THAKUR COLLEGE OF SCIENCE & COMMERCE

NAAC Accredited with Grade "A" (3rd Cycle)

THAKUR TRUSTS

ISO 9001 : 2015 Certified

Best College Award by University of Mumbai for the Year 2018-2019

Degree College

Computer Journal

CERTIFICATE

SEMESTER IV **UID No.** 2020858

Class S.Y.B.S.C (CS) **Roll No.** 4334 **Year** 2021-2022

This is to certify that the work entered in this journal is the work of Mst. / Ms. SHAIKH KAYSAN RAZAUDDIN SHABANA who has worked for the year 2021-2022 in the Computer Laboratory.

Teacher In-Charge

Head of Department

Date :

Examiner

☆☆ INDEX ☆☆

☆☆ INDEX ☆☆

PRACTICAL NO : 01

Aim : Develop the presentation layer of Library Management software application with suitable menus.

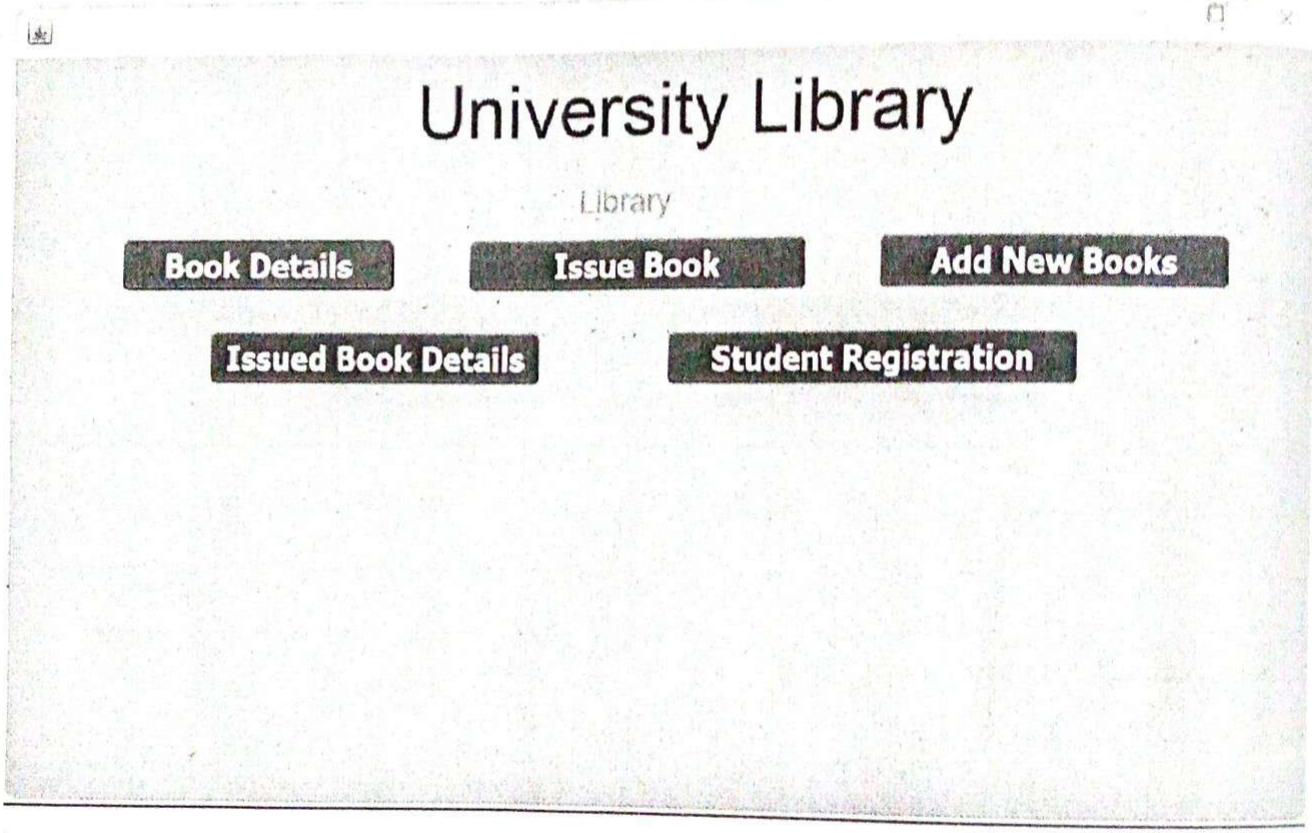
Step 1 : Create a java application using IDE --> Click on File --> New Project --> Java --> Java Application --> Give name of the project --> Click on Finish.

Step 2 : Create a JFrame form inside the source package --> Right click on package --> Select JFrame Form --> Give a name to your JFrame Form --> Then click on Finish.

Step 3 : Open JFrame For designing the interface.

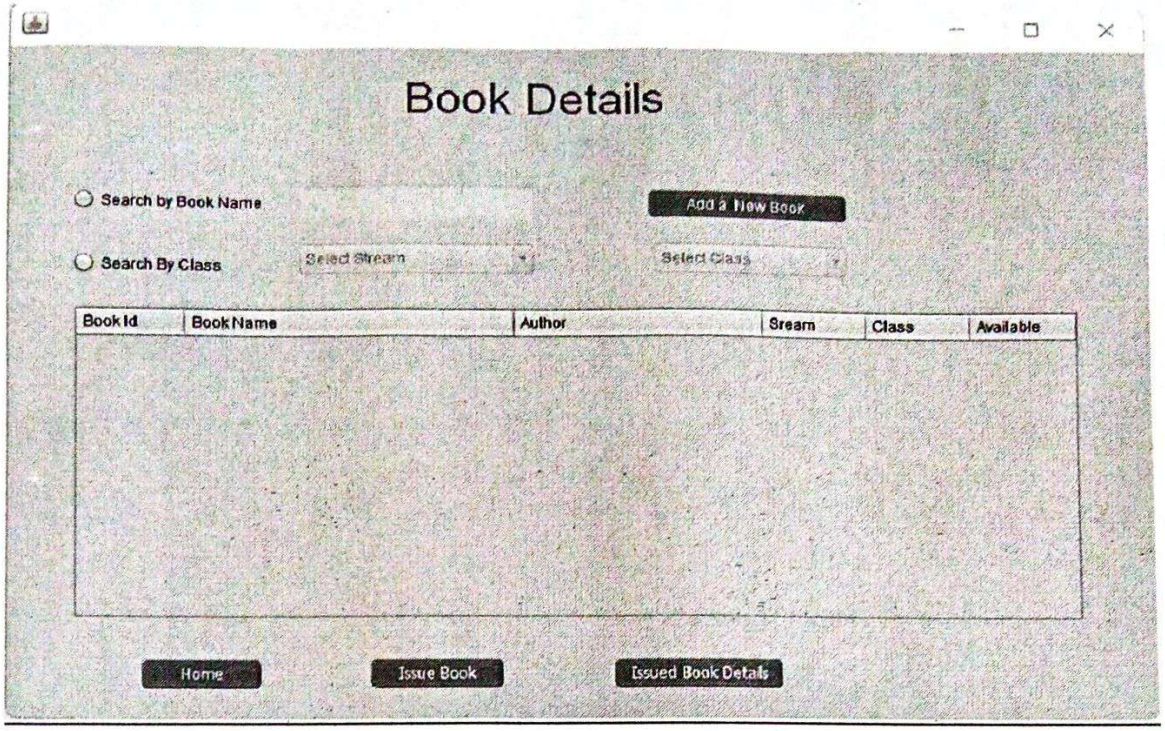
MainLayer.java

Drag and Drop the Swing Palette in the desi
part which are needed for this frame.



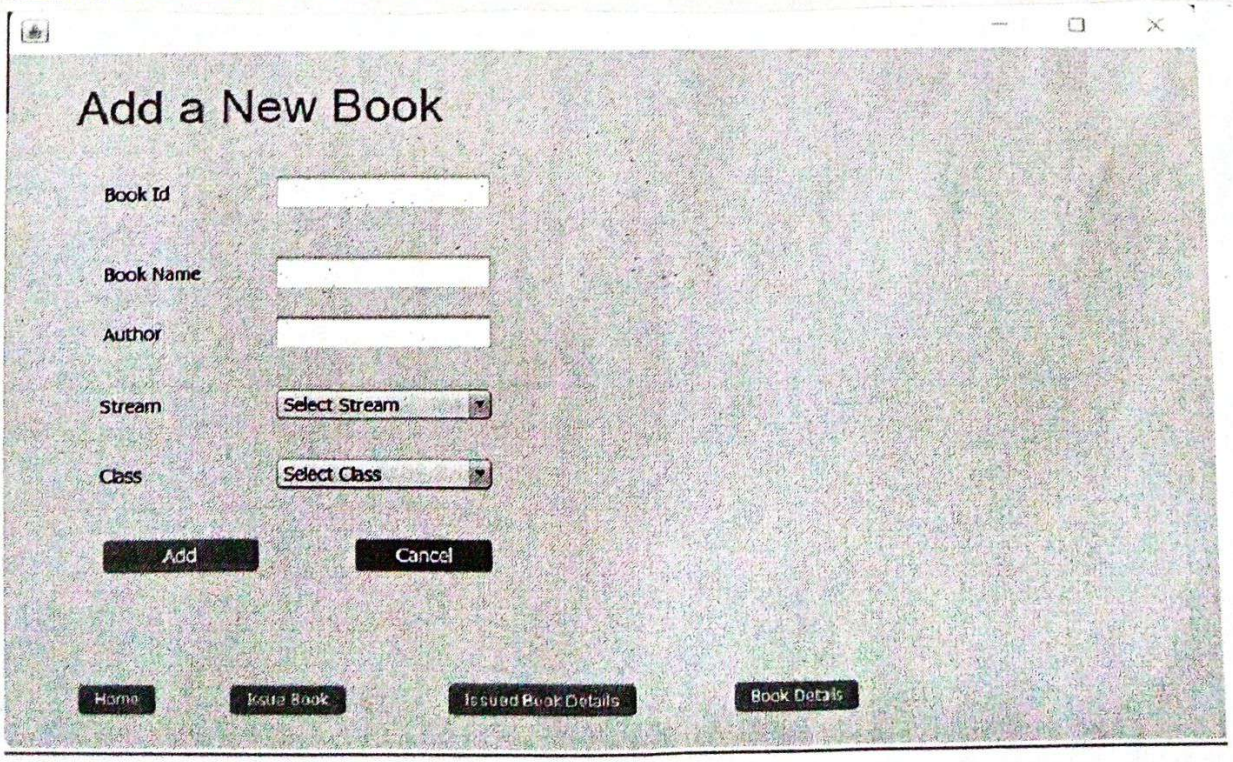
BookDetails.java

Drag and Drop the Swing Palette in the design
part which are needed for this frame.



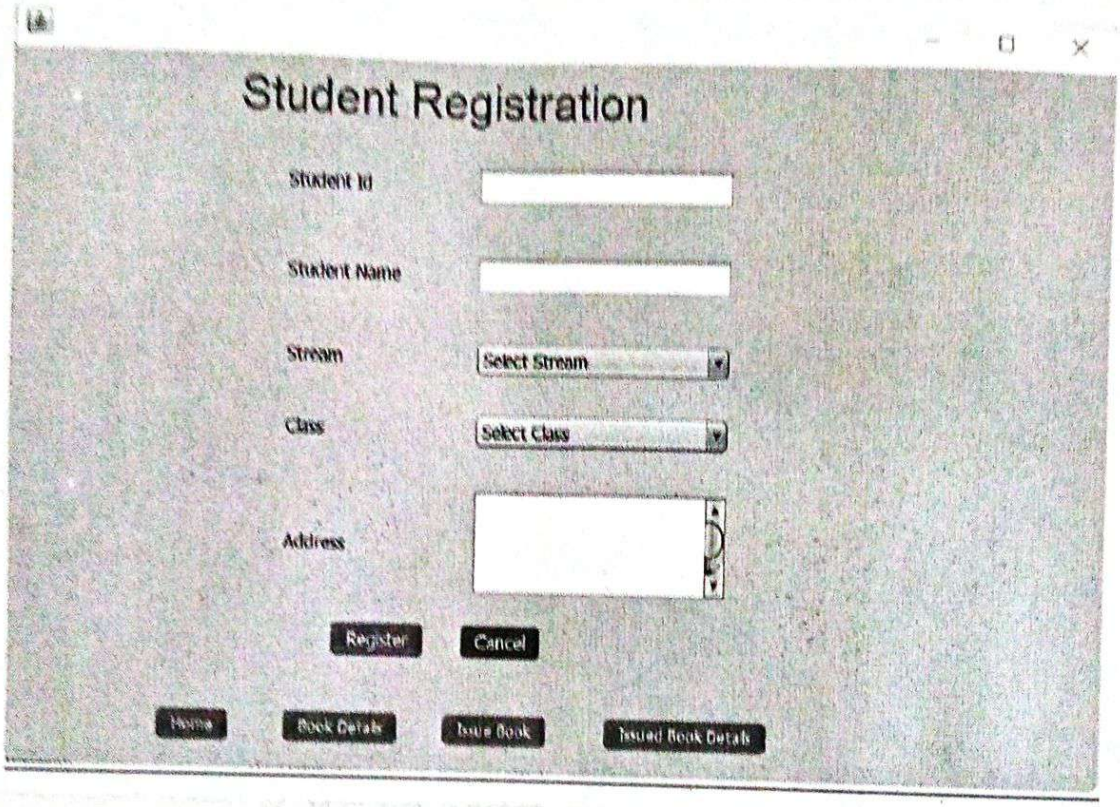
AddNew Book.java

Drag and Drop the Swing Palette in the design Part which are needed for this frame.



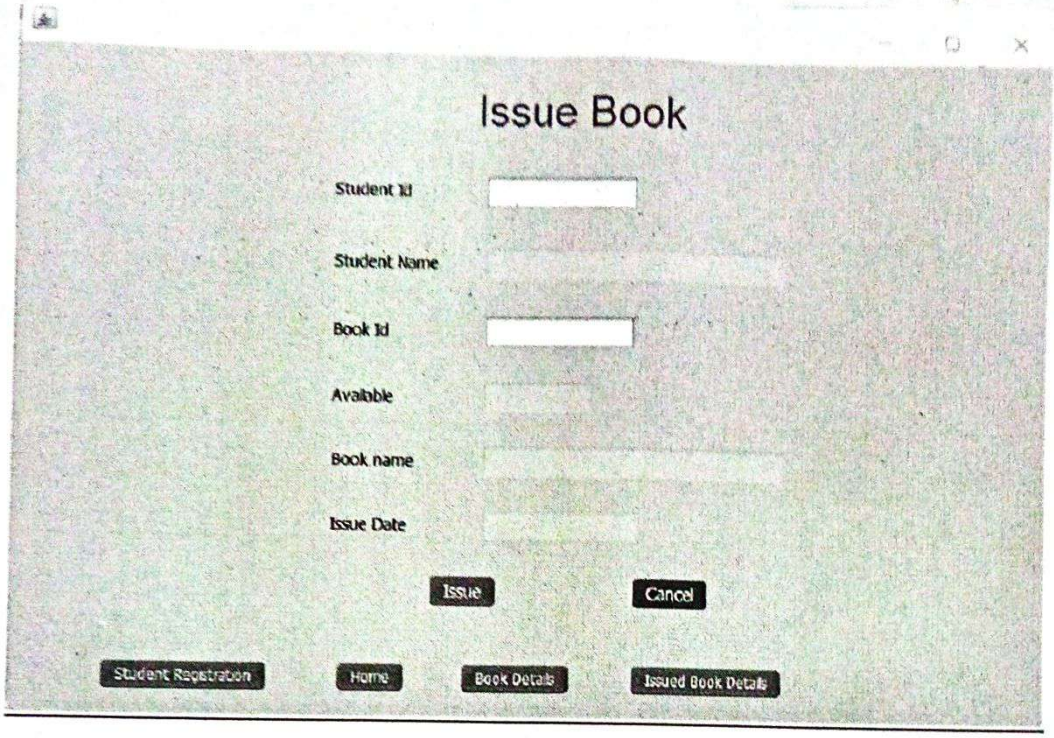
RegisterStudent.java

Drag and Drop the Swing Palette in the design part which we needed for this frame.



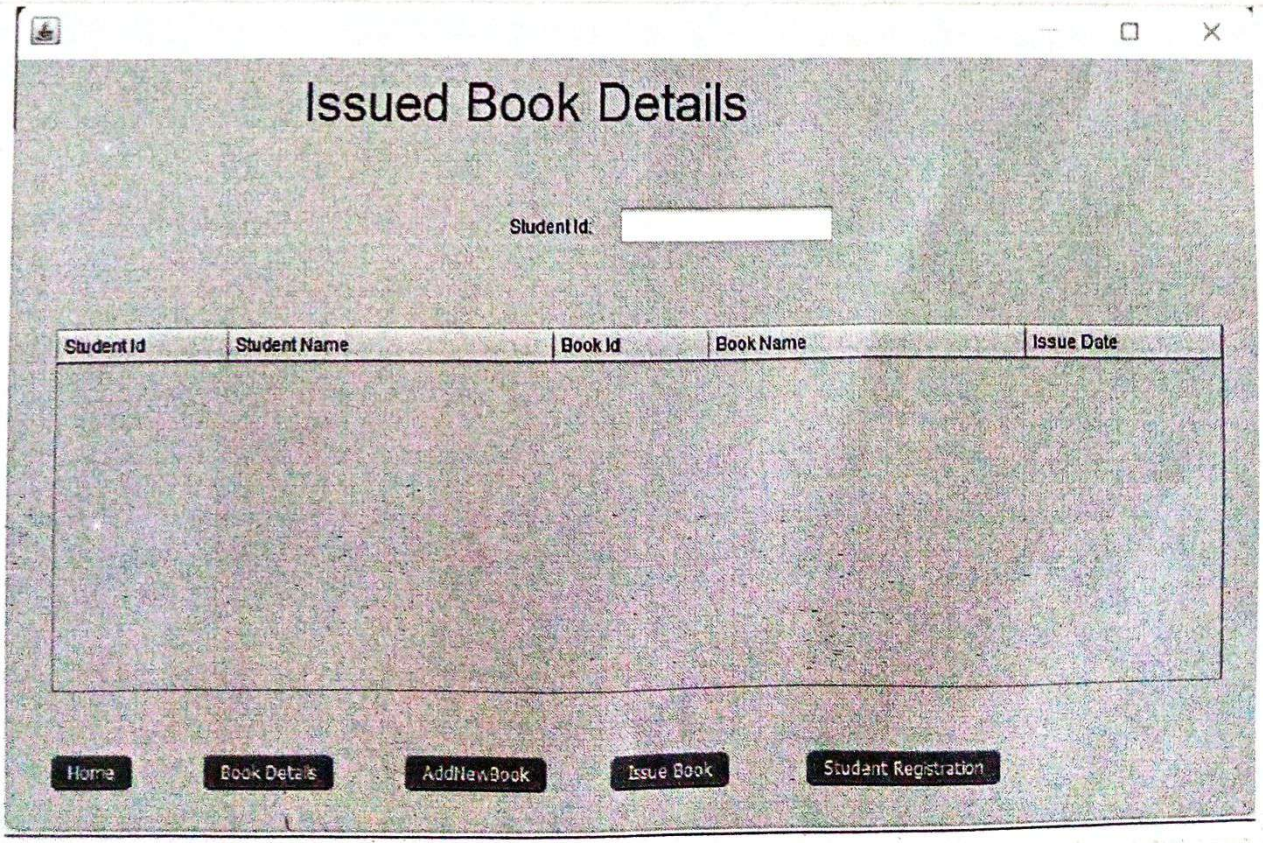
IssueBook.java

Drag and Drop the Swing Palette in the design part which are needed for this frame.



IssuedBook.java

Drag and Drop the Swing Palette in the design part which are needed for this frame.



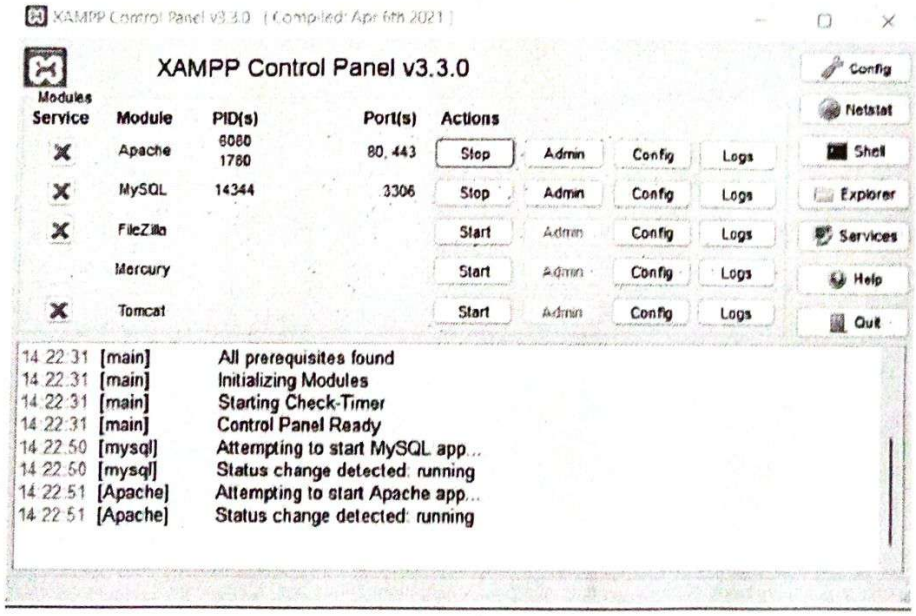
Conclusion : Sucessfully implemented presentation layer of Library management software using java.

PRACTICAL NO : 02

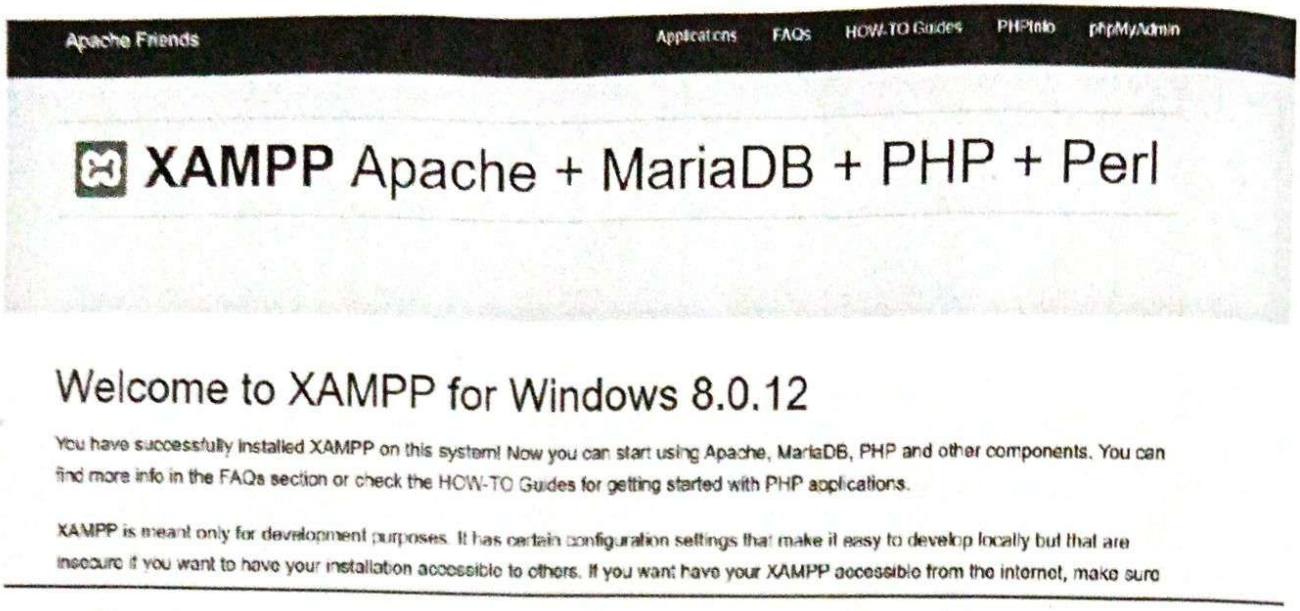
Aim : Design suitable data base for Library Management System.

Step 1 : Download and install xampp.

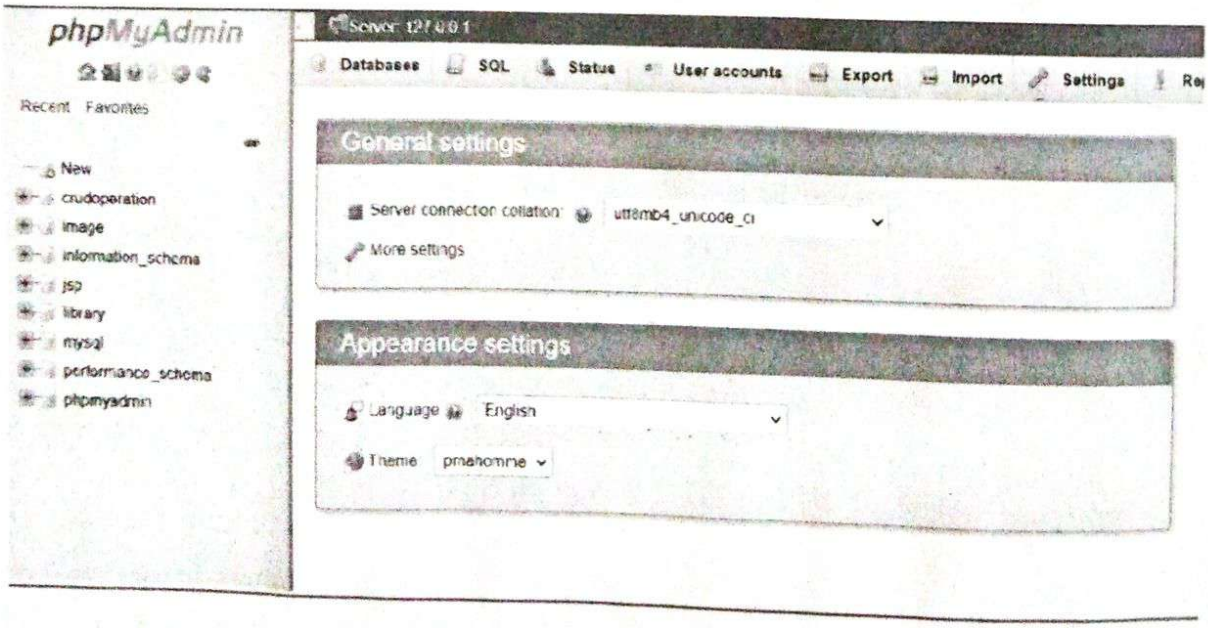
Step 2 : Start apache and mysql.



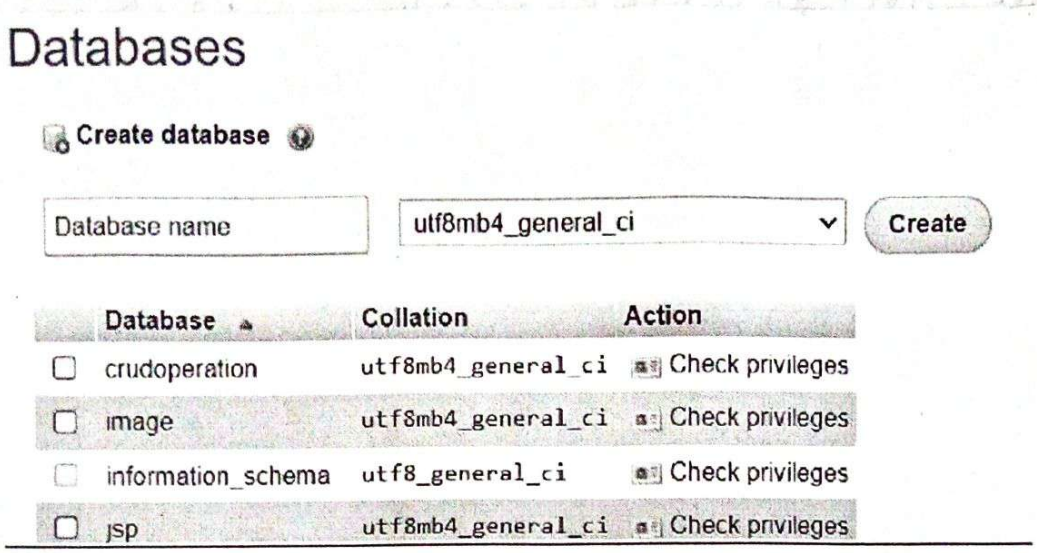
Step 3 : Visit to 127.0.0.1 for xampp page and click on phpMyadmin to create a database.



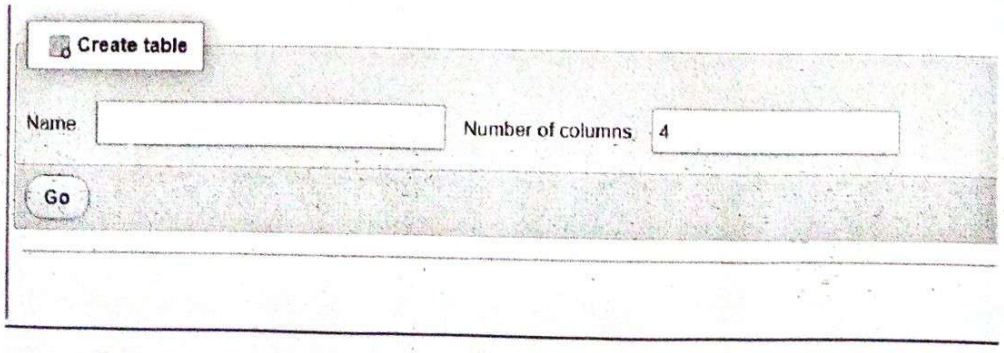
Step 4 : On left side of the screen all the database name are their click on new to create a new database.



Step 5 : Enter database name and click on create to create a new database.

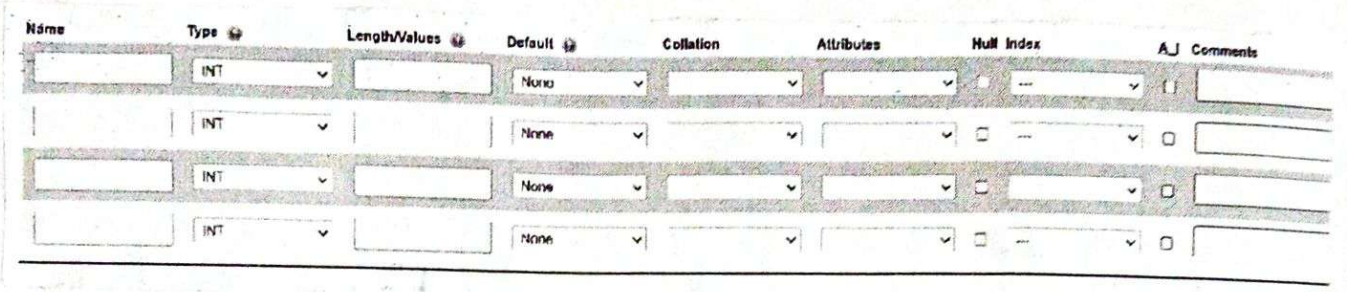


Step 6 : Then enter your table name and number of columns needed and click on GO.

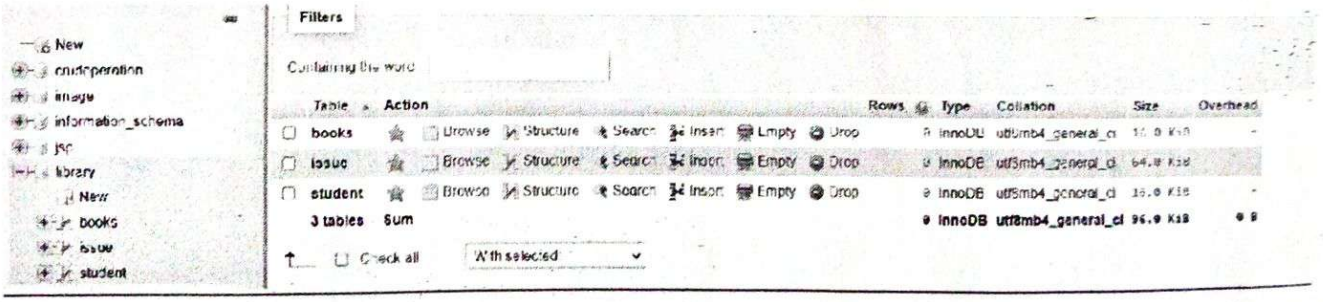


Step 7 : Then enter name of the column in name section --> add proper data type next to it --> then give appropriate length values --> If you want to make it

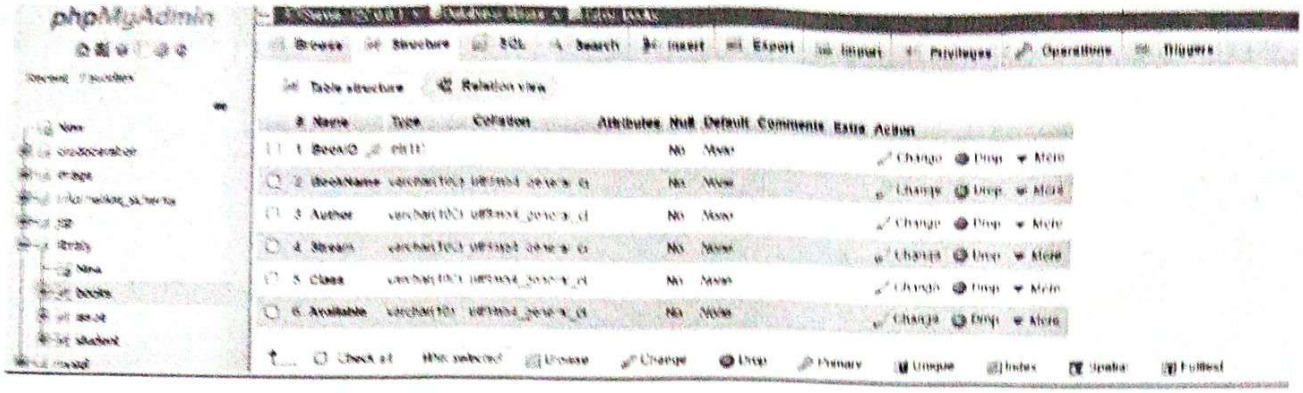
primary key click on null index their you will get that option --> If you want to make that column auto increment then check on that A_I option.
(Note: You can keep all the columns as it is except name, type and length values).

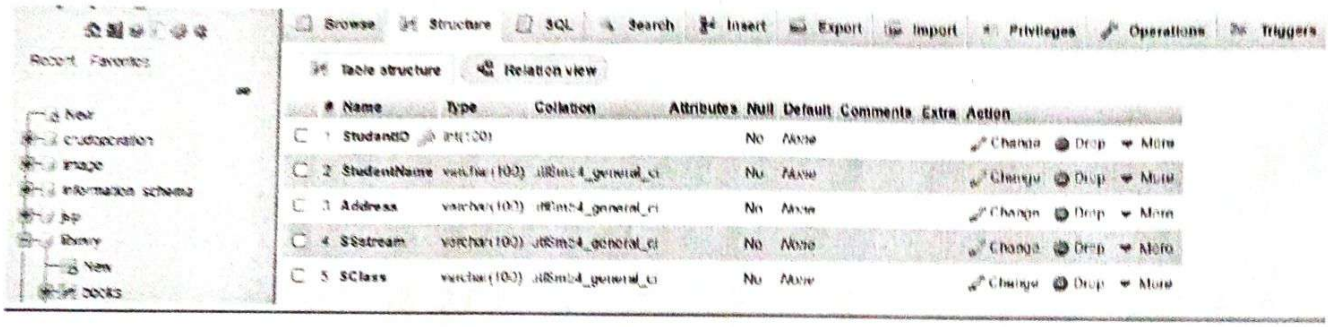


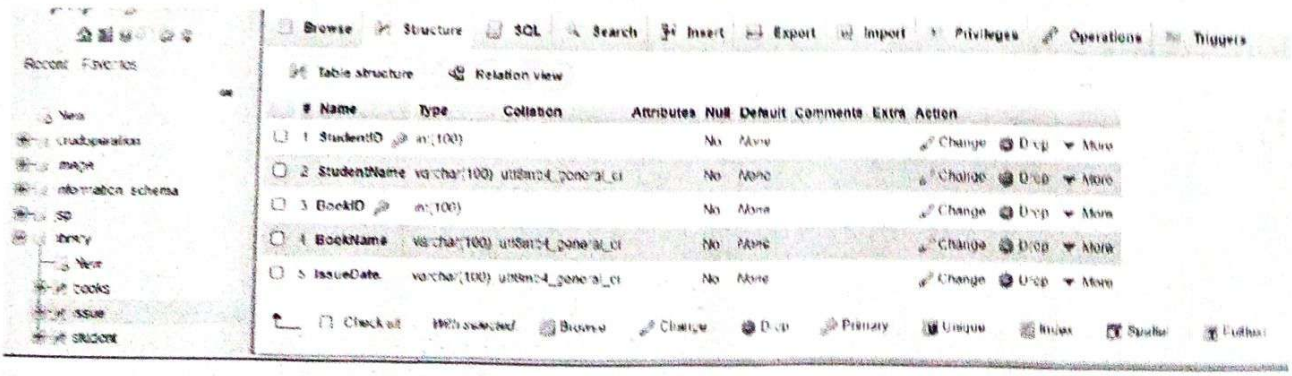
Step 8: Create appropriate database and table for library management software, For adding one then more table in database expand you database and click on new.



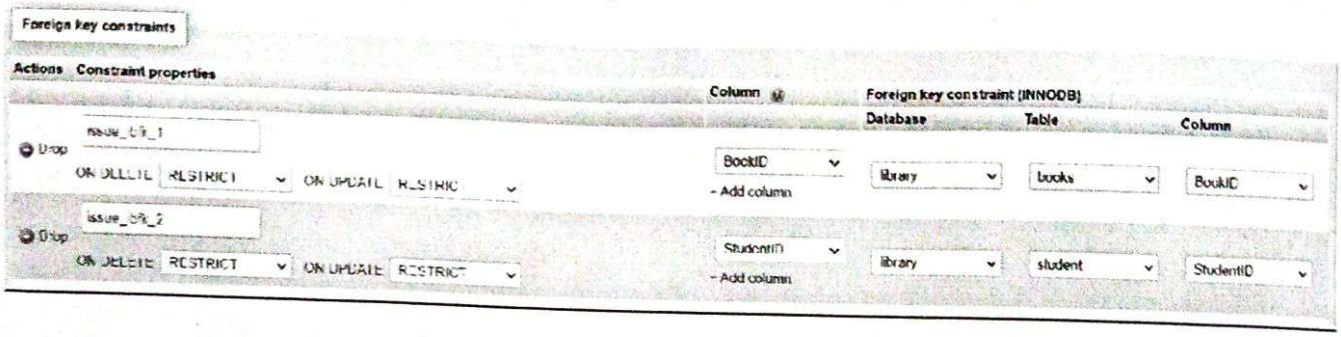
Step 9: As per presentation layer in practical 1 following database and tables are created.
In books table BookID is a primary key.
In student table StudentID is a primary key.
In issue table StudentID and BookID are foreign key


phpMyAdmin
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Table structure
Relation view
Name
Type
Collation
Attributes
Null
Default
Comments
Extra
Action
1 BookID
2 BookName
3 Author
4 Stream
5 Class
6 Available
Check all
With selected:
Browse
Change
Drop
Primary
Unique
Index
Spatial
Fulltext


Recent
Favorites
New
crudoperation
image
information_schema
jsp
library
New
books
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Table structure
Relation view
Name
Type
Collation
Attributes
Null
Default
Comments
Extra
Action
1 StudentID
2 StudentName
3 Address
4 SStream
5 SClass
No
None
Change
Drop
More


Recent
Favorites
New
crudoperation
image
information_schema
sp
library
New
books
issue
student
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Table structure
Relation view
Name
Type
Collation
Attributes
Null
Default
Comments
Extra
Action
1 StudentID
2 StudentName
3 BookID
4 BookName
5 IssueDate
No
None
Change
Drop
More
Check all
With selected:
Browse
Change
Drop
Primary
Unique
Index
Spatial
Fulltext

Step 10 : To add foreign Key click on relation view on the top the window then follow the below steps



Keep Constraint properties to default values --> In 1st column select column which you want to make foreign key --> In database select database where reference table is stored --> In table select the reference table name --> In column select the reference column.

Conclusion : Successfully created a database for Library management software.

PRACTICAL NO : 04

Aim : Develop Java application to store image in a database as well as retrieve image from database.

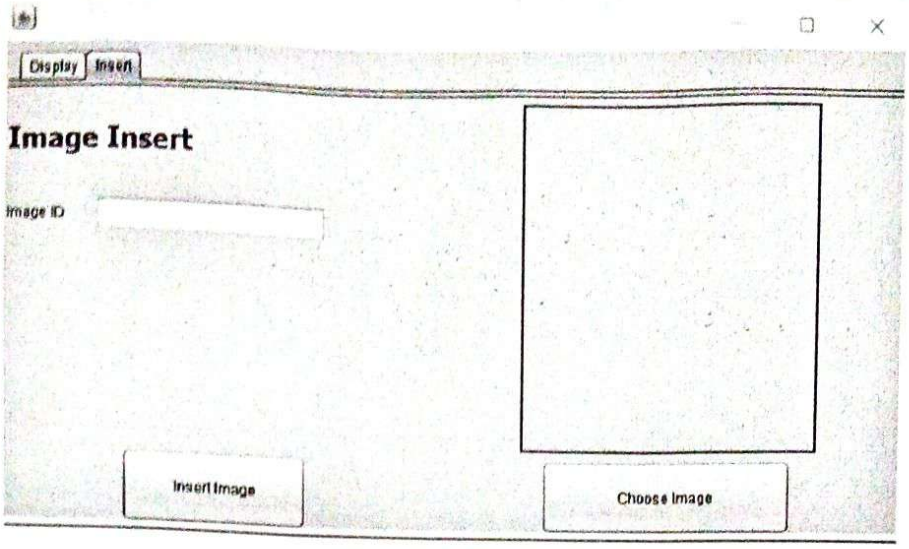
Step 1 : Create a java application --> Add one JFrame form and java class file.

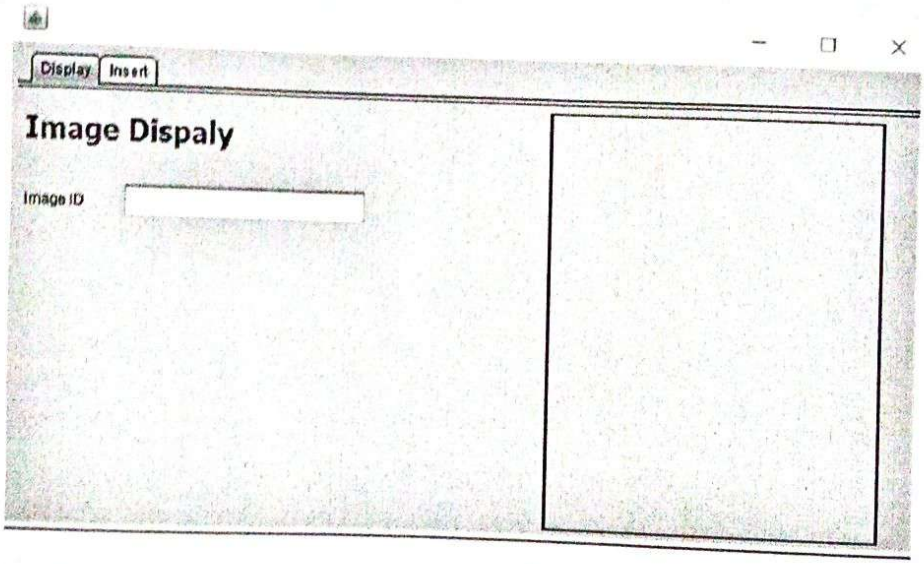
Step 2 : Start xampp --> Add database with one table of two columns --> Keep data type of one column as integer to store id and keep datatype as BLOB to store image in it.

Step 3 : Expand your project and add Mysql JDBC library in library folder.

Step 4 : Click on services in IDE --> Right click on database --> Click on New connection --> Select CMysql CConnector/Jdriver --> Then one window will appear in which you have to enter your database name and test the connection, If connection is established click on finish.

Step 5 : Make the design in JFrame form as shown in the below snapshots.





Step 6 : Now write the below code for this JFrame Form.

Main.java

```
package image;

import
java.io.ByteArrayOutputStream;
import java.io.File;
import
java.io.FileInputStream;
import
javax.swing.ImageIcon;
import
javax.swing.JFileChooser;
import java.awt.Image;
import java.sql.Connection;
import
java.sql.PreparedStatement;
import java.sql.ResultSet;


public class Main extends
javax.swing.JFrame {private ImageIcon
format=null;
  String
  fname=null;int
  s=0;
  byte[] pimage=null;
  Connection
  conn=null;

   public Main() {
     initComponents();
     conn=DBConnect.connect
     ();
   }

   @SuppressWarnings("unchecked")
   // <editor-fold defaultstate="collapsed" desc="Generated
   Code">private void initComponents() {

     jTabbedPane1 = new
     javax.swing.JTabbedPane();jPanel1 = new
     javax.swing.JPanel();
     jLabel3 = new
     javax.swing.JLabel(); jLabel4 =
     new javax.swing.JLabel(); txtdisid
```

650

```
= new javax.swing.JTextField();
lbldisplay = new
javax.swing.JLabel(); Insert = new
javax.swing.JPanel(); jLabel1 =
new javax.swing.JLabel(); jLabel2
= new javax.swing.JLabel(); txtid =
new javax.swing.JTextField();
lblimage = new
javax.swing.JLabel(); jButton1 =
new javax.swing.JButton();
jButton2 = new
javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
```

```
jLabel3.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
jLabel3.setText("Image Dispaly");

jLabel4.setText("Image ID");

txtdisid.addActionListener(new
  java.awt.event.ActionListener() {public void
  actionPerformed(java.awt.event.ActionEvent evt) {
    txtdisidActionPerformed(evt);
  }
});
txtdisid.addKeyListener(new
  java.awt.event.KeyAdapter() {public void
  keyReleased(java.awt.event.KeyEvent evt) {
    txtdisidKeyReleased(evt);
  }
});

lbldisplay.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0),
2));

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
  jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
  .addGroup(jPanel1Layout.createSequentialGroup()
    .addContainerGap()
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
    G)
      .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
javax.swing.GroupLayout.PREFERRED_SIZE)
      .addGroup(jPanel1Layout.createSequentialGroup()
        .addComponent(jLabel4)
        .addGap(32, 32, 32)
        .addComponent(txtdisid,
javax.swing.GroupLayout.PREFERRED_SIZE, 202,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
    43,Short.MAX_VALUE)
            .addComponent(lbldisplay, javax.swing.GroupLayout.PREFERRED_SIZE, 282,
    javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(33, 33, 33))
        );
        jPanel1Layout.setVerticalGroup(
          jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        Insert.setLayout(InsertLayout);
        InsertLayout.setHorizontalGroup(
          InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(InsertLayout.createSequentialGroup()
            .addGap(98, 98, 98)
              .addComponent(jLabel3,
    javax.swing.GroupLayout.PREFERRED_SIZE, 49,
    javax.swing.GroupLayout.PREFERRED_SIZE)
              .addGap(18, 18, 18)

    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
              .addComponent(jLabel4)
              .addComponent(txtdisid, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(0, 240, Short.MAX_VALUE)))
          .addContainerGap())
      );

        jTabbedPane1.addTab("Display", jPanel1);

      jLabel1.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
      jLabel1.setText("Image Insert");

      jLabel2.setText("Image ID");

      txtid.addActionListener(new
        java.awt.event.ActionListener() { public void
        actionPerformed(java.awt.event.ActionEvent evt) {
          txtidActionPerformed(evt);
        }
      });
```

```
lblimage.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0), 2));

jButton1.setText("Choose Image");
jButton1.addActionListener(new
java.awt.event.ActionListener() {
  public void actionPerformed(java.awt.event.ActionEvent
    evt) {jButton1ActionPerformed(evt);
  }
});

jButton2.setText("Insert Image");
jButton2.addActionListener(new
java.awt.event.ActionListener() {
  public void actionPerformed(java.awt.event.ActionEvent
    evt) {jButton2ActionPerformed(evt);
  }
});

      javax.swing.GroupLayout InsertLayout = new javax.swing.GroupLayout(Insert);

            .addComponent(jButton1, javax.swing.GroupLayout.DEFAULT_SIZE, 59,
          Short.MAX_VALUE)
          .addComponent(jButton2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
    );

    jTabbedPane1.addTab("Insert", Insert);

        .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 156,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 214,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(90, 90, 90))
      .addGroup(InsertLayout.createSequentialGroup()
        .addGap(0, 0, Short.MAX_VALUE)
        .addGroup(InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
        )
```

```
            .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 210,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGroup(InsertLayout.createSequentialGroup()
              .addComponent(jLabel2)
              .addGap(26, 26, 26)
              .addComponent(txtid,
javax.swing.GroupLayout.PREFERRED_SIZE, 194,
javax.swing.GroupLayout.PREFERRED_SIZE)))
          .addGap(168, 168, 168)
          .addComponent(lblimage, javax.swing.GroupLayout.PREFERRED_SIZE, 253,
javax.swing.GroupLayout.PREFERRED_SIZE)
          .addGap(69, 69, 69))
      );
      InsertLayout.setVerticalGroup(
        InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(InsertLayout.createSequentialGroup()
          .addGroup(InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
          )
            .addGroup(InsertLayout.createSequentialGroup()
              .addContainerGap()
              .addComponent(lblimage,
javax.swing.GroupLayout.PREFERRED_SIZE, 276,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(InsertLayout.createSequentialGroup()
              .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 82,
javax.swing.GroupLayout.PREFERRED_SIZE)
              .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel2)
                .addComponent(txtid, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))))
          .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
          .addGroup(InsertLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
          )


      javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
          .addContainerGap()
          .addComponent(jTabbedPane1)
          .addContainerGap())
      );
      layout.setVerticalGroup(
```

```
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
          .addContainerGap()
          .addComponent(jTabbedPane1)
          .addContainerGap())
      );

      pack();
    }// </editor-fold>

    private void txtidActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
      // using this code you can brows image and image set to
      lableJFileChooser fchoser=new JFileChooser();
      fchoser.showOpenDialog(null);
      File
      f=fchoser.getSelectedFile();
      fname=f.getAbsolutePath();
      ImageIcon micon=new
      ImageIcon(fname);try {
        File image=new File(fname);
        FileInputStream fis=new FileInputStream(image);
        ByteArrayOutputStream baos=new
        ByteArrayOutputStream();byte[] buf=new byte[1024];
        for(int readnum; (readnum=fis.read(buf)) !=-1;)
        {
          baos.write(buf,0,readnum);
        }
        pimage=baos.toByteArray();
        lblimage.setIcon(resizeImage(fname,
        buf));
      } catch (Exception e) {
      }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
      // This code use to insert image to database


      }
      else
      {

      }
```

```
        } catch (Exception e) {

        }
    }

    public ImageIcon resizeImage(String imagePath, byte[] pic){
        // This code use to resize image to fit
        lableImageIcon myImage=null;
        if(imagePath !=null)
        {
        myImage=new ImageIcon(imagePath);

        }else{

        myImage=new ImageIcon(pic);
        }

        Image img=myImage.getImage();
        Image img2=img.getScaledInstance(lblimage.getHeight(),
lblimage.getWidth(),Image.SCALE_SMOOTH);
        ImageIcon image=new
        ImageIcon(img2);return image;
    }

    /**
     * @param args the command line arguments
     */
public static void main(String args[])
                                        {
    /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try
        {
            for (javax.swing.UIManager.LookAndFeelInfo
info :
javax.swing.UIManager.getInstalledLookAndFeels()
) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassNa
                    me()); break;
                }
            }
```

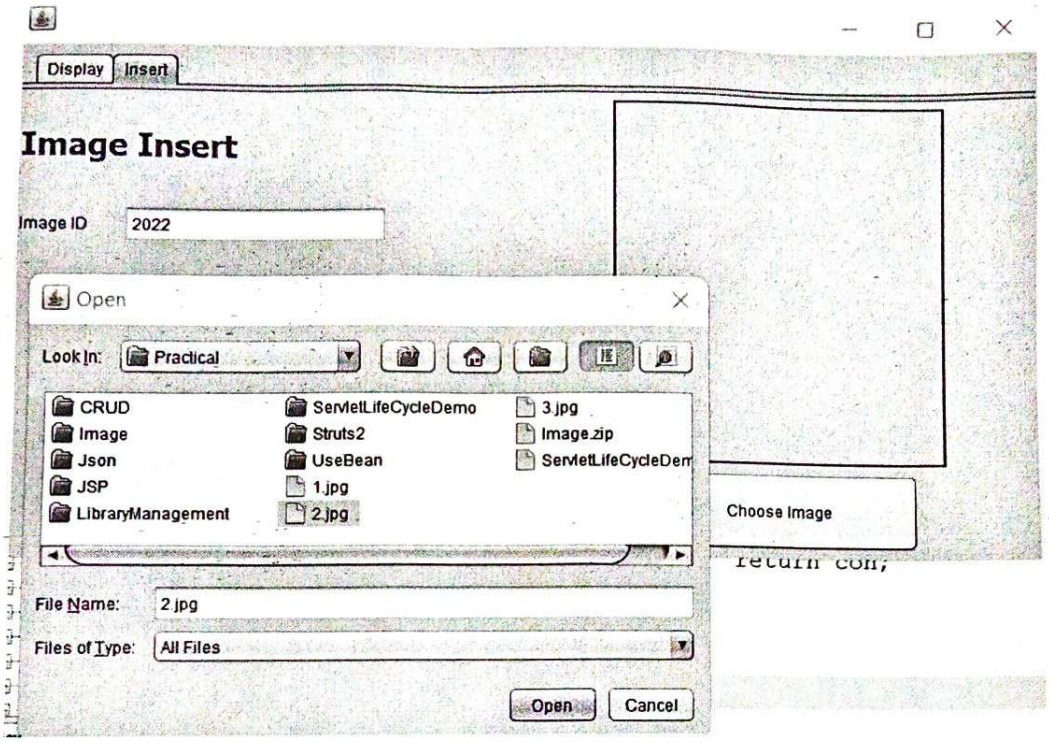
```
        } catch (ClassNotFoundException ex) {
java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.Level.SEVERE, null,

      ex);
      } catch (InstantiationException ex)
        java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.Level.SEVERE,
        null,

 ex);
   } catch (IllegalAccessException ex) {
   java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.Level.SEVERE, null,

ex);
 } catch (javax.swing.UnsupportedLookAndFeelException ex) {
    java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.Level.SEVERE, null,

  ex);

 }
       //</editor-fold>

       /* Create and display the form */
       java.awt.EventQueue.invokeLater(new
       Runnable() {
          public void run() {
             new Main().setVisible(true);
          }
       });
           }
      // Variables declaration - do not
      modifyprivate javax.swing.JPanel
      Insert; private javax.swing.JButton
      jButton1; private
      javax.swing.JButton jButton2;
      private javax.swing.JLabel jLabel1;
      private javax.swing.JLabel jLabel2;
      private javax.swing.JLabel jLabel3;
      private javax.swing.JLabel jLabel4;
      private javax.swing.JPanel jPanel1;
      private javax.swing.JTabbedPane
      jTabbedPane1;private javax.swing.JLabel
      lbldisplay;
      private javax.swing.JLabel lblimage;
      private javax.swing.JTextField
      txtdisid;private
```

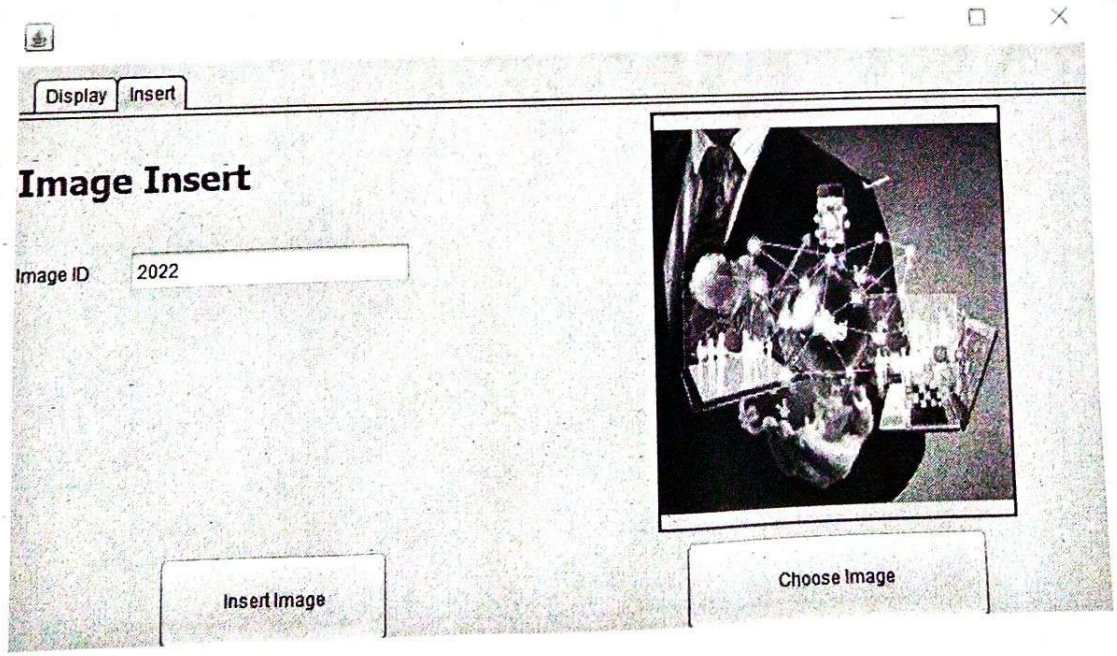
```
    javax.swing.JTextField txtid;
    // End of variables declaration
}
```

DBConnect.java

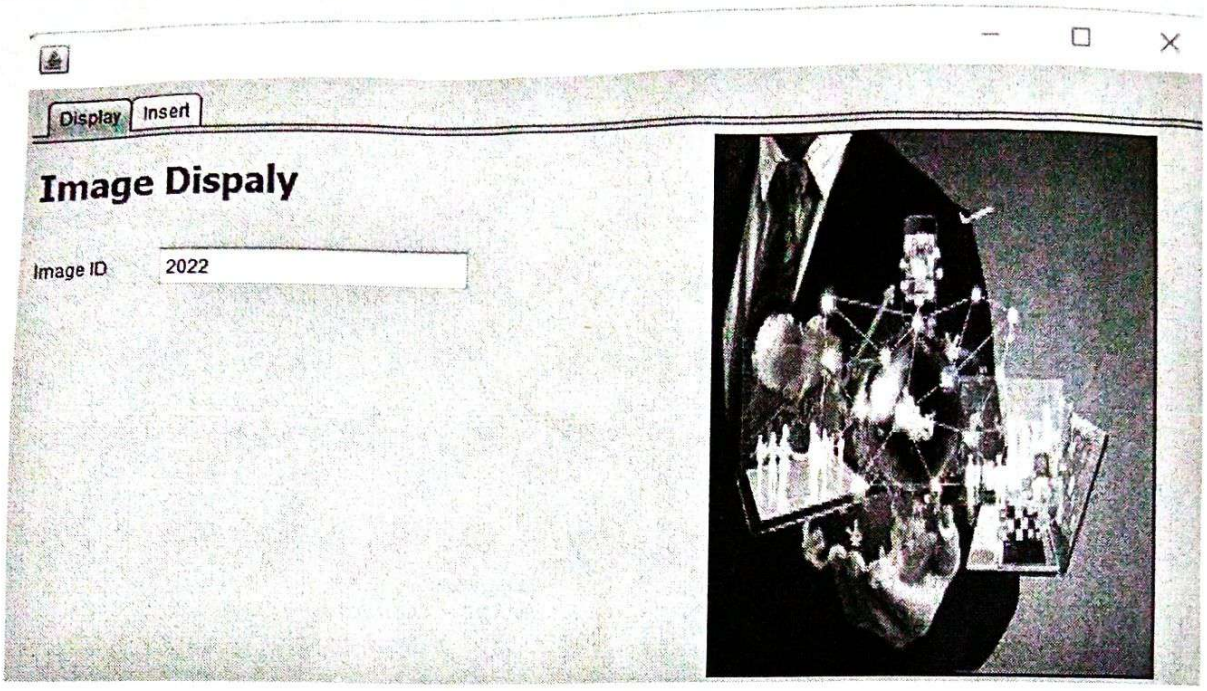
```
package image;

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

public class DBConnect {
    public static Connection connect()
    {
     Connection
       con=null;try {
          Class.forName("com.mysql.jdbc.Driver");
          con=DriverManager.getConnection("jdbc:mysql://localhost:3306/image?","root","");
 }
 catch (Exception e)
    {
       System.out.println("inter.DBConnect.connect()");
    }
   return con;
 }
 }
```

Output:

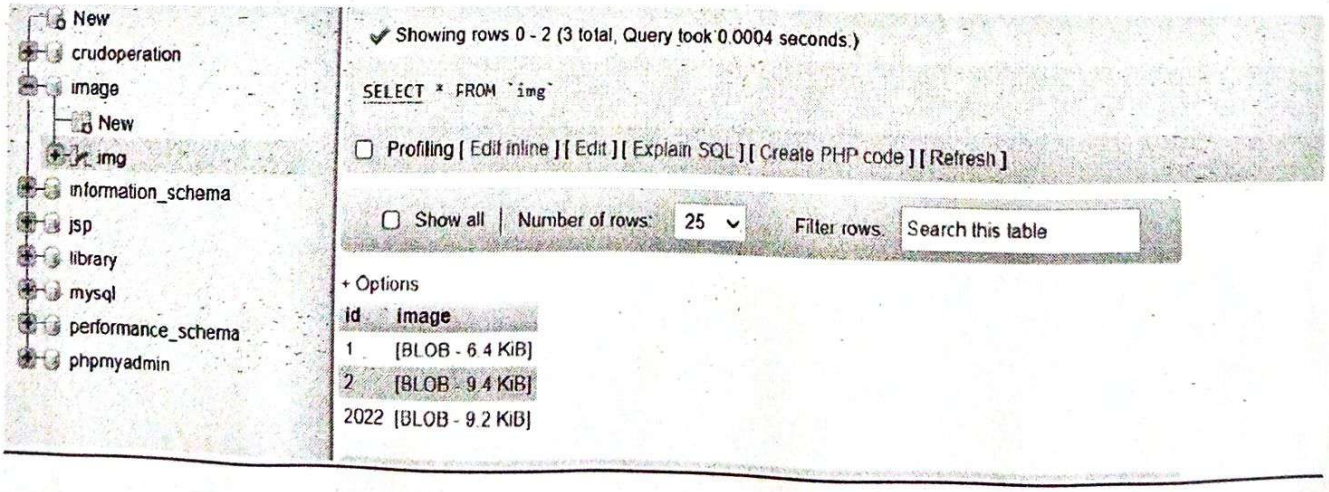




After clicking on insert button this image will be added in database with id 2022.



As you can after entering the same id on display image side it is displaying the same image which was stored in database.



Conclusion: Successfully stored and retrived image from database using java.

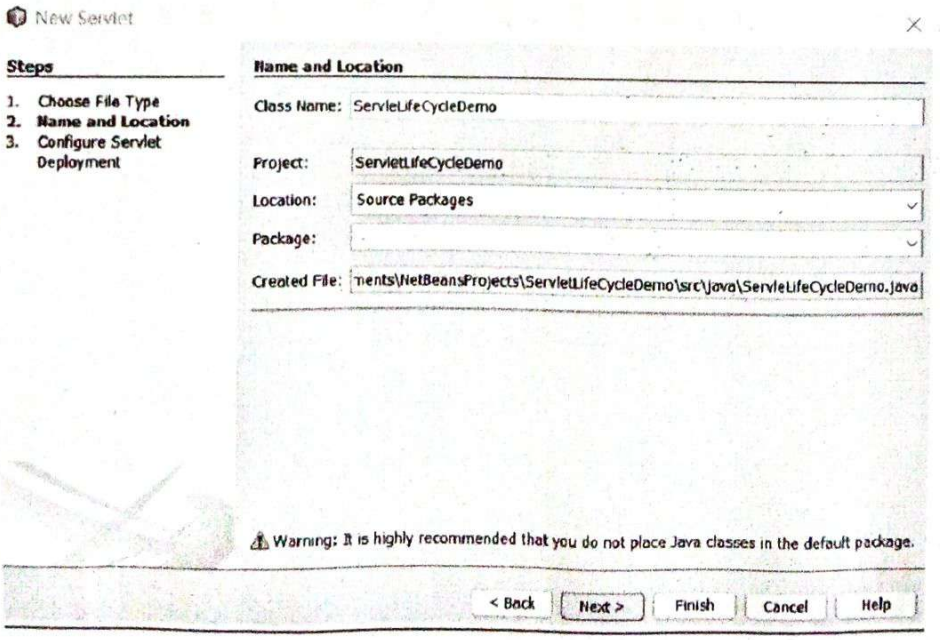
880

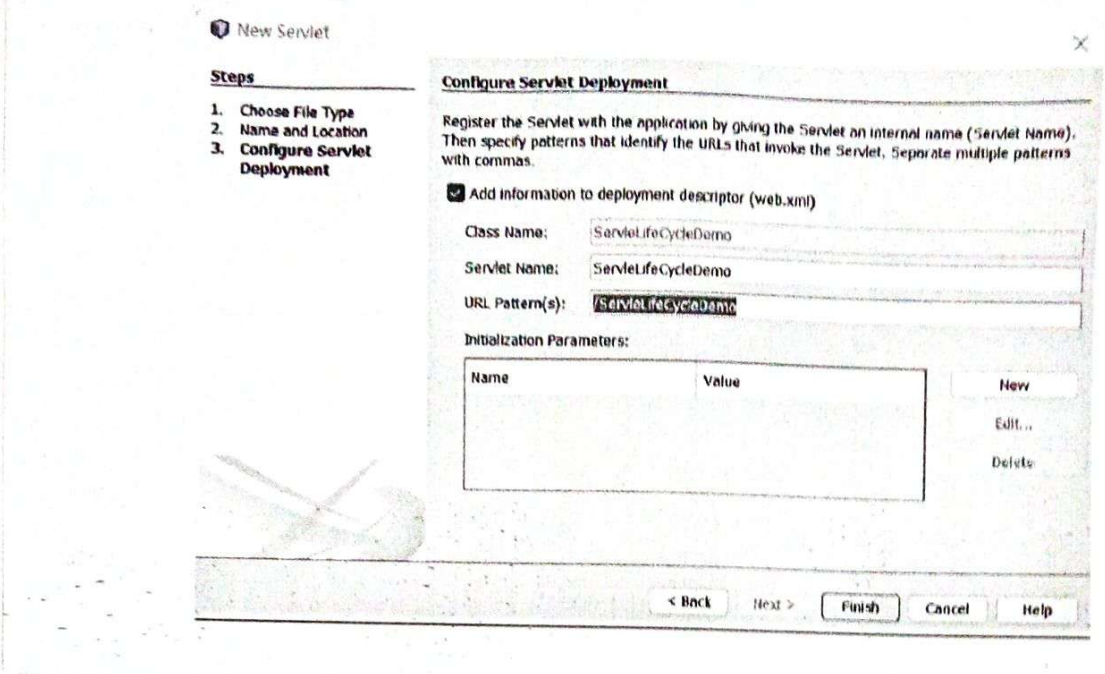
PRACTICAL NO : 05

Aim : Write a Java Application to demonstrate servlet life cycle.

Step 1 : Create java a web application --> Click on File --> New Project --> Java Web --> Web Application --> Give name of the project --> Select Server --> Click on finish

Step 2 : Create a Servlet file inside the source package --> Right click on package --> Select Servlet --> Give a name to your servlet file --> Check on the deployment descriptor (web.xml) --> Note down the url pattern written over their --> Then Click on finish.





Step 3: Write the servlet life cycle code inside servlet file you have created.
(Note: Their will be some auto-generated code in servlet file keep only that blocks of code which is needed).

ServletLifeCycleDemo.java

```
import
java.io.IOException;
import
java.io.PrintWriter;
import
javax.servlet.ServletException;
import
javax.servlet.ServletRequest;
import
javax.servlet.ServletResponse;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;


public class ServletLifeCycleDemo extends HttpServlet
{

  public void init() throws ServletException
  {
    System.out.println("Servlet has been Initialized...");
  }


  public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException
  {
    PrintWriter out = res.getWriter();
    out.println("Servlet Life Cycle Has three stages \n. Initialization \n. Service \n.
    Destroy");System.out.println("Servlet started servicing...");
  }


  public void destroy()
  {
    System.out.println("Servlet has been Destroyed...");
  }
}
```

Step 4 : Write the html page code inside index.html File present inside Web Pages Folder.

Index.html

```
<html>
<head>
    <title>Servlet Life Cycle</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form action="ServletLifeCycleDemo" method="get">
      <input type="submit" value="Click Here">
    </form>
  </body>
</html>
```

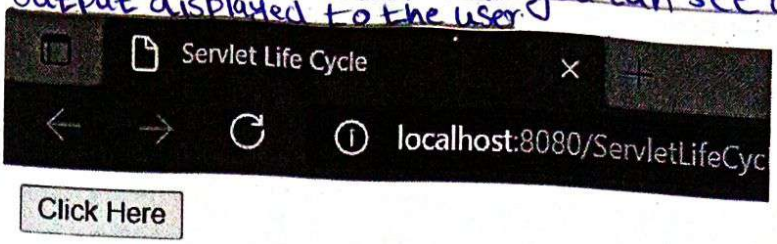
Step5 : Make sure you keep the url name same inside html file as it is written inside web.xml file for that url, web.xml file is present inside configuration file.

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web- app_3_1.xsd">
  <servlet>
    <servlet-name>ServletLifeCycleDemo</servlet-name>
    <servlet-class>ServletLifeCycleDemo</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>ServletLifeCycleDemo</servlet-name>
    <url-pattern>/ServletLifeCycleDemo</url-pattern>
  </servlet-mapping>
</web-app>
```

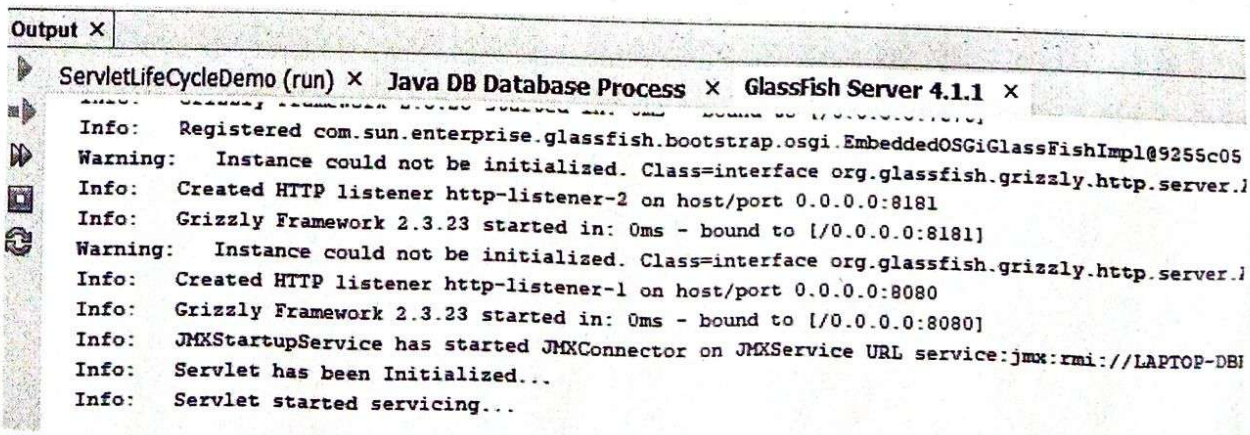
Output:

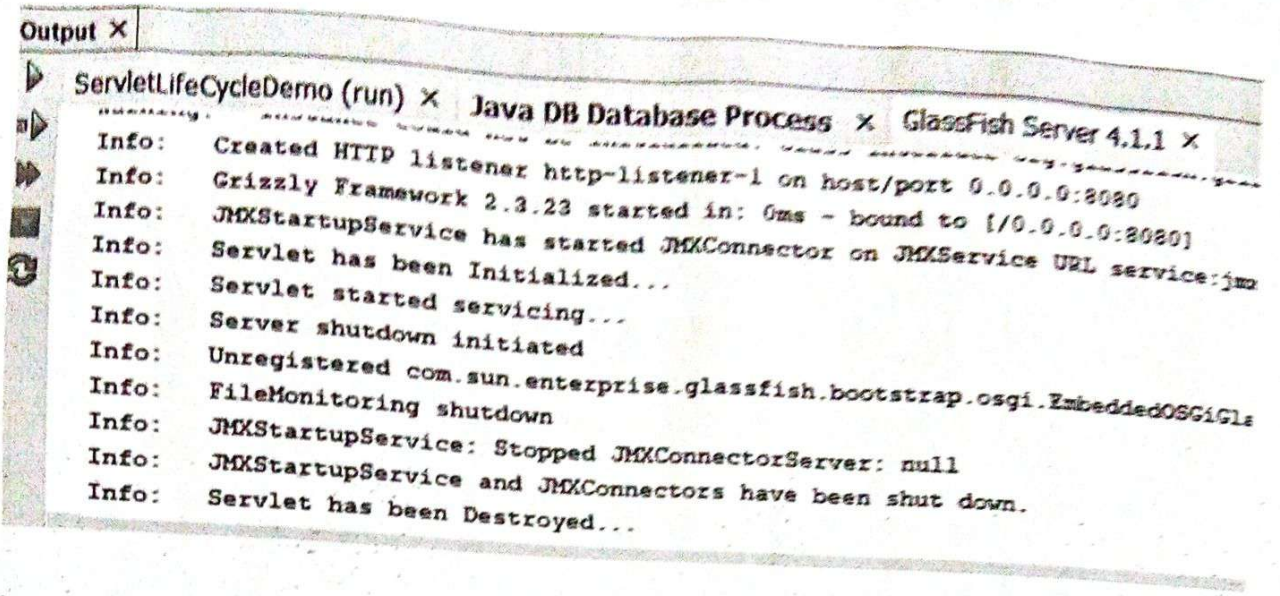
After clicking on button you can see appropriate output displayed to the user.



Servlet Life Cycle Has three stages
. Initialization
. Service
. Destroy

As you can see below simultaneously when servlet is running its init, service and destroy methods are also giving us the output in terminal.




Output
ServletLifeCycleDemo (run)
Java DB Database Process
GlassFish Server 4.1.1
Info: Created HTTP listener http-listener-1 on host/port 0.0.0.0:8080
Info: Grizzly Framework 2.3.23 started in: 0ms - bound to [/0.0.0.0:8080]
Info: JMXStartupService has started JMXConnector on JMXService URL service:jmx
Info: Servlet has been Initialized...
Info: Servlet started servicing...
Info: Server shutdown initiated
Info: Unregistered com.sun.enterprise.glassfish.bootstrap.osgi.EmbeddedOSGiGla
Info: FileMonitoring shutdown
Info: JMXStartupService: Stopped JMXConnectorServer: null
Info: JMXStartupService and JMXConnectors have been shut down.
Info: Servlet has been Destroyed...

Conclusion: Successfully implemented Servlet Life Cycle using java.

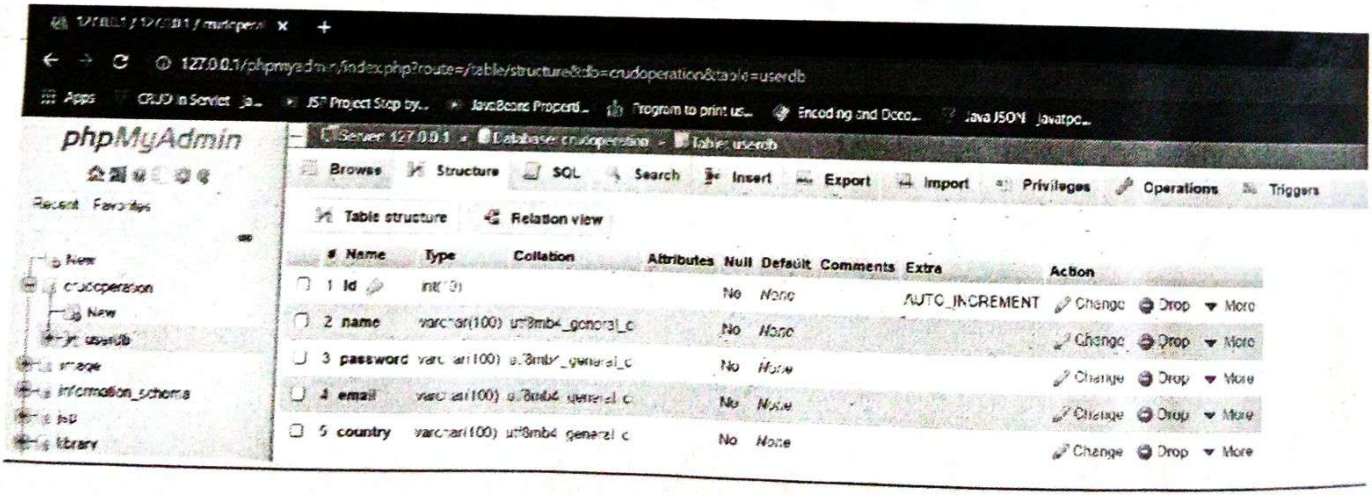
PRACTICAL NO : 06

Aim : Design database for employee administration. Develop servlet using to perform CRUD operations.

Step 1 : Create a java web application --> Create different servlet files and 2 java class file.

Step 2 : Expand your project and add Mysql JDBC library in libraries folder.

Step 3 : Create database on xampp.



Step 4 : Click on services in IDE --> Right click on database --> Click on New connection --> Select Mysql(Connector/Jdriver) --> Then one window will appear in which you have to enter your database name and test the connection. If connection is established Click on finish.

Step 5 : In servlet file write the codes for CRUD operation

Step 6 : Inside one class file write database connection and CRUD statements and in other class file write getters and setters code variables.

Emp.java

```
public class Emp
{
private int id;
private String name,password,email,country;

  public int
     getId() {
     return id;
  }

  public void setId(int
     id) {this.id = id;
```

```
    }

    public String
      getName() {return
      name;
    }

    public void setName(String
      name) {this.name = name;
    }

    public String
      getPassword() {return
      password;
    }

    public void setPassword(String
      password) {this.password =
      password;
    }

    public String
      getEmail() {return
      email;
    }

    public void setEmail(String
      email) {this.email = email;
    }

    public String
      getCountry() {return
      country;
    }

    public void setCountry(String
      country) {this.country = country;
    }
}
```

EmpDao.java

```
import
java.util.*;
import
java.sql.*;
```

```
public class EmpDao
{
  public static Connection
    getConnection(){Connection
    con=null;
    try{
      Class.forName("com.mysql.jdbc.Driver");
      con=DriverManager.getConnection("jdbc:mysql://localhost:3306/crudoperation?","root","");
    }

    catch(Exception e)
    {System.out.println(e);}
    return con;
  }
  public static int save(Emp
    e){int status=0;
    try{
      Connection con=EmpDao.getConnection();
      PreparedStatement ps=con.prepareStatement("insert into
userdb(name,password,email,country)values (?,?,?,?)");
      ps.setString(1,e.getName());
      ps.setString(2,e.getPassword());
      ps.setString(3,e.getEmail());
      ps.setString(4,e.getCountry());

      status=ps.executeUpdate();

      con.close();
    }catch(Exception ex){ex.printStackTrace();}

    return status;
  }
  public static int update(Emp
    e){int status=0;
    try{
      Connection con=EmpDao.getConnection();
      PreparedStatement ps=con.prepareStatement("update userdb
setname=?,password=?,email=?,country=? where id=?");
      ps.setString(1,e.getName());
      ps.setString(2,e.getPassword());
      ps.setString(3,e.getEmail());
      ps.setString(4,e.getCountry());
      ps.setInt(5,e.getId());

      status=ps.executeUpdate();
```

```
        con.close();
      }catch(Exception ex){ex.printStackTrace();}
      return status;
    }
    public static int delete(int
      id){int status=0;
      try{
        Connection con=EmpDao.getConnection();
        PreparedStatement ps=con.prepareStatement("delete from userdb where
        id=?");ps.setInt(1,id);
status=ps.executeUpdate();

        con.close();
      }catch(Exception e){e.printStackTrace();}

      return status;
    }
    public static Emp getEmployeeById(int
      id){Emp e=new Emp();
      try{
        Connection con=EmpDao.getConnection();
        PreparedStatement ps=con.prepareStatement("select * from userdb where
        id=?");ps.setInt(1,id);
        ResultSet
        rs=ps.executeQuery();
        if(rs.next()){
          e.setId(rs.getInt(1));
          e.setName(rs.getString(2));
          e.setPassword(rs.getString(3));
          e.setEmail(rs.getString(4));
          e.setCountry(rs.getString(5));
        }
        con.close();
      }catch(Exception ex){ex.printStackTrace();}

      return e;
    }
    public static List<Emp> getAllEmployees(){
      List<Emp> list=new ArrayList<Emp>();

      try{
        Connection con=EmpDao.getConnection();
        PreparedStatement ps=con.prepareStatement("select * from
        userdb");ResultSet rs=ps.executeQuery();
        while(rs.next()){
```

```
            Emp e=new Emp();
            e.setId(rs.getInt(1));
            e.setName(rs.getString(2));
            e.setPassword(rs.getString(3));
            e.setEmail(rs.getString(4));
            e.setCountry(rs.getString(5));
            list.add(e);
          }
          con.close();
        }catch(Exception
        e){e.printStackTrace();}return list;
}
```

ViewServlet.java

```
import
java.io.IOException;
import
java.io.PrintWriter;
import java.util.List;
import javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
@WebServlet("/ViewServlet")
public class ViewServlet extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse
response)throwsServletException, IOException
  {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();
    out.println("<a href='index.html'>Add New
    Employee</a>");out.println("<h1>Employees
    List</h1>");

    List<Emp>

    list=EmpDao.getAllEmployees();

    out.print("<table border='1'

    width='100%'");

out.print("<tr><th>Id</th><th>Name</th><th>Password</th><th>Email</th><th>Country</th><th>
Edit
</th><th>Delete</th></tr>");
     for(Emp e:list){

out.print("<tr><td>"+e.getId()+"</td><td>"+e.getName()+"</td><td>"+e.getPassword()+"</td>
<td>"+e. getEmail()+"</td><td>"+e.getCountry()+"</td><td><a
href='EditServlet?id="+e.getId()+"'>edit</a></td><td><a
href='DeleteServlet?id="+e.getId()+"'>delete</a></td></tr>");
     }
     out.print("</table>"
     );out.close();
```

```
    }
  }
```

SaveServlet.java

```
import
java.io.IOException;
import
java.io.PrintWriter;
import javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
@WebServlet("/SaveServlet")

public class SaveServlet extends HttpServlet {
  protected void doPost(HttpServletRequest request, HttpServletResponse
response)throwsServletException, IOException
  {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();

    String name=request.getParameter("name");
    String
    password=request.getParameter("password");
    String email=request.getParameter("email");
    String country=request.getParameter("country");

   Emp e=new Emp();
   e.setName(name);
   e.setPassword(password);
   e.setEmail(email);
   e.setCountry(country);

   int
   status=EmpDao.save(e);
   if(status>0){
     out.print("<p>Record saved successfully!</p>");
     request.getRequestDispatcher("index.html").include(request, response);
   }else{
     out.println("Sorry! unable to save record");
   }
```

```
        out.close();
    }
}
```

EditServlet.java

```
import
java.io.IOException;
import
java.io.PrintWriter;
import javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;

import
javax.servlet.http.HttpServletResponse;
@WebServlet("/EditServlet")
public class EditServlet extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse
response)throwsServletException, IOException
  {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();
    out.println("<h1>Update
    Employee</h1>");

      String sid=request.getParameter("id");
      int id=Integer.parseInt(sid);
    Emp e=EmpDao.getEmployeeById(id);

    out.print("<form action='EditServlet2'
    method='post'>");out.print("<table>");
    out.print("<tr><td></td><td><input type='hidden' name='id'
    value='"+e.getId()+"'/></td></tr>");out.print("<tr><td>Name:</td><td><input type='text'
    name='name'
value='"+e.getName()+"'/></td></tr>");
    out.print("<tr><td>Password:</td><td><input type='password'
    name='password'
value='"+e.getPassword()+"'/></td></tr>");
    out.print("<tr><td>Email:</td><td><input type='email'
    name='email'
value='"+e.getEmail()+"'/></td></tr>");
    out.print("<tr><td>Country:</td><td>");
    out.print("<select name='country'
    style='width:150px'>");
```

```
        out.print("<option>India</option>");
        out.print("<option>USA</option>");
        out.print("<option>UK</option>");
        out.print("<option>Other</option>");
        out.print("</select>");
        out.print("</td></tr>");
        out.print("<tr><td colspan='2'><input type='submit' value='Edit & Save '/></td></tr>");
        out.print("</table>");
        out.print("</form>");

        out.close();
    }
}
```

EditServlet2.java

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

import javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;

String sid=request.getParameter("id");
int id=Integer.parseInt(sid);

import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
@WebServlet("/EditServlet2")
public class EditServlet2 extends HttpServlet {
  protected void doPost(HttpServletRequest request, HttpServletResponse
response)throwsServletException, IOException
  {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();

      String sid=request.getParameter("id");
      int id=Integer.parseInt(sid);
```

```
        String name=request.getParameter("name");
        String
        password=request.getParameter("password");
        String email=request.getParameter("email");
        String country=request.getParameter("country");

        Emp e=new Emp();
        e.setId(id);
        e.setName(name);
        e.setPassword(password);
        e.setEmail(email);
        e.setCountry(country);

        int
        status=EmpDao.update(e);
        if(status>0){
          response.sendRedirect("ViewServlet");
        }else{
          out.println("Sorry! unable to update record");
        }
        out.close();
    }
}
```

DeleteServlet.java

```
import java.io.IOException;
import javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
@WebServlet("/DeleteServlet")
public class DeleteServlet extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse
response)throwsServletException, IOException
  {
    String
    sid=request.getParameter("id");int
    id=Integer.parseInt(sid);
    EmpDao.delete(id);
    response.sendRedirect("ViewServlet"
    );
```

```
    }
}
```

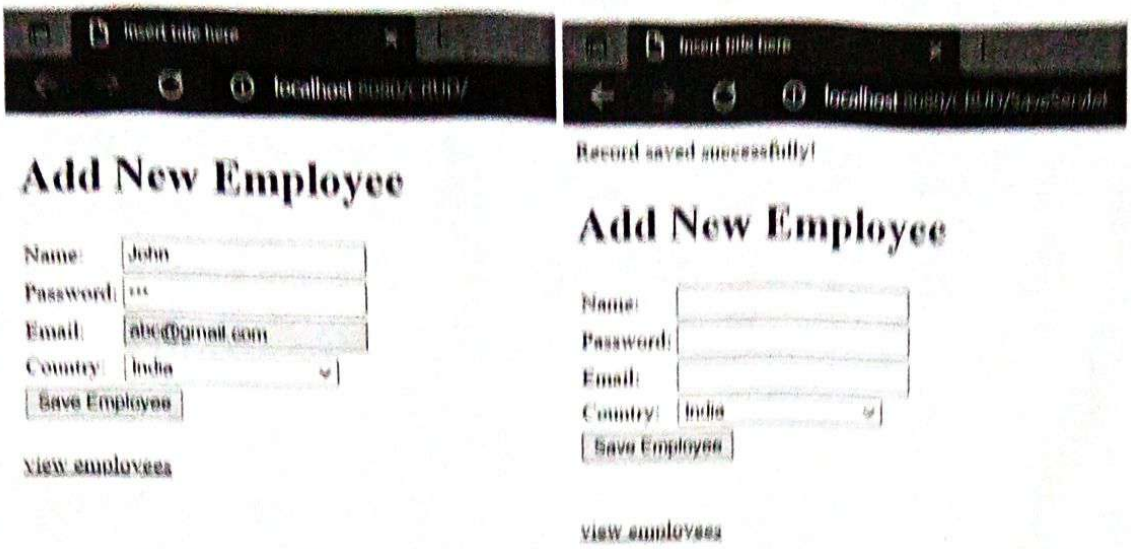
Index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Insert title here</title>
</head>
<body>
<h1>Add New Employee</h1>
<form action="SaveServlet" method="post">
<table>
<tr><td>Name:</td><td><input type="text" name="name"/></td></tr>
<tr><td>Password:</td><td><input type="password" name="password"/></td></tr>
<tr><td>Email:</td><td><input type="email" name="email"/></td></tr>
<tr><td>Country:</td><td>
<select name="country" style="width:150px">
<option>India</option>
<option>USA</option>
<option>UK</option>
<option>Other</option>
</select>
</td></tr>
<tr><td colspan="2"><input type="submit" value="Save Employee"/></td></tr>
</table>
</form>
<br/>
<a href="ViewServlet">view employees</a>
</body>
</html>
```

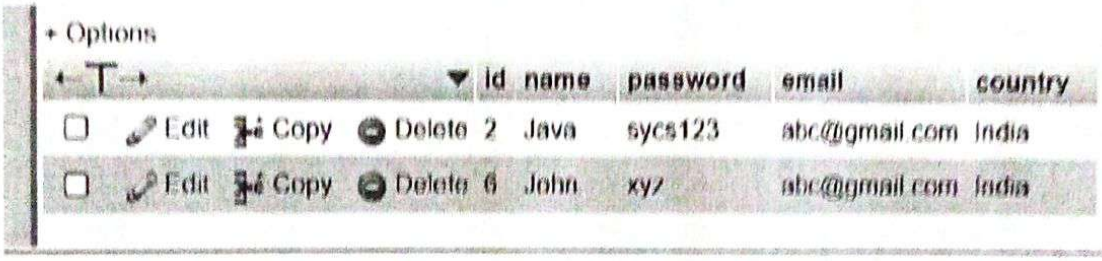
Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1"
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">
 <servlet>
   <servlet-name>SaveServlet</servlet-name>
   <servlet-class>SaveServlet</servlet-class>
 </servlet>
 <servlet>
   <servlet-name>EditServlet</servlet-name>
   <servlet-class>EditServlet</servlet-class>
 </servlet>
 <servlet>
   <servlet-name>EditServlet2</servlet-name>
   <servlet-class>EditServlet2</servlet-class>
 </servlet>
 <servlet>
   <servlet-name>DeleteServlet</servlet-name>
   <servlet-class>DeleteServlet</servlet-class>
 </servlet>

 <servlet>
   <servlet-name>ViewServlet</servlet-name>
   <servlet-class>ViewServlet</servlet-class>
 </servlet>
 <servlet-mapping>
   <servlet-name>SaveServlet</servlet-name>
   <url-pattern>/SaveServlet</url-pattern>
 </servlet-mapping>
 <servlet-mapping>
   <servlet-name>EditServlet</servlet-name>
   <url-pattern>/EditServlet</url-pattern>
 </servlet-mapping>
 <servlet-mapping>
   <servlet-name>EditServlet2</servlet-name>
   <url-pattern>/EditServlet2</url-pattern>
 </servlet-mapping>
 <servlet-mapping>
   <servlet-name>DeleteServlet</servlet-name>
   <url-pattern>/DeleteServlet</url-pattern>
 </servlet-mapping>
```

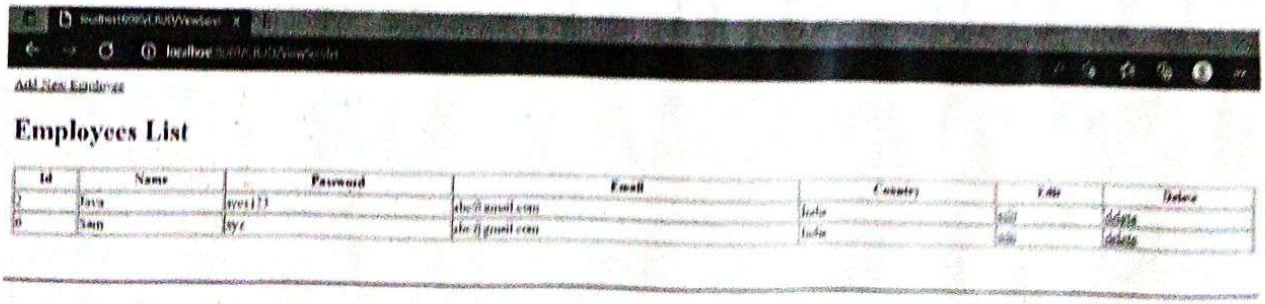
Output:



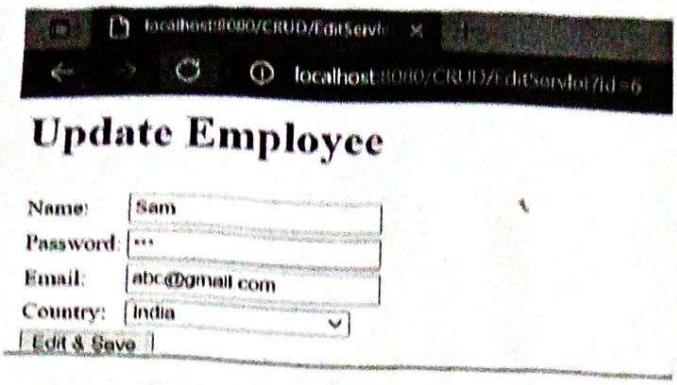
As you can see in above snapshot its showing record is successfully saved lets check whether the data is updated on xampp or not.

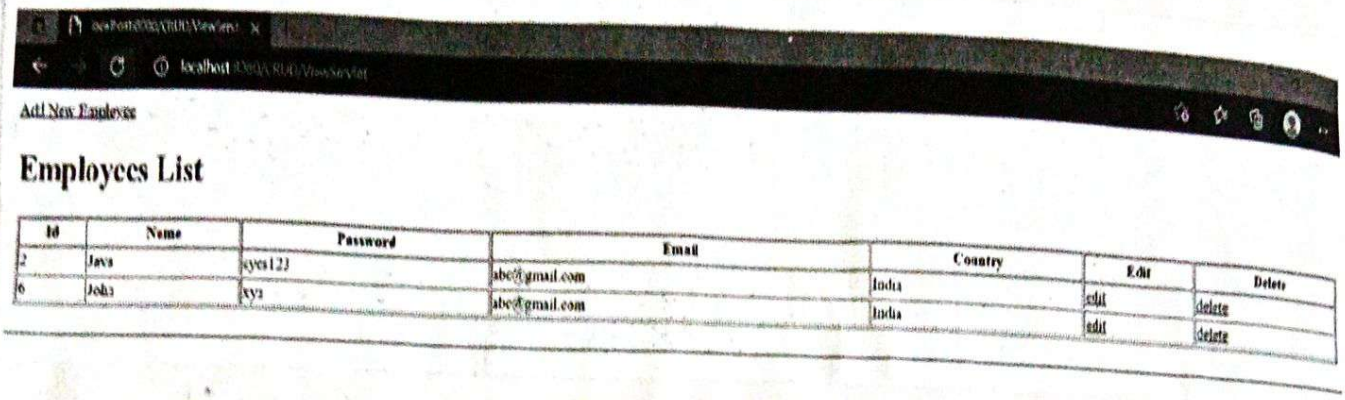


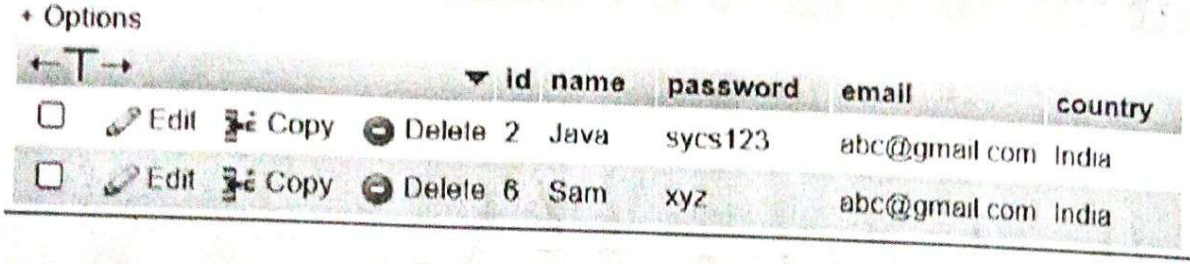
After Clicking on View Employees.



After Clicking on Edit.

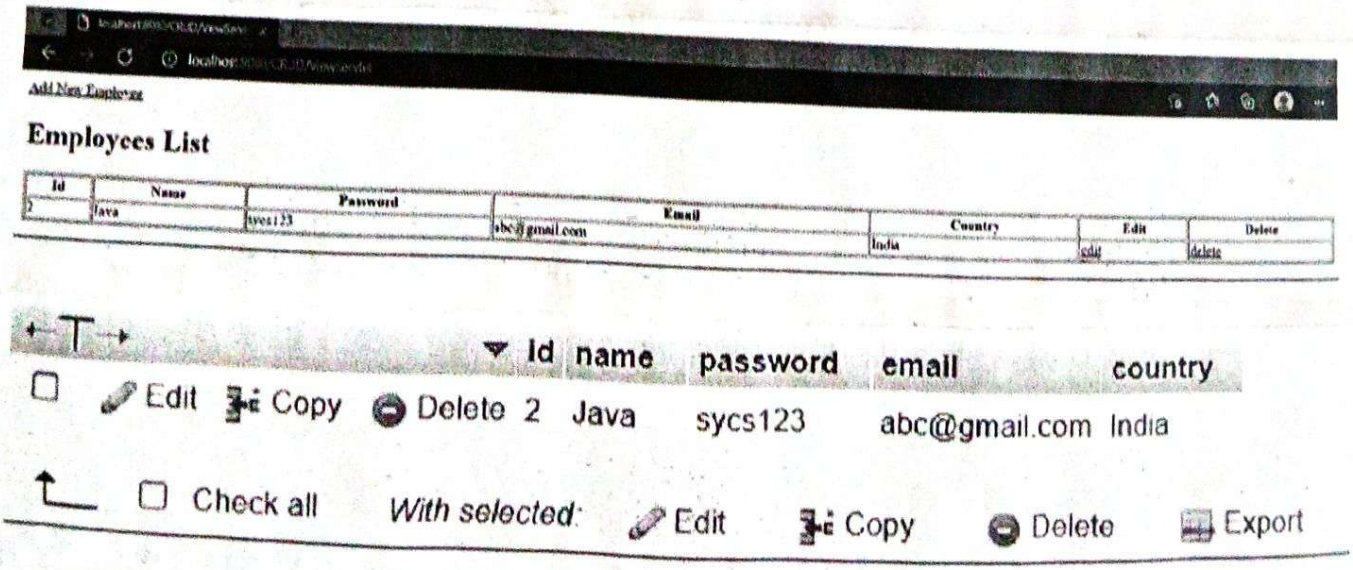






As you can see in above snapshot record is updated in xampp as well as in the web browser.

After clicking on delete.



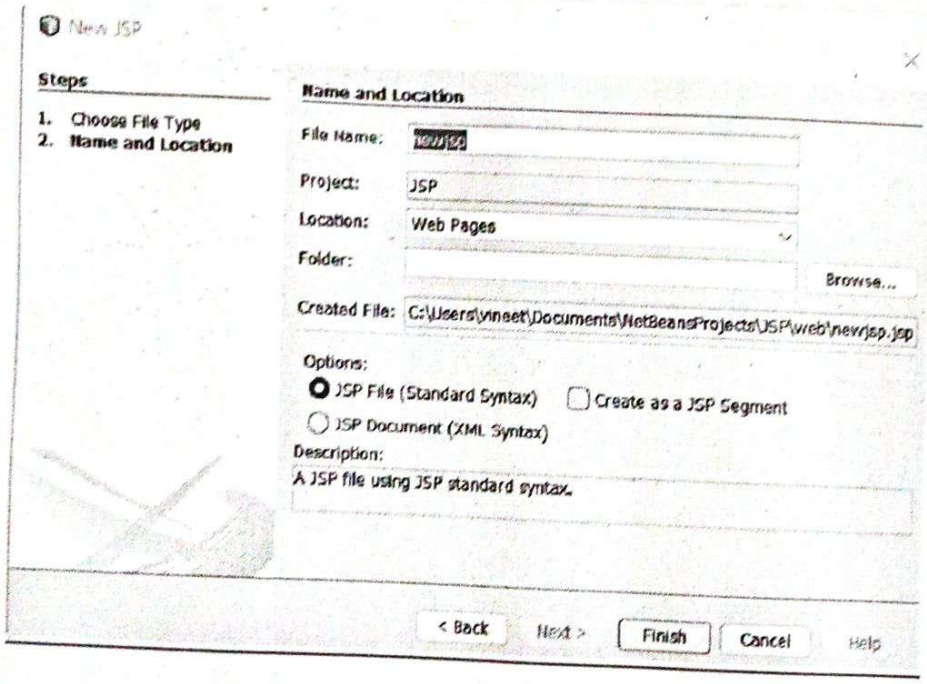
As you can see in above snapshot record is deleted in xampp as well as in the web browser.

Conclusion : Successfully implemented CRUD operation using servlet in java.

PRACTICAL NO : 07

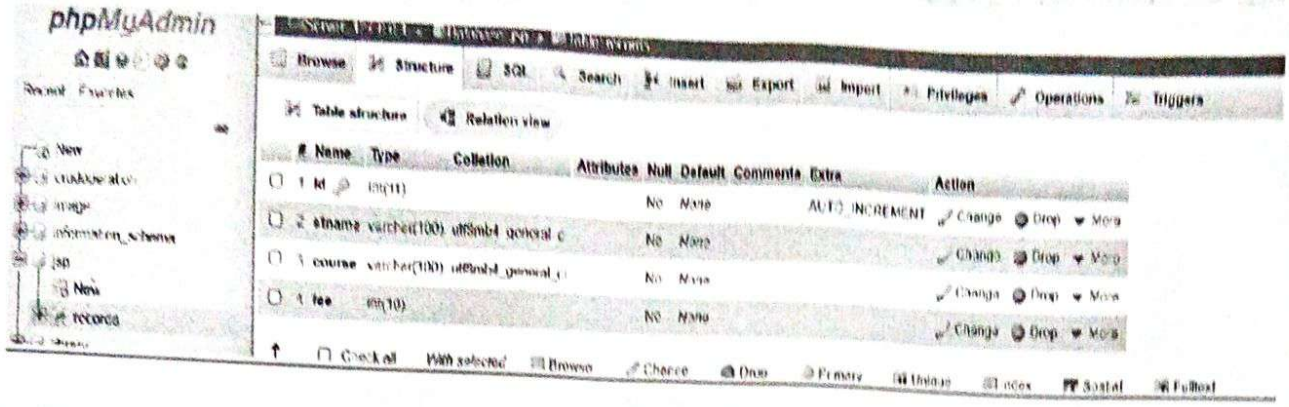
Aim : Create Employees table in EMP database. Perform select, insert, update, delete operations on Employee table using JSP.

Step 1 : Create a java web application --> Create JSP files --> Expand your project --> Right click on Web Pages folder --> Select JSP File --> Give name of the file and click on finish.



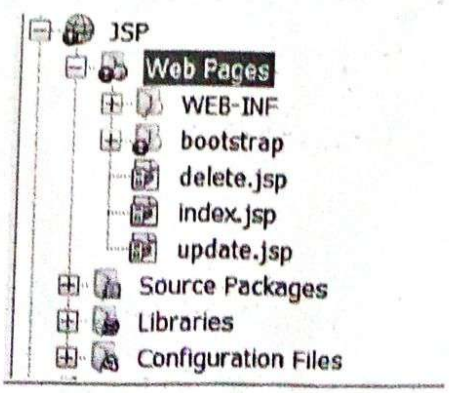
Step 2 : Expand your project and add Mysql JDBC library in libraries folder.

Step 3 : Create database on xampp.



Step 4: Click on services in IDE --> Right click on database --> Click on New connection --> Select Mysql (connector/J driver) --> Then one window will appear in which you have to enter your database name and test the connection, If connection is established click on finish.

Step 5: Download bootstrap 5.1.3 zip file for designing faster --> Extract to it in folder --> Then import that folder inside your project in Web pages folder. (Note: you can ignore the error in bootstrap file).



Step 6 : Create JSP files and write code inside it.

Index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<%@page import="java.sql.*" %>
<%

  if(request.getParameter("submit")!=null)
  {
    String name =
    request.getParameter("sname"); String
    course = request.getParameter("course");
    String fee = request.getParameter("fee");

    Connection con;
    PreparedStatement
    pst;ResultSet rs;

    Class.forName("com.mysql.jdbc.Driver");
    con = DriverManager.getConnection("jdbc:mysql://localhost/jsp","root","");
    pst = con.prepareStatement("insert into records(stname,course,fee)values(?,?,?)");
    pst.setString(1, name);
    pst.setString(2,
    course);
    pst.setString(3, fee);
    pst.executeUpdate();

   %>
  <script
     >
  alert("Record Added");
  </script>
  <%
  }
 %>
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
    <link href="bootstrap/css/bootstrap.css" rel="stylesheet" type="text/css"/>
    <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet" type="text/css"/>
  </head>
  <body>
    <h1>Student Registation System Crud using-JSP</h1>
    </br>
    <div class="row">
      <div class="col-sm-4">
        <form  method="POST" action="#" >

          <div alight="left">
            <label class="form-label">Student Name</label>
            <input type="text" class="form-control" placeholder="Student Name"
name="sname"id="sname" required >
          </div>

          <div alight="left">
            <label class="form-label">Course</label>
            <input type="text" class="form-control" placeholder="Course" name="course"
required      id="course"
>
          </div>

          <div alight="left">
            <label class="form-label">Fee</label>
            <input type="text" class="form-control" placeholder="Fee" name="fee" id="fee" required
>
          </div>
            </br>

          <div alight="right">
            <input type="submit" id="submit" value="submit" name="submit" class="btn btn-info">
            <input type="reset" id="reset" value="reset" name="reset" class="btn btn-warning">
          </div>

        </form>
      </div>

      <div class="col-sm-8">
```

```
        <div class="panel-body">
          <table id="tbl-student" class="table table-responsive table-bordered" cellpadding ="0"
width="100%">

  </body>
</html>
            <thead>
              <tr>
                  <th>Student Name</th>
                  <th>Course</th>
                  <th>Fee</th>
                  <th>Edit</th>
                  <th>Delete</th>
              </tr>

              <%

               Connection con;
               PreparedStatement
               pst;ResultSet rs;

               Class.forName("com.mysql.jdbc.Driver");
               con = DriverManager.getConnection("jdbc:mysql://localhost/jsp","root","");

                String query = "select * from
                records";Statement st =
                con.createStatement();

                 rs = st.executeQuery(query);

                   while(rs.next())
                   {
                     String id = rs.getString("id");
                %>

            <tr>
              <td><%=rs.getString("stname") %></td>
              <td><%=rs.getString("course") %></td>
              <td><%=rs.getString("fee") %></td>
              <td><a href="update.jsp?id=<%=id%>">Edit</a></td>
              <td><a href="delete.jsp?id=<%=id%>">Delete</a></td>
            </tr>

              <%

              }
```

```
                %>
              </table>
            </div>
          </div>
        </div>
```

Update.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.*" %>

<%
  if(request.getParameter("submit")!=null)
  {
    String id = request.getParameter("id");
    String name =
    request.getParameter("sname"); String
    course = request.getParameter("course");
    String fee = request.getParameter("fee");

    Connection con;
    PreparedStatement
    pst;ResultSet rs;

    Class.forName("com.mysql.jdbc.Driver");
    con = DriverManager.getConnection("jdbc:mysql://localhost/jsp","root","");
    pst = con.prepareStatement("update records set stname = ?,course =?,fee= ? where id = ?");
    pst.setString(1, name);
    pst.setString(2,
    course);
    pst.setString(3, fee);
    pst.setString(4, id);
    pst.executeUpdate();

    %>

    <script>
      alert("Record Updated");
    </script>
  <%
  }

%>

<!DOCTYPE html>
<html>
```

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>JSP Page</title>

  <link href="bootstrap/css/bootstrap.css" rel="stylesheet" type="text/css"/>
  <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet" type="text/css"/>
</head>
<body>
  <h1>Student Update</h1>


  <div class="row">
    <div class="col-sm-4">
      <form  method="POST" action="#" >

        <%
          Connection con;
          PreparedStatement
          pst;ResultSet rs;

          Class.forName("com.mysql.jdbc.Driver");
          con =

          DriverManager.getConnection("jdbc:mysql://localhost/jsp","root","");

          String id = request.getParameter("id");

          pst = con.prepareStatement("select * from records where id =
          ?");pst.setString(1, id);
          rs = pst.executeQuery();

          while(rs.next())
          {

        %>
        <div alight="left">
          <label class="form-label">Student Name</label>
          <input type="text" class="form-control" placeholder="Student Name"
value="<%=rs.getString("stname")%>" name="sname" id="sname" required >
        </div>

        <div alight="left">
          <label class="form-label">Course</label>
          <input type="text" class="form-control" placeholder="Course" name="course"
value="<%=rs.getString("course")%>" id="course" required >
        </div>
```

```
            <div alight="left">
              <label class="form-label">Fee</label>
              <input type="text" class="form-control" placeholder="Fee" name="fee"
id="fee"value="<%= rs.getString("fee")%>" required >
            </div>

           <% } %>

              </br>

           <div alight="right">
              <input type="submit" id="submit" value="submit" name="submit" class="btn btn-info">
              <input type="reset" id="reset" value="reset" name="reset" class="btn btn-warning">
           </div>

              <div align="right">

                <p><a href="index.jsp">Click Back</a></p>


              </div>

         </form>
       </div>
     </div>

  </body>
</html>
```

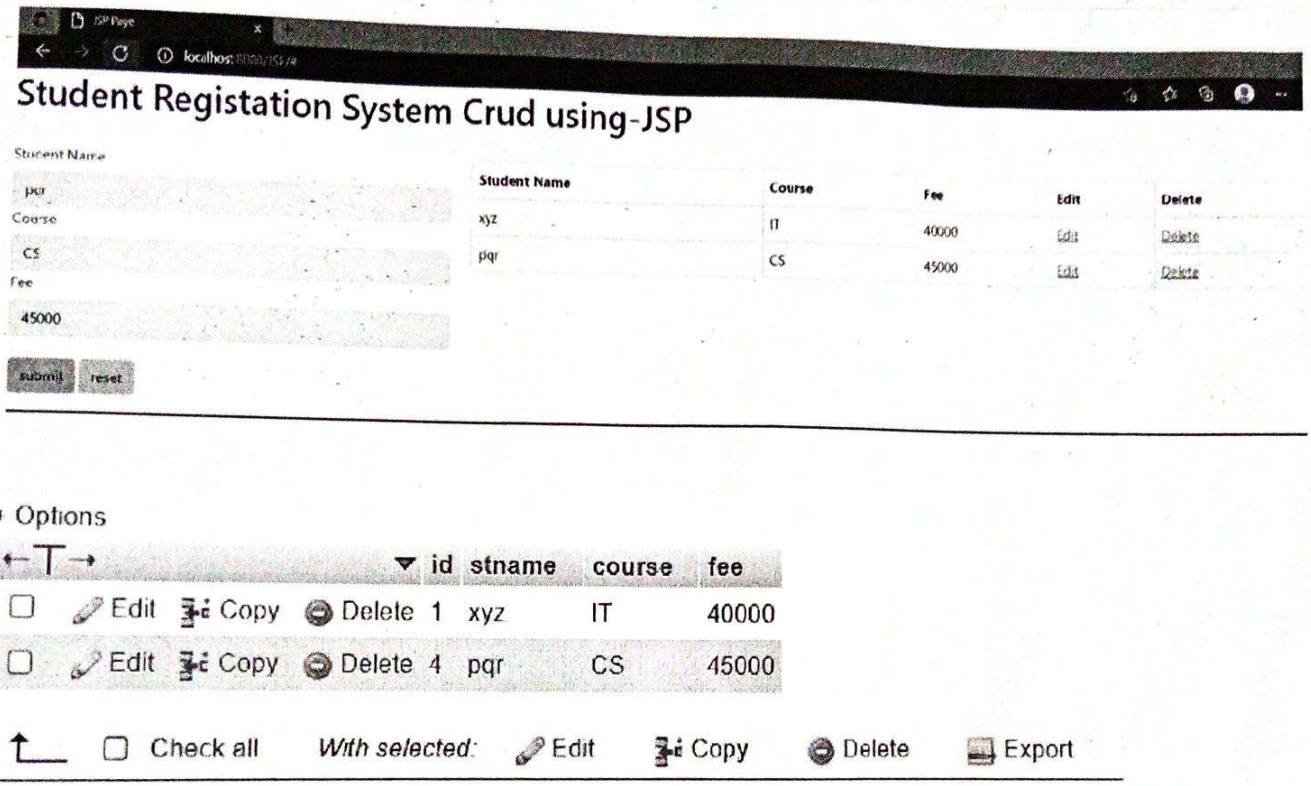
Delete.jsp

```
<%@page import="java.sql.*" %>


<%
    String id =
    request.getParameter("id");
    Connection con;
    PreparedStatement
    pst;
    ResultSet rs;

    Class.forName("com.mysql.jdbc.Driver");
    con =
    DriverManager.getConnection("jdbc:mysql://localhost/jsp","root","");
```

```
pst = con.prepareStatement("delete from records where id = ?");
pst.setString(1, id);
pst.executeUpdate();

%>

<script>

  alert("Record Deleted");

</script>
```

Output:

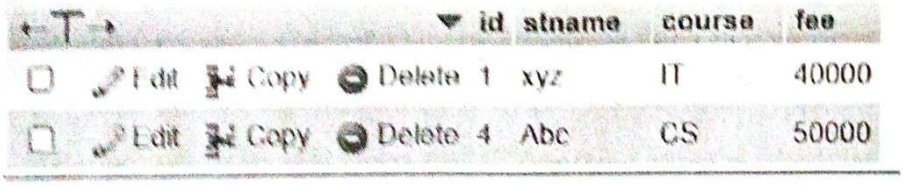
Entering the value and click on submit the value will get stored and it will be displayed in the table



After Clicking on edit

Student Update

Student Name	Course	Fee	Edit	Delete
xyz	IT	40000	Edit	Delete
Abc	CS	50000	Edit	Delete

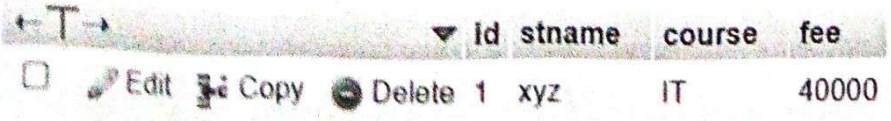


After clicking on delete:

Student Registation System Crud using-JSP

Student Name	Course	Fee	Edit	Delete
xyz	IT	40000	Edit	Delete

+ Options



Check all With selected: Edit Copy Delete Export

Conclusion: Successfully implemented servlet select, insert, update, delete operation using JSP in java.

PRACTICAL NO: 08

Aim: Write a student class with three properties. The useBean action declares a JavaBean for use in a JSP. Write Java Application to access JavaBeans Properties.

Step 1: Create a java web application --> Create JSP files --> Expand your project --> Right click on Web Page folder --> Select JSP file --> Give name of the file and click on Finish.

Step 2: Create 1 java class file and import getters and setters in it.

StudentBean.java

```
package bean;

public class StudentBean {

  private String course;

  public String

  getCourse() {
    return course;
  }

  public void setCourse(String
    course) {this.course = course;
  }


  private String lastname;

  public String

  getLastname() {
    return lastname;
  }


  public void setLastname(String
    lastname) {this.lastname = lastname;
  }


  private String firstname;


  public String


  getFirstname() {
    return firstname;
```

```
    }

    public void setFirstname(String firstname) {
      this.firstname = firstname;
    }


}
```

Step 3 : Create 2 JSP File, 1 html & write code inside it:

Index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form action="first.jsp" method="get">
     Enter First Name: <input type="text"
     name="firstname"/><br/><br/>Enter Last Name: <input
     type="text" name="lastname"/><br/><br/> Enter Course: <input
     type="text" name="course"/><br/><br/>
     <input type="submit" value="submit"/>

    </form>
  </body>
</html>
```

First.jsp

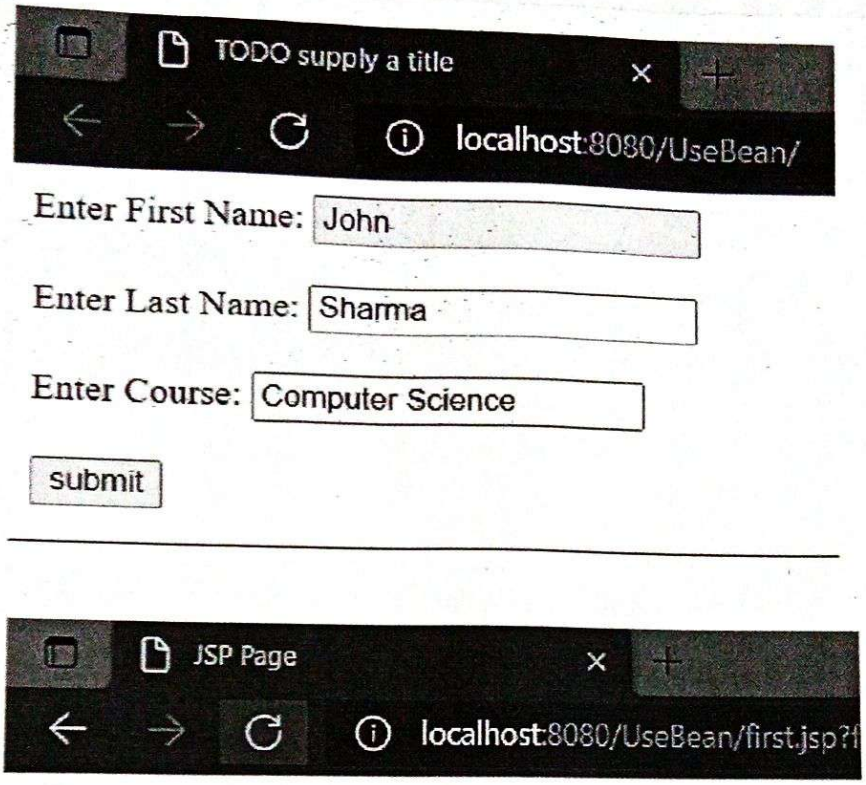
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <% String
    firstname=request.getParameter("firstname");String
    lastname=request.getParameter("lastname"); String
    course=request.getParameter("age");
```

```
    %>
    <jsp:useBean id = "b1" class = "bean.StudentBean" scope="request"/>
    <jsp:setProperty name = "b1" property = "firstname"/>
    <jsp:setProperty name = "b1" property = "lastname"/>
    <jsp:setProperty name = "b1" property = "course"/>
    <jsp:forward page="second.jsp"/>
  </body>
</html>
```

Second.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <% String
    firstname=request.getParameter("firstname");String
    lastname=request.getParameter("lastname"); String
    course=request.getParameter("age");
    %>
    <jsp:useBean id = "b1" class = "bean.StudentBean" scope="request"/>
    Student First Name: <jsp:getProperty name = "b1" property =
    "firstname"/><br/><br/>Student Last Name: <jsp:getProperty name = "b1"
    property = "lastname"/><br/><br/> Student Course: <jsp:getProperty name = "b1"
    property = "course"/><br/><br/>
  </body>
</html>
```

Output :


TODO supply a title
localhost:8080/UseBean/
Enter First Name: John
Enter Last Name: Sharma
Enter Course: Computer Science
submit
JSP Page
localhost:8080/UseBean/first.jsp?f
Student First Name: John
Student Last Name: Sharma
Student Course: Computer Science

Conclusion : Successfully created Java web application to access JavaBeans properties and also created useBean action that declares a JavaBean for use in a JSP.

PRACTICAL NO: 09

Aim: Design application using Struts 2: Application must accept username and greet user when command button is pressed.

Step 1: Download struts2-suite-1.3.5 zip file --> Extract in a folder --> Open netbeans 8.2 --> Click on tools --> Click on plugins --> Go to downloaded --> Click on add plugins --> Access that folder where you have extracted the file --> you will get 3 files with extension .nbm in that folder import them and click on install.

Step 2: Create java a web application --> Click on file --> New Project --> Java Web --> Web Application --> Give name of the project --> Select Server --> Select Struts2 --> Click on Finish.

Step 3: Create 1 package and inside that add 1 java class file --> Create getters and setters in it.

Hello_action.java

```
package action_jsp;

import com.opensymphony.xwork2.ActionSupport;

// everything in here must be kept public to have package level

access.public class hello_action extends ActionSupport
{
  private String name;
  // getter and setter methods for the form element value

  public String getName()
  {
    return name;
  }
  public void setName(String name)
  {
    this.name = name;
  }
//method that returns success or failure on
  actionpublic String execute()
  {
    return "success";
  }
}
```

Step 4 : Create 1 folder in Web Pages folder and inside that add 2 jsp file.

index.jsp

```
<%@page contentType = "text/html" pageEncoding = "UTF-8"%>
<%@taglib  prefix = "s" uri = "/struts-tags" %>

<html>
  <body>
    <s:form method = "post" action = "hello_action">

      <s:label name = "label1"/>
      <s:textfield value="Enter your name" name = "name"/>
      <s:submit value = "CLICK" name = "submit" align="left"/>

    </s:form>
  </body>
</html>
```

result.jsp

```
<%@page contentType = "text/html" pageEncoding = "UTF-8"%>
<%@taglib prefix = "s" uri = "/struts-tags" %>
<html>
<body>
    <h1>Hello <s:property value = "name"/></h1>
</body>
</html>
```

Step 5: Check your web.xml and struts.xml both files will be in configuration file.

Struts.xml

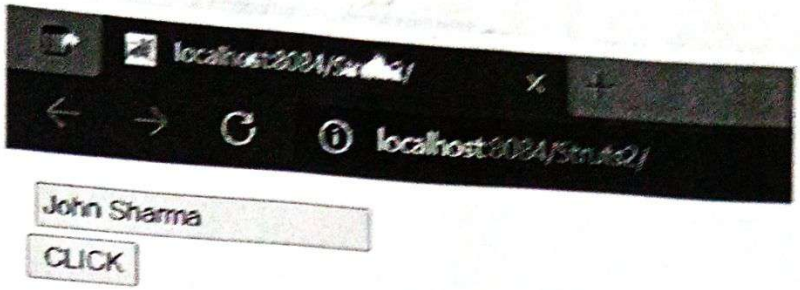
```
<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration
2.0//EN""http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <include file="example.xml"/>
  <!-- Configuration for the default package. -->
  <package name="default" extends="struts-default">
    <action name = "hello_action" class = "action_jsp.hello_action" method = "execute">
    <result name = "success">result.jsp</result>
    <result name = "failure">index.jsp</result>
    </action>
 </package>
</struts>
```

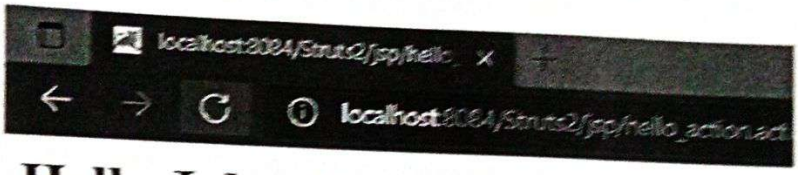
Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web- app_3_1.xsd">
  <filter>
    <filter-name>struts2</filter-name>
    <filter-class>org.apache:struts2.dispatcher.FilterDispatcher</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <session-config>
    <session-
```

```
        timeout>30
      </session-timeout>
    </session-config>
    <welcome-file-list>
      <welcome-file>jsp/index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

Output:





Hello John Sharma

Conclusion : Successfully designed application using Struts2.

PRACTICAL NO : 11

1. Explain architecture of RMI ?

→ i. RMI stands for Remote Method Invocation.
ii. It is a mechanism that allows an object residing in one system (JVM) to access / invoke an object running on another JVM.
iii. RMI is used to build distributed applications; it provides remote communication between Java programs.
iv. It is provided in the package java.rmi.

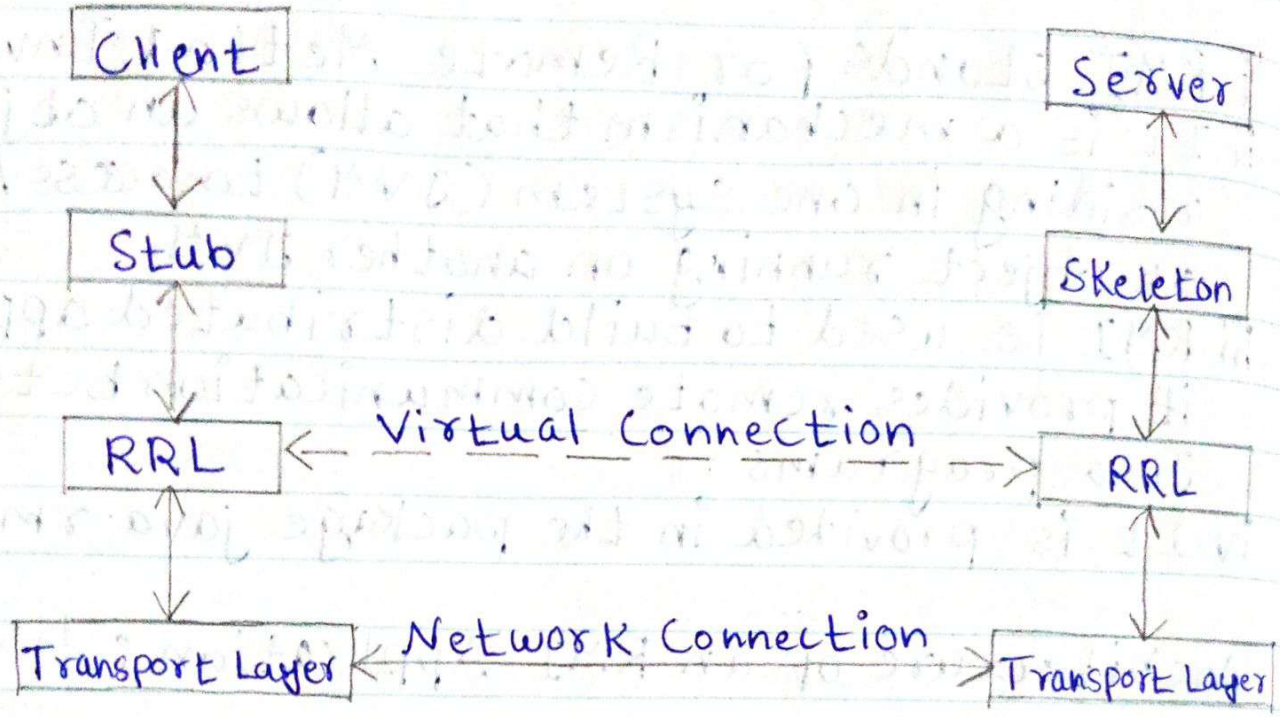
Architecture of an RMI Application :-

In an RMI application, we write two programs, a server program (resides on the server) and a client program (resides on the client).

a) Inside the server program, a remote object is created and reference of that object is made available for the client (using the registry).

b) The client program requests the remote objects on the server and tries to invoke its methods.

The following diagram shows the architecture of an RMI application :-



Let us discuss the components of this architecture :-

a) Transport Layer - This layer connects the client and the server. It manages the existing connection and also set up a new connection.

b) Stub - A stub is a representation of the remote object at client. it resides in client system.

c) Skeleton - This is the object which reside on server side.

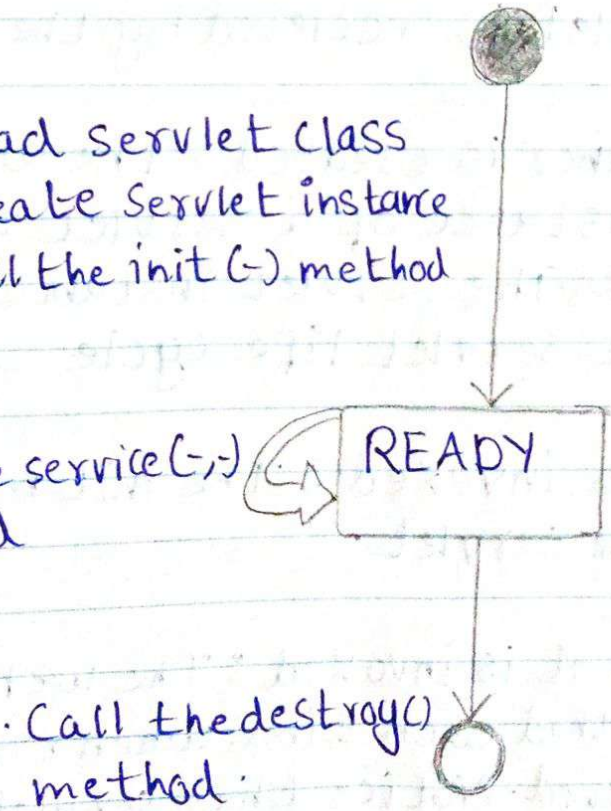
d) RRL (Remote Reference Layer) - It is the layer which manages the references made by client to remote object.

2 Explain Servlet life cycle ?

→ i. Servlet technology is used to create a web application (resides at server side and generates a dynamic page or web page).

ii. The web container maintains the life cycle of the servlet :

a) Servlet class is loaded.
b) Servlet instance is created.
c) init method is invoked.
d) service method is invoked.
e) destroy method is invoked.



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

a. Servlet class is loaded : The class loader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is recieved by the web container.

b. Servlet instance is created : The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle.

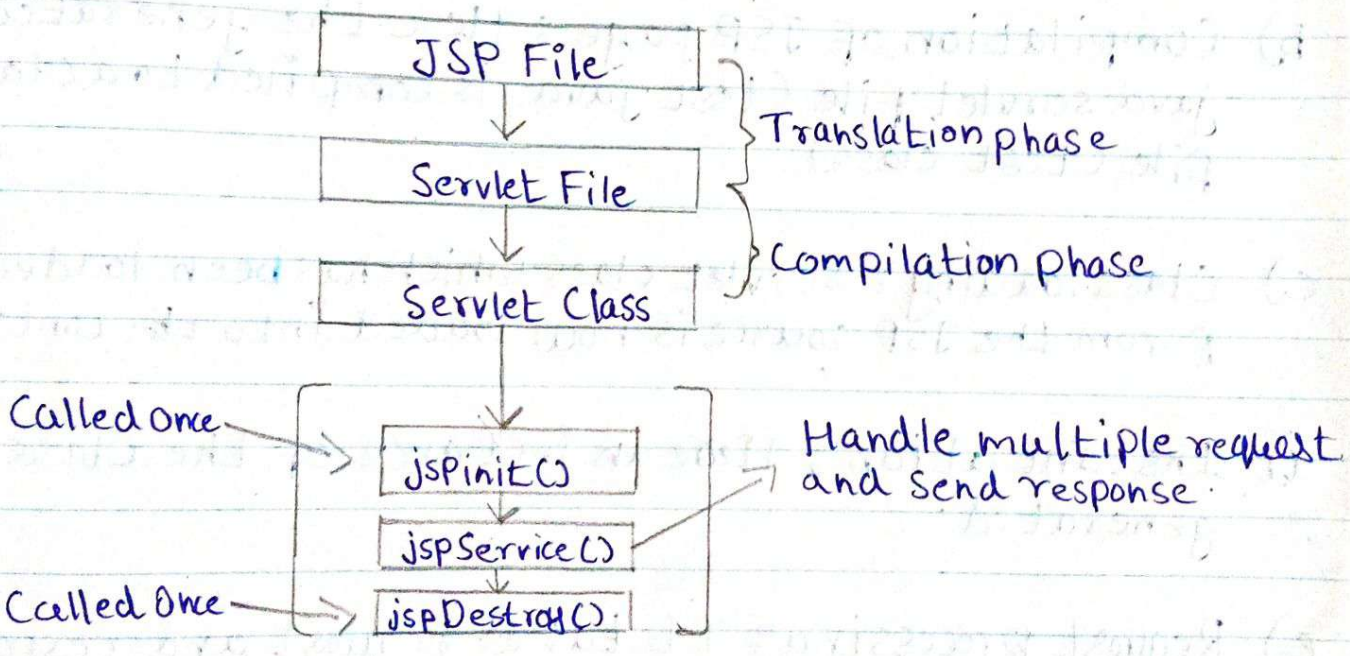
c init method is invoked : The init method is used to initialize the servlet.

d. service method is invoked : The web container calls the service method each time when request for the servlet is recieved. Notice that servlet is initialized only

e. destroy method is invoked : The web container calls the destroy method before removing servlet instance.

3. Explain JSP life cycle ?

→ i. A Java Server Page life cycle is defined as the process that started with it's creation which later translated to servlet and afterward servlet lifecycle comes into play. This is how the process goes on until the destruction.



JSP Life Cycle

Following steps are involved in the JSP Life Cycle:-

- Translation of JSP page to Servlet.
- Compilation of JSP page (Compilation of JSP into test.java).
- Initialization (jspInit() method invoked by container).
- Instantiation (object of the generated Servlet created).

- Request processing (_jspService() is invoked).
- JSP Cleanup (jspDestroy() method is invoked).

a) Translation of JSP page to servlet: This is the first step of the JSP life cycle. This translation phase deals with the syntactic correctness of JSP. Here test.jsp file is translated to test.java.

b) Compilation of JSP page: Here the generated java servlet file (test.java) is compiled to a class file (test.class).

c) Class loading: Servlet class which has been loaded from the JSP source is now loaded into the container.

d) Instantiation: Here an instance of the class is generated.

e) Request processing: It takes request and response objects as parameters. This method cannot be overridden.

f) JSP Cleanup: In order to remove the JSP from the use by the container or to destroy the method for servlet jspDestroy() method is used. This method is called once, if you need to perform any cleanup task like closing open files.

4. Explain Filters ?

→ i. A filter is an object that is invoked at the preprocessing and post processing of a request.
ii. It is mainly used to perform filtering tasks such as conversion, logging, compression, encryption and decryption, input validation etc.
iii. Filters differ from web components in that filters usually do not themselves create a response. Instead, a filter provides functionality that can be "attached" to any kind of web resource for filtering task.
iv. The servlet fiter is pluggable, i.e its entry is defined in the web.xml file, if we remove the entry of filter from the web.xml file, filter will be removed automatically and we don't need to change the servlet. So, the maintenance cost will be less.

The main tasks that a filter can perform as follows :-

a) Query the request and act accordingly.
b) Block the request-and-response pair from passing any further.
c) Modify the request headers and data. You do this by providing a customized version of response.

d) Modify the response headers and data. You do this by providing a customized version of response.
e) Interact with external resources.

Usage of Filter :-

a) recording all incoming requests.
b) logs the IP addresses of the computers from which the requests originate.
c) Conversion.
d) data compression.
e) encryption and decryption.
f) input validation etc.

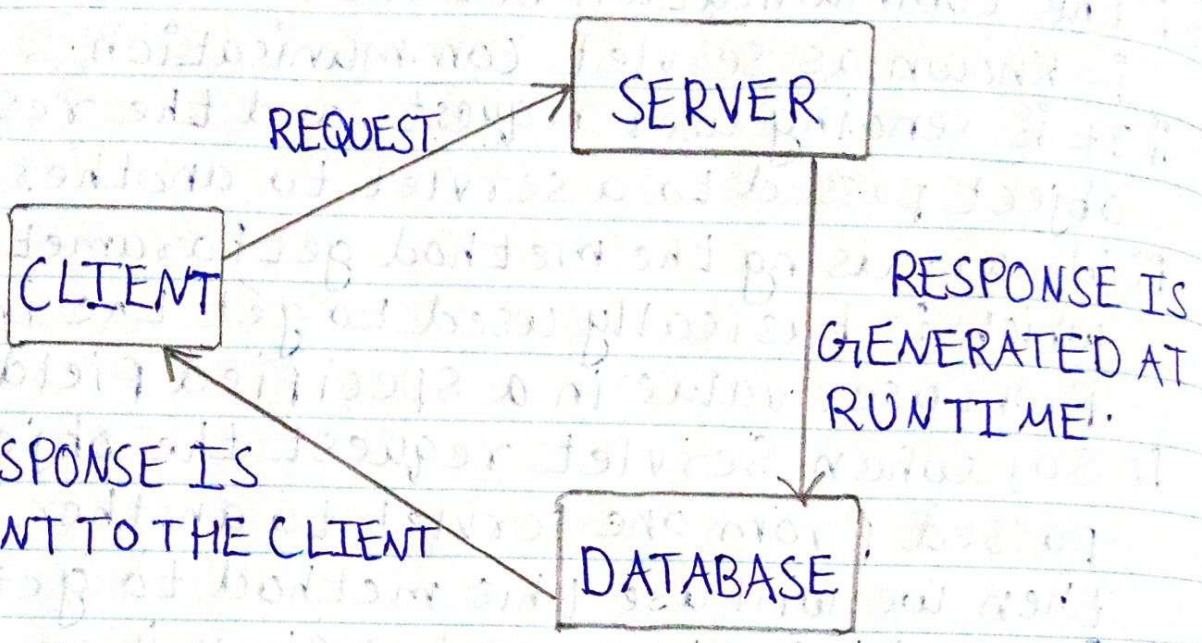
Advantages of Filter :-

i. Fiter is pluggable.
ii. One filter don't have dependency onto another resource.
iii. Less Maintenance.

5. Explain interaction between servlet ?,

→ i. The communication between the Java servlets is known as servlet communication.
ii. It is sending user request, and the response object passed to a servlet to another servlet.
iii. We are using the method getParameter(), which is basically used to get the input from user value in a specified field.
iv. So, when Servlet request the object is passed from one servlet to another servlet, then we will use this method to get the input that the user has given in an HTML/JSP form.
v. Java servlets are the server-side programs (running inside a web server) that can handle the clients' requests and revert the customized or dynamic response for each request.
vi. The dynamic response, which is responded could be based on the user's input like search, online shopping, online transication with data retrieved from databases or other applications, or time-sensitive data (such as news and stock prices).

Java Servlets typically run on the HTTP protocol.



Request Dispatcher in Servlet :-

i. It is an interface that provides the facility of dispatching the requesting to other resources, it may be HTML, servlet, or JSP.

ii. There are two Methods defined in the RequestDispatcher interface they are:

a) public void forward: (ServletRequest srequest, ServletResponse sresponse) throws ServletException, java.io.IOException: This method forwards a request from a servlet to other resources (i.e., servlet, JSP or HTML file)

b) public void include: Includes the content of a resource (servlet, JSP page, or HTML file) in the response.