactId>
          <version>11.2.0.4</version>
      </dependency>
      <!-- https://mvnrepository.com/artifact/javax.servlet.jsp.jstl/jstl -->
      <dependency>
          <groupId>javax.servlet</groupId>
          <artifactId>jstl</artifactId>
          <version>1.2</version>
      </dependency>
      <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api
-->
      <dependency>
          <groupId>javax.servlet</groupId>
          <artifactId>javax.servlet-api</artifactId>
          <version>4.0.1</version>
          <scope>provided</scope>
      </dependency>
  </dependencies>

  <build>
    <finalName>MVCpro</finalName>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven
defaults (may be moved to parent pom) -->
      <plugins>
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.1.0</version>
        </plugin>
        <!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_war_packaging -->
        <plugin>
          <artifactId>maven-resources-plugin</artifactId>
          <version>3.0.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>3.8.0</version>
        </plugin>
        <plugin>
          <artifactId>maven-surefire-plugin</artifactId>
          <version>2.22.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-war-plugin</artifactId>
          <version>3.2.2</version>
        </plugin>
```

```
        <plugin>
          <artifactId>maven-install-plugin</artifactId>
          <version>2.5.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-deploy-plugin</artifactId>
          <version>2.8.2</version>
        </plugin>
      </plugins>
    </pluginManagement>
  </build>
</project>
```

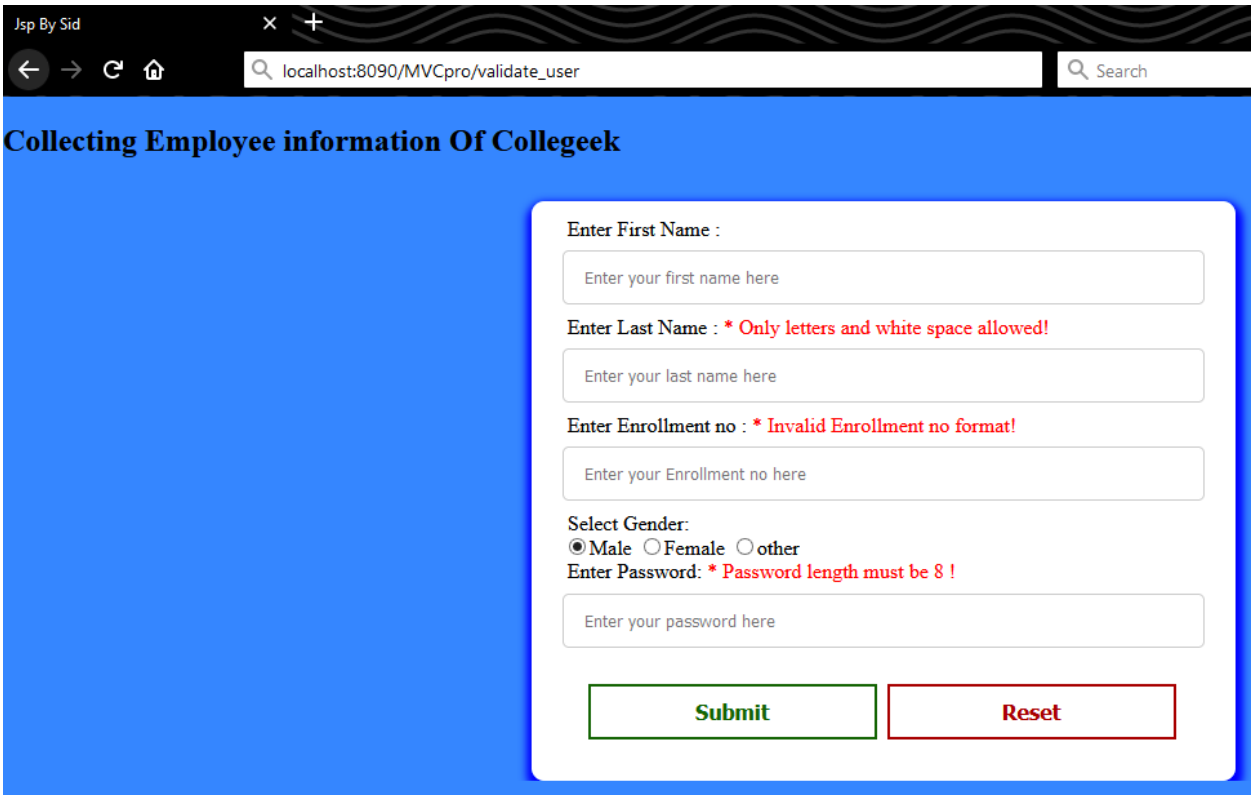


Fig10.1 – Invalid Login

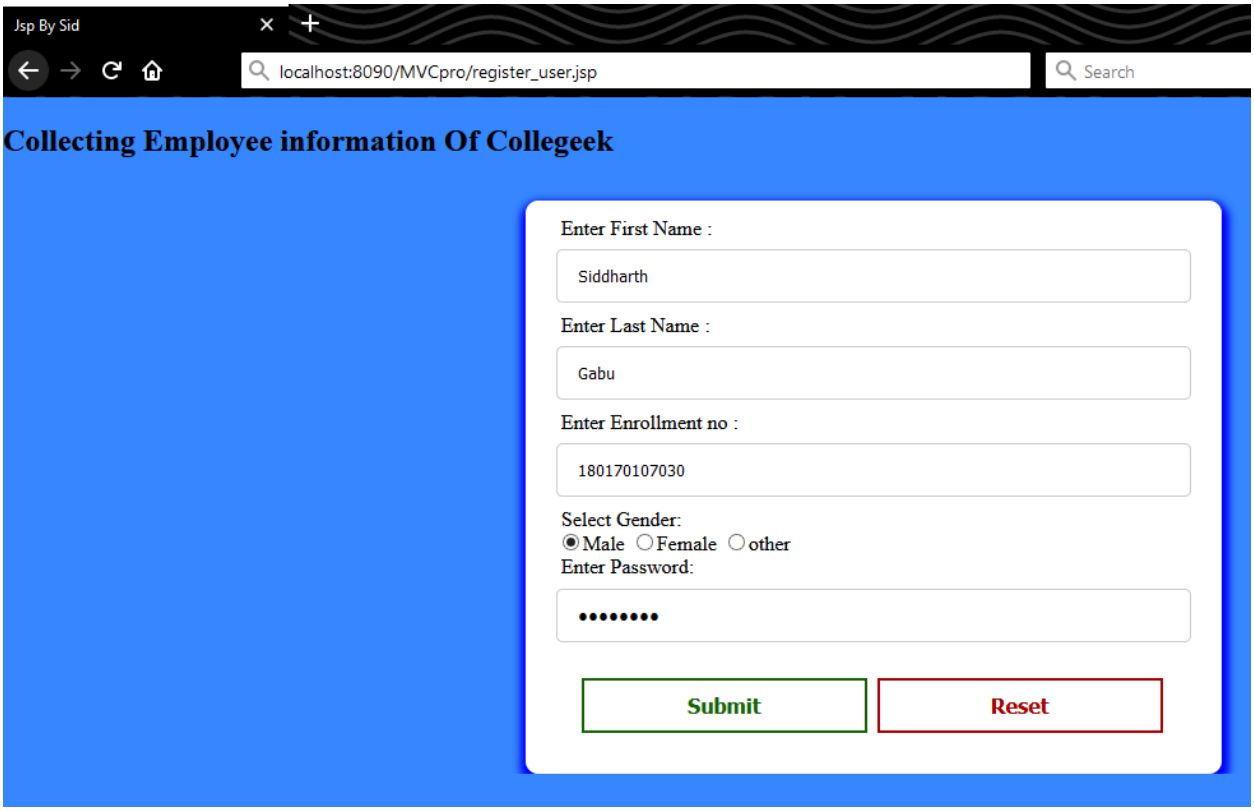


Fig10.2 – register user

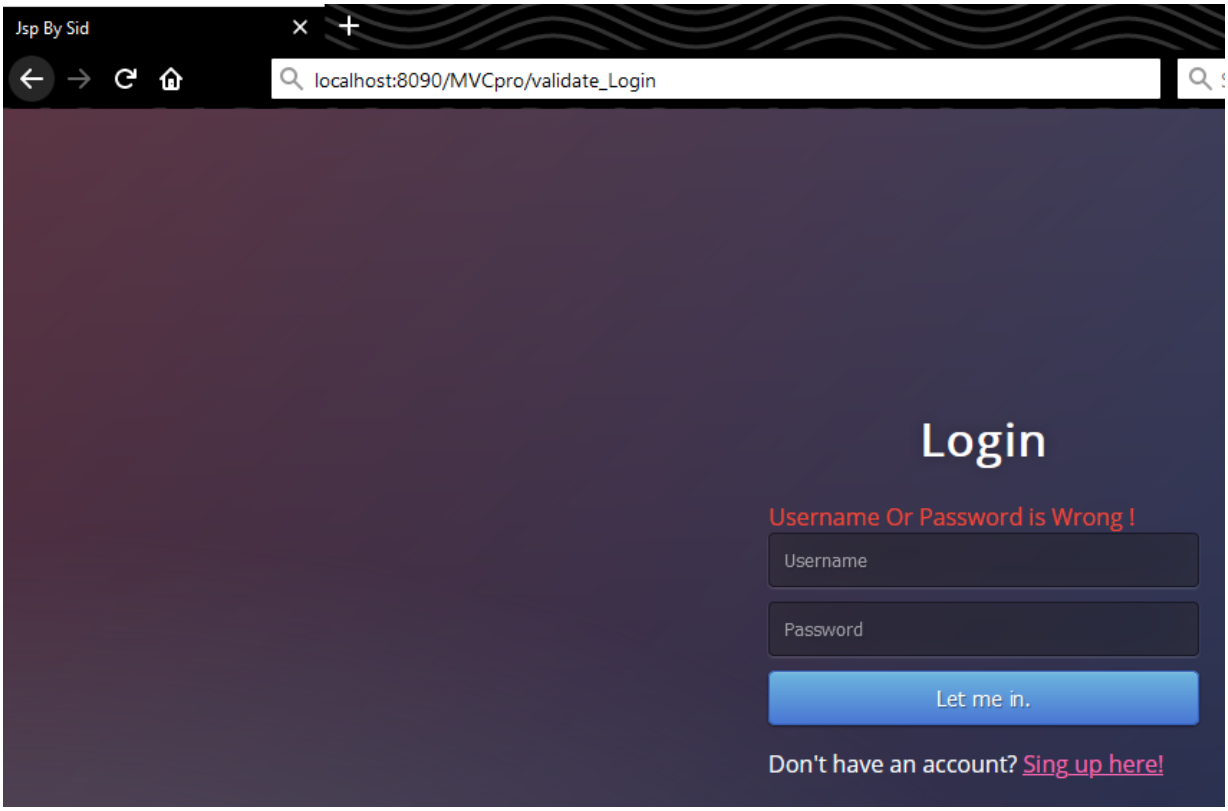


Fig10.3 – invalid Login

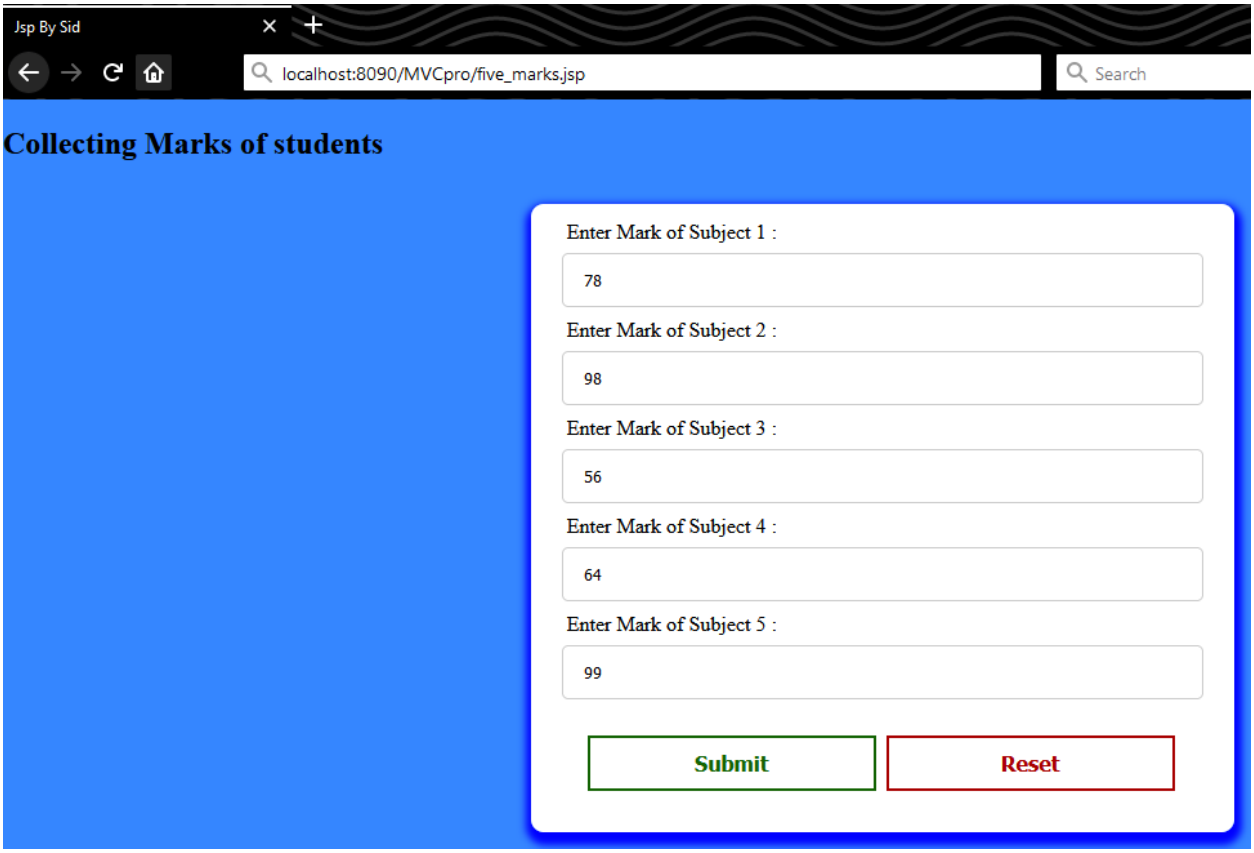
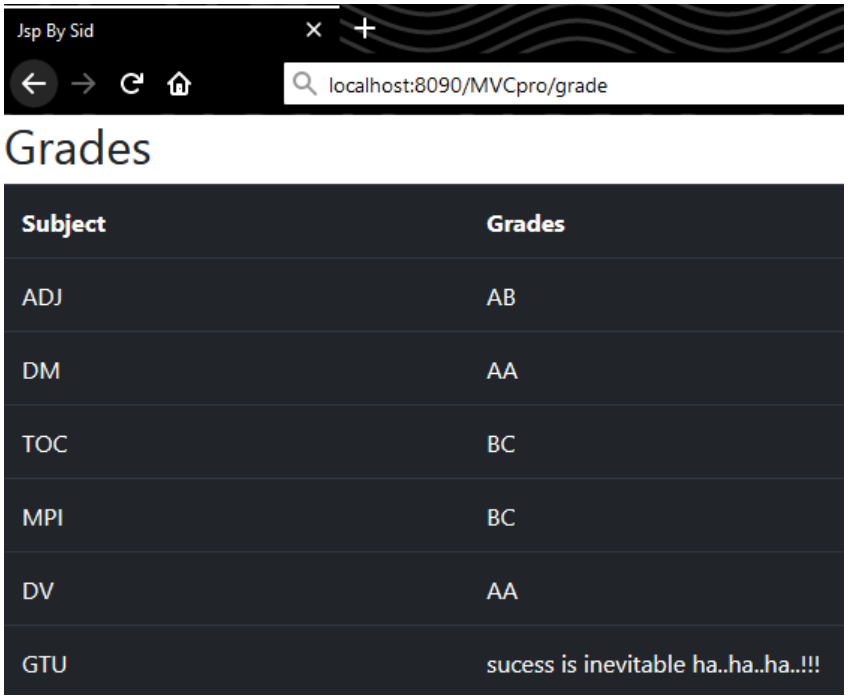


Fig10.4 – Collecting Marks



Subject	Grades
ADJ	AB
DM	AA
TOC	BC
MPI	BC
DV	AA
GTU	sucess is inevitable ha..ha..ha..!!!

Fig10.5 - Grades