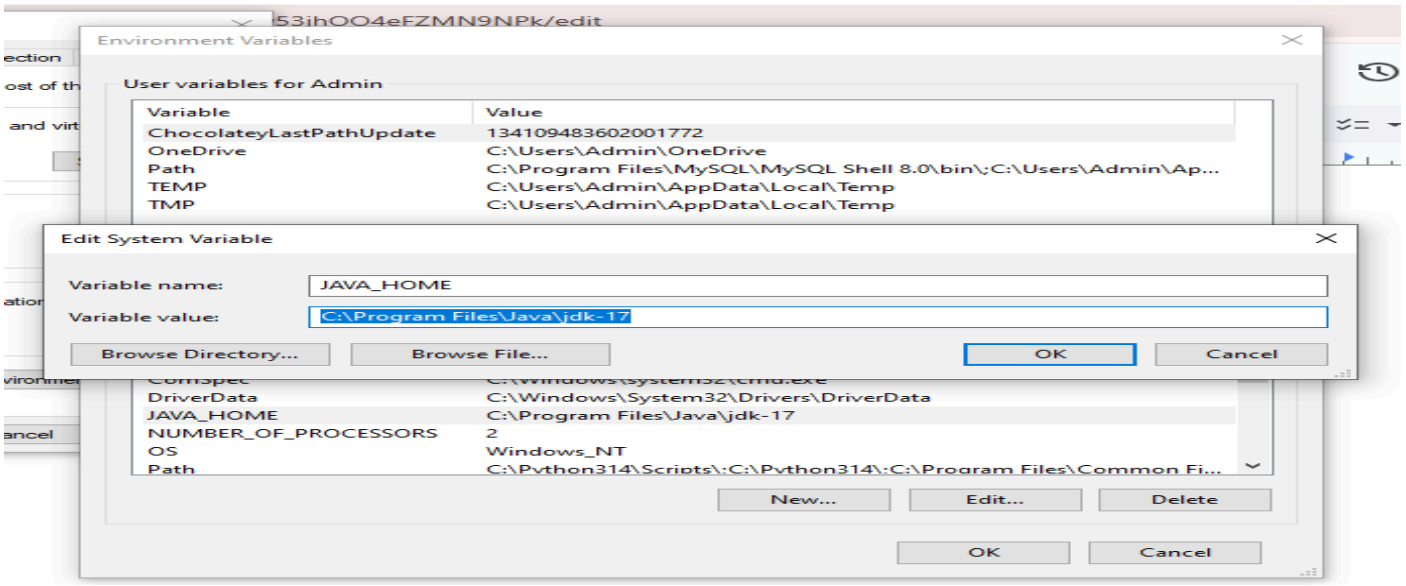
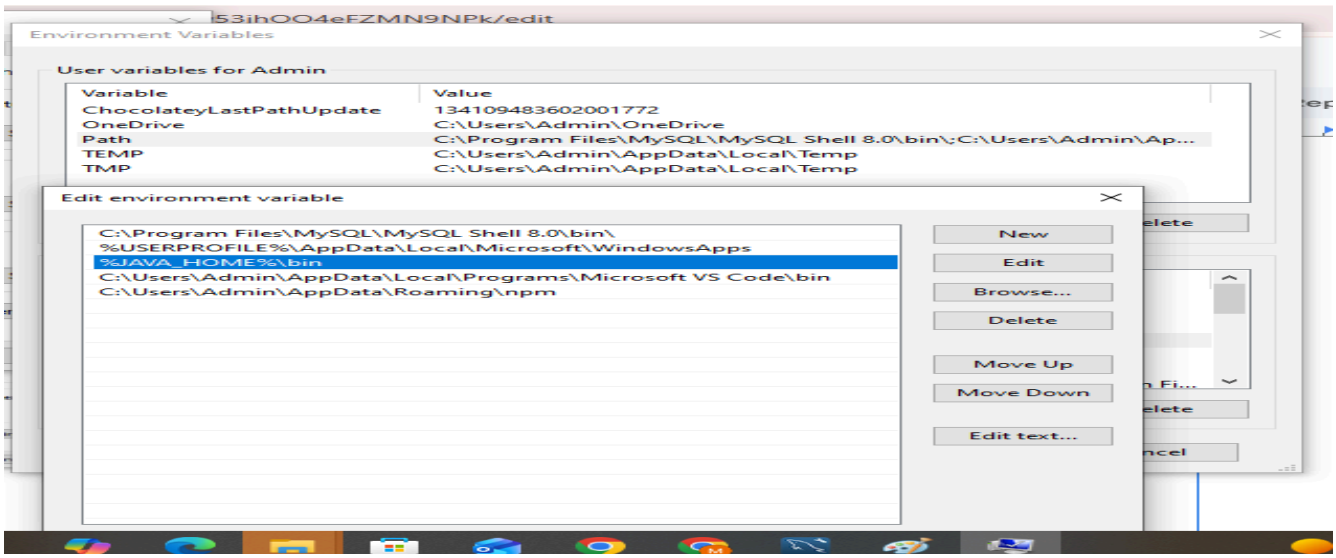
Steps to configure Eclipse:

Download

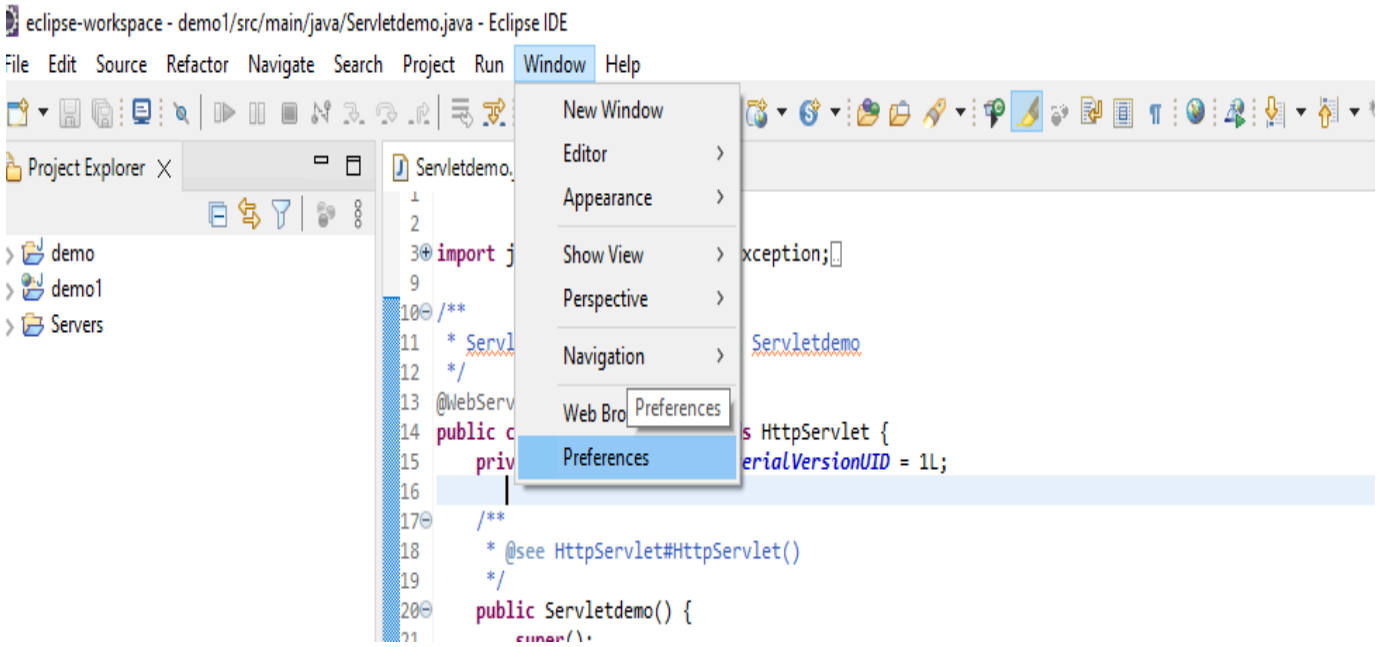
- **eclipse-jee-2023-06-R-win32-x86_64**
- **Jdk 17**
- **Apache-tomcat-9.0.91-windows-x64**
- **Mysql j connector for jdbc**

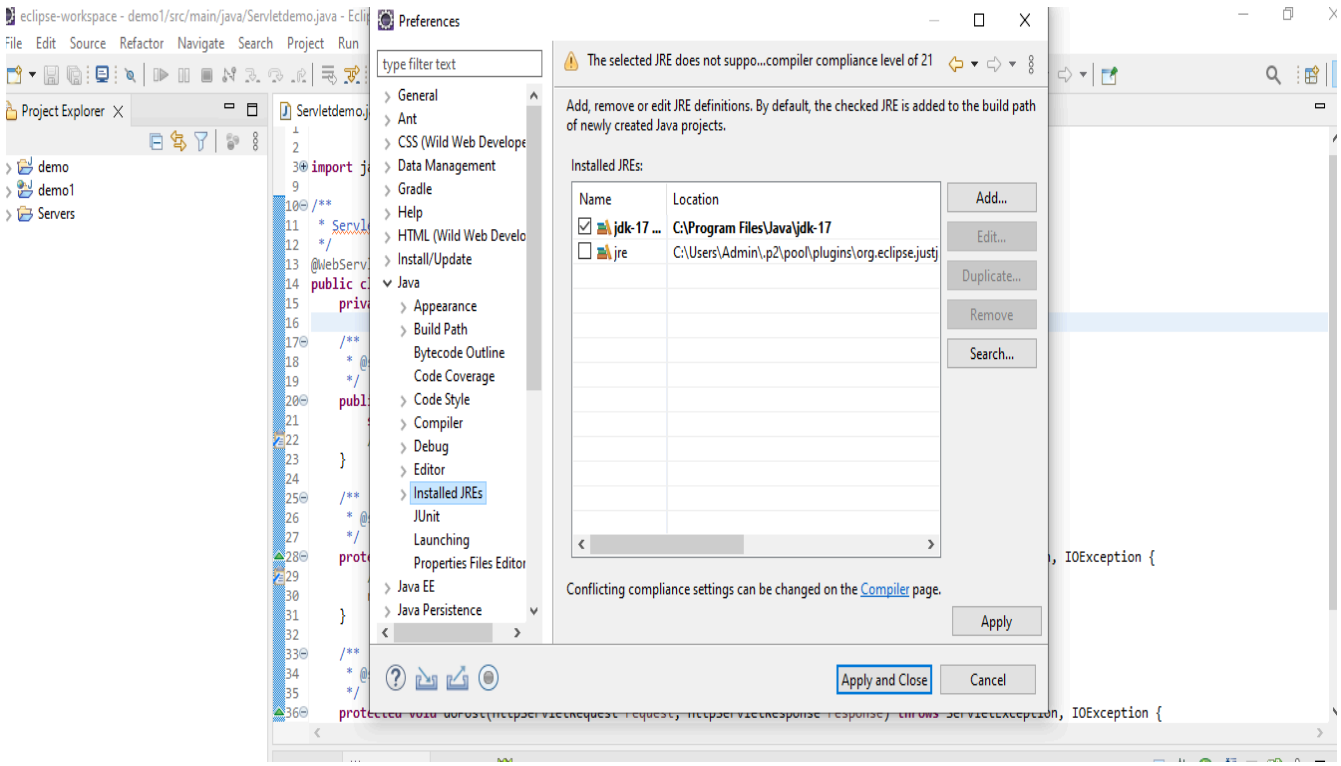
- Go to → Downloads\eclipse-jee-2023-06-R-win32-x86_64\eclipse → Double click eclipse icon.
- Check for java version. If java version is more than 17 then download and install jdk 17 and set environmental variable (JAVA_HOME ,path)



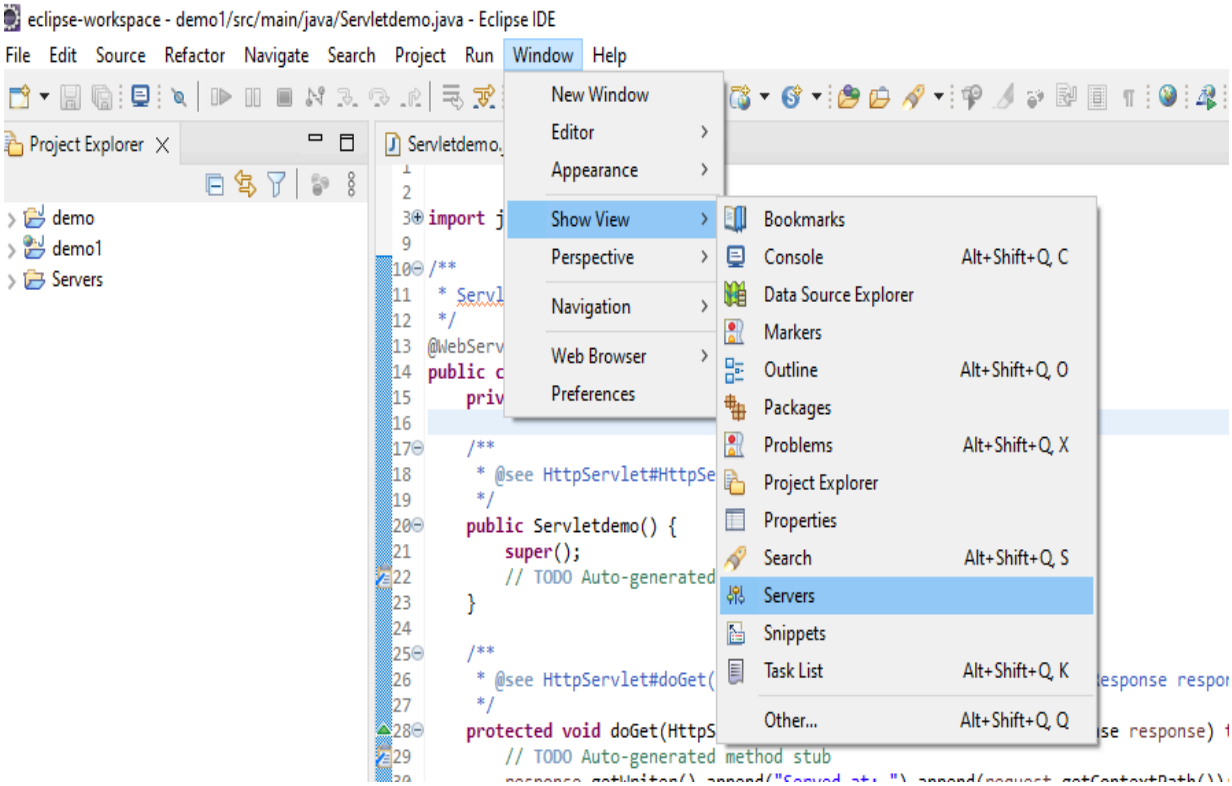


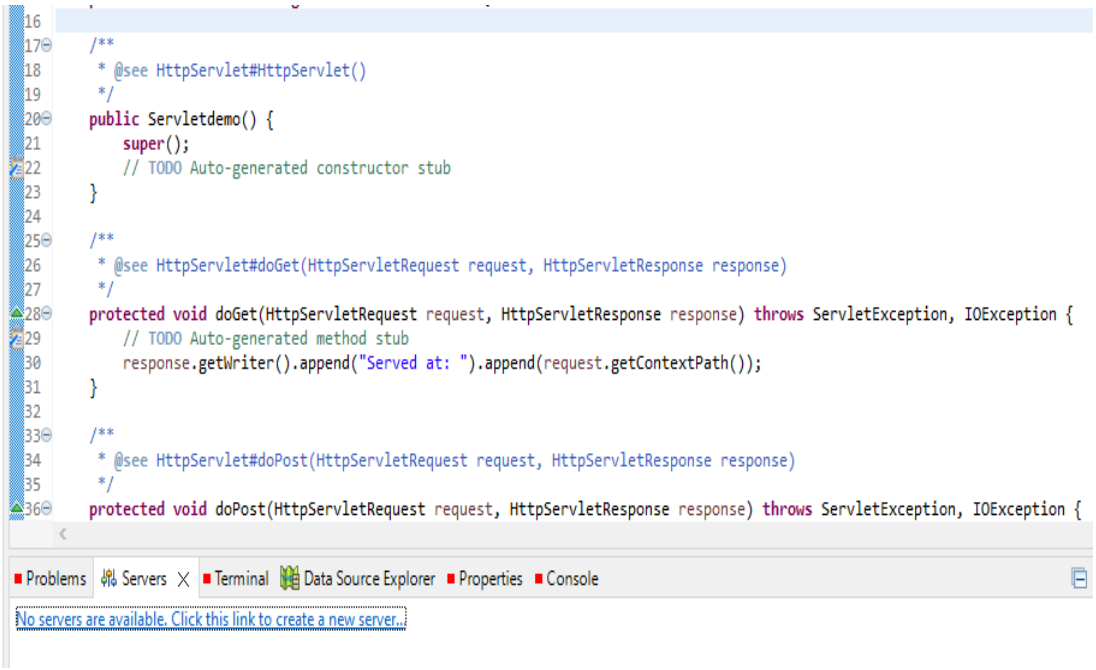
- Click on Windows → Preferences → Java → Installed jre & Check if jdk 17 path is set. If jdk 17 is not set then select folder in c:\program files\java\jdk-17.



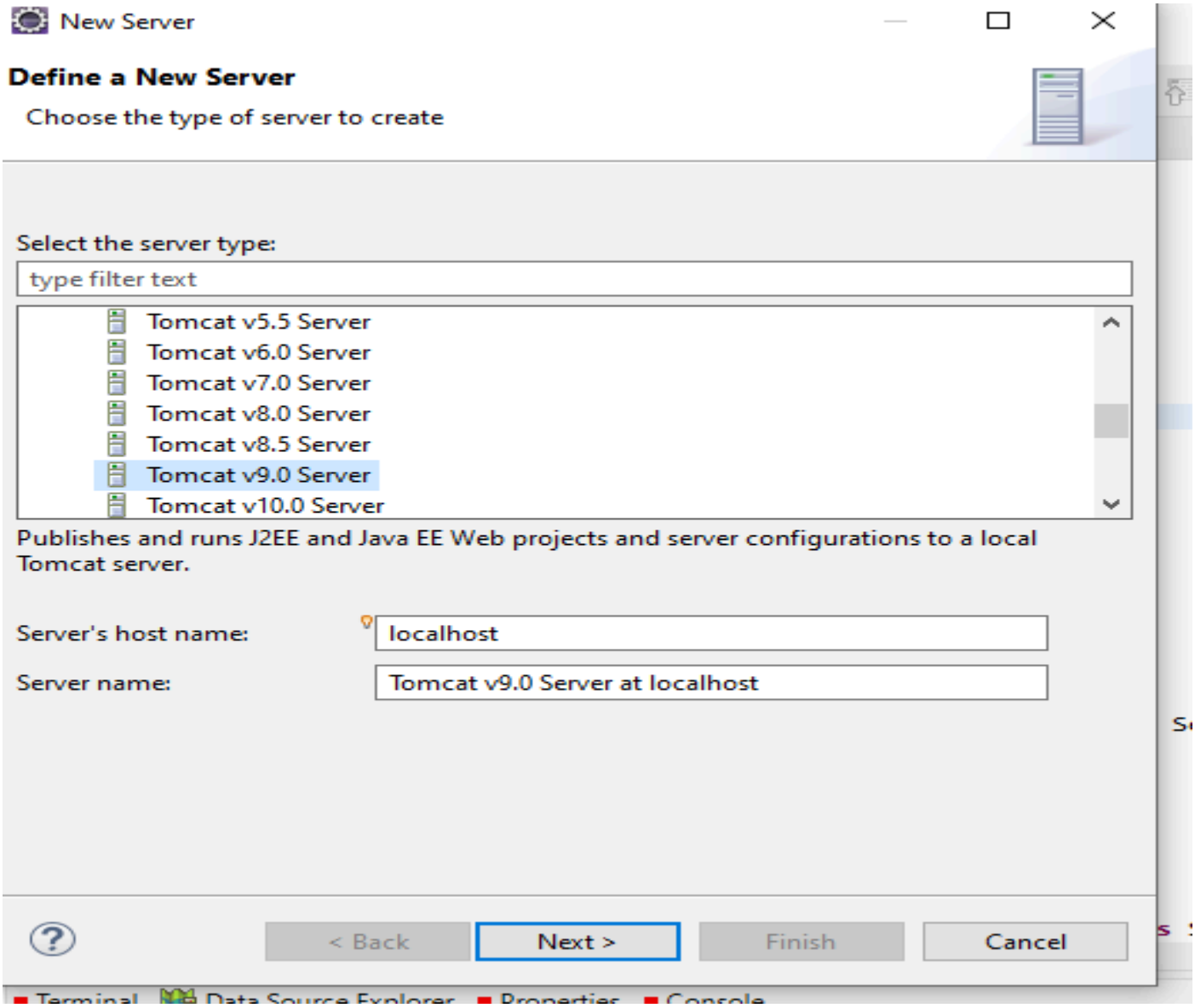


- Check for Downloads\apache-tomcat-9.0.91-windows-x64\apache-tomcat-9.0.91 if it is not available download apache server 9.0. if it is there follow the steps.





- **Click on this link.**

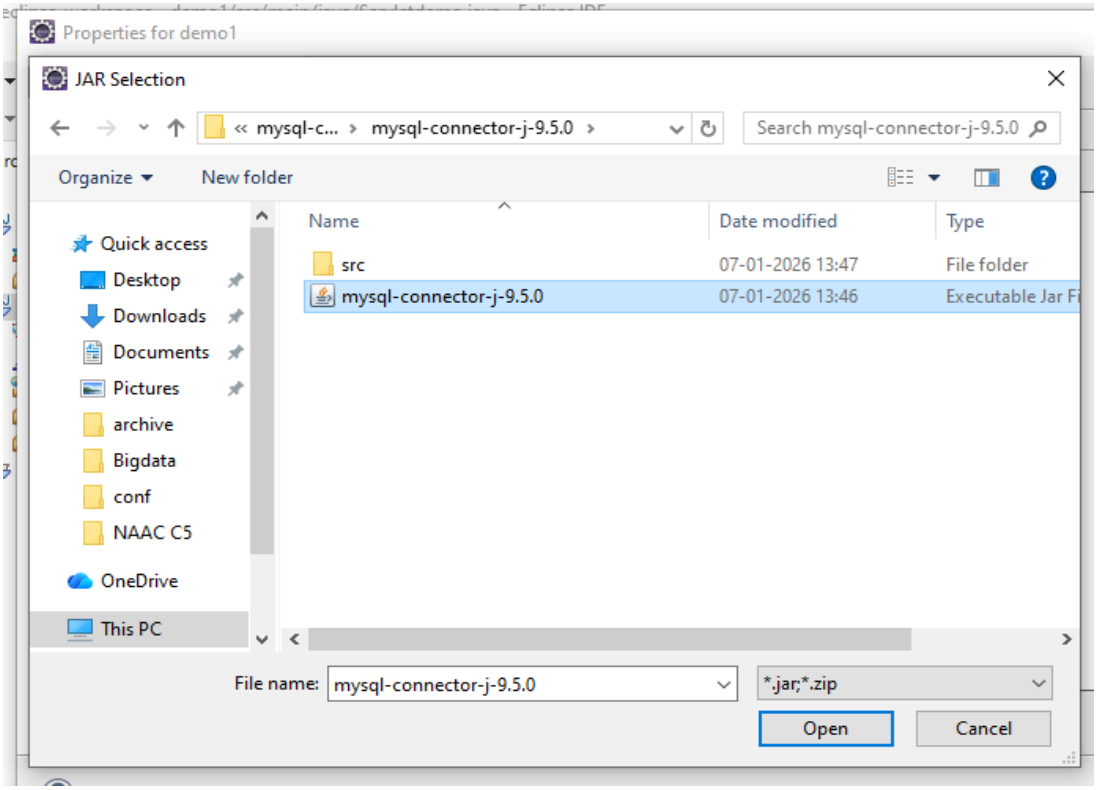


- Click on the next and tomcat directory folder till the path Downloads\apache-tomcat-9.0.91-windows-x64\apache-tomcat-9.0.91

- Create new dynamic web project:
  Go to File → new → dynamic web project.

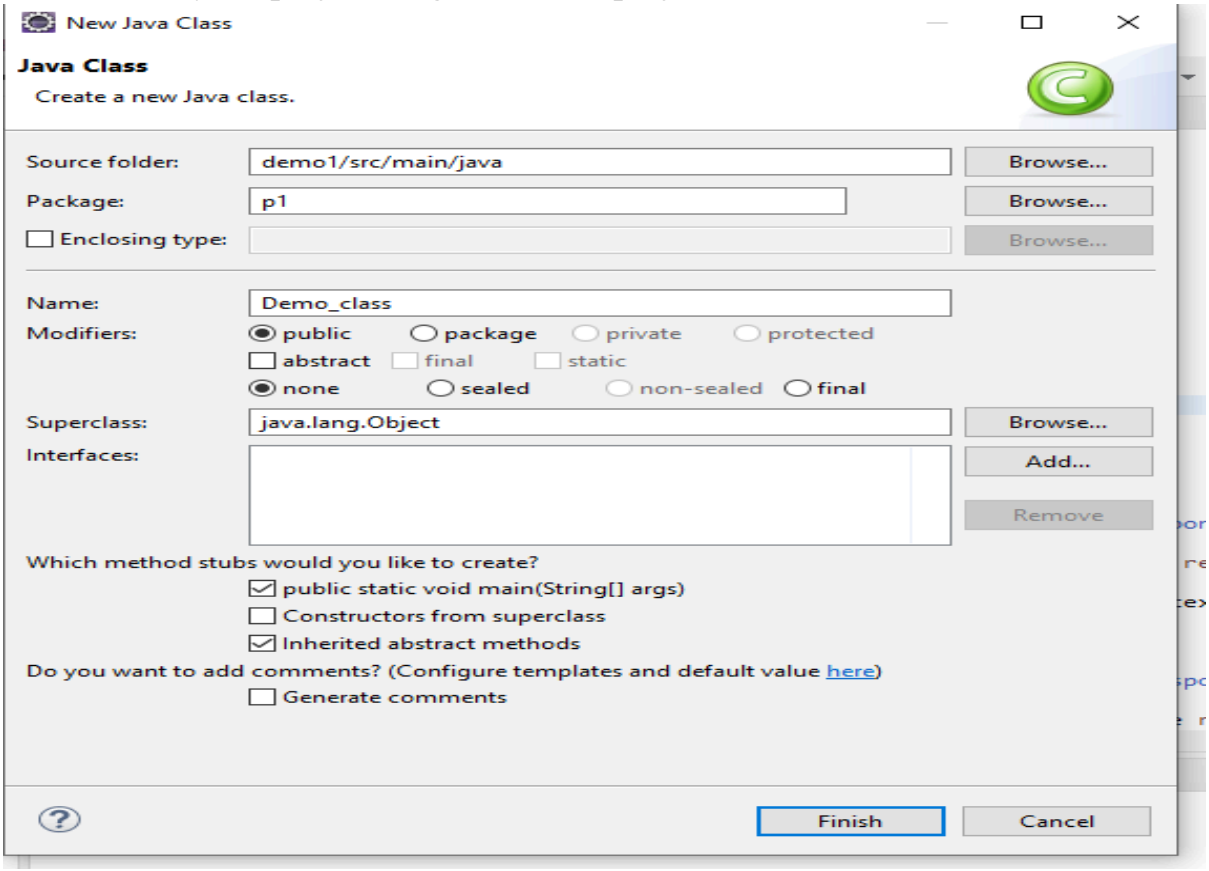
- For jdbc program download mysql j connector zip file. Extract this file after download.

- Add this jar file to your project. Right click to your project name → Build path → configure build path → Libraries → classpath → add external jar → select your extracted mysql j connector folder till jar file.



Click on open.

- Add class to your project. Right clock to project name →new → class

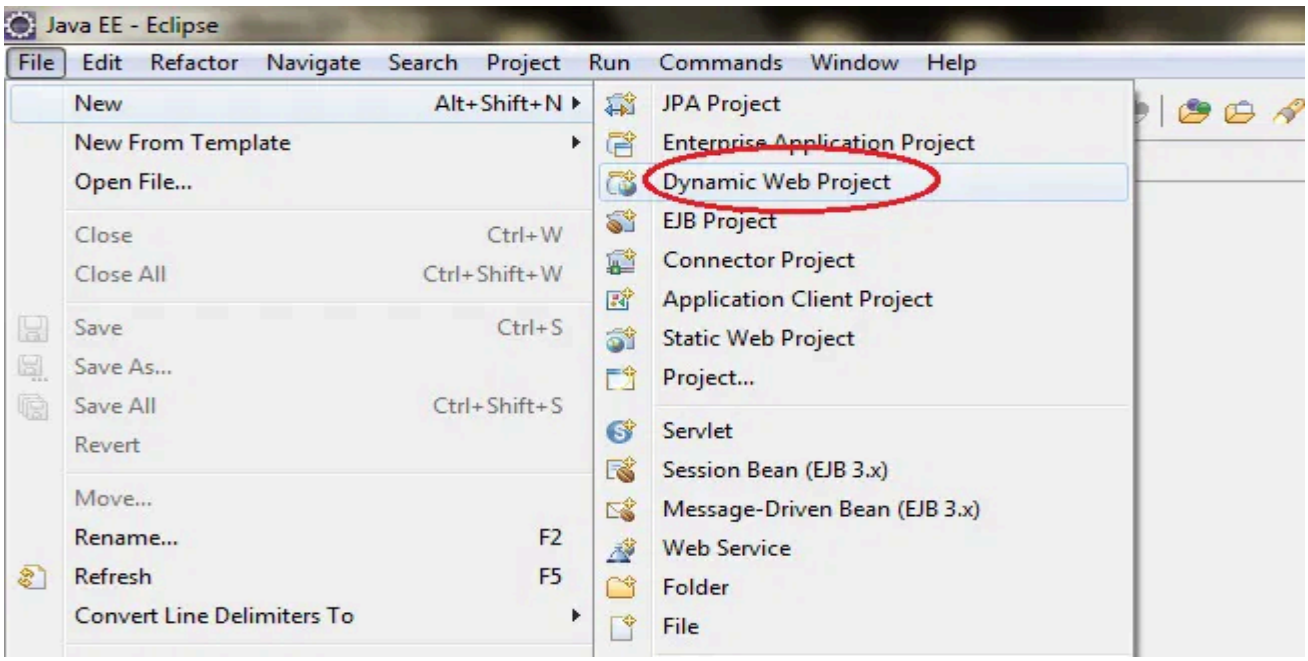


- Add name of your class , specify package name and tick public static void main().
- Write code for jdbc in class. And right click on class file and run it as java application.

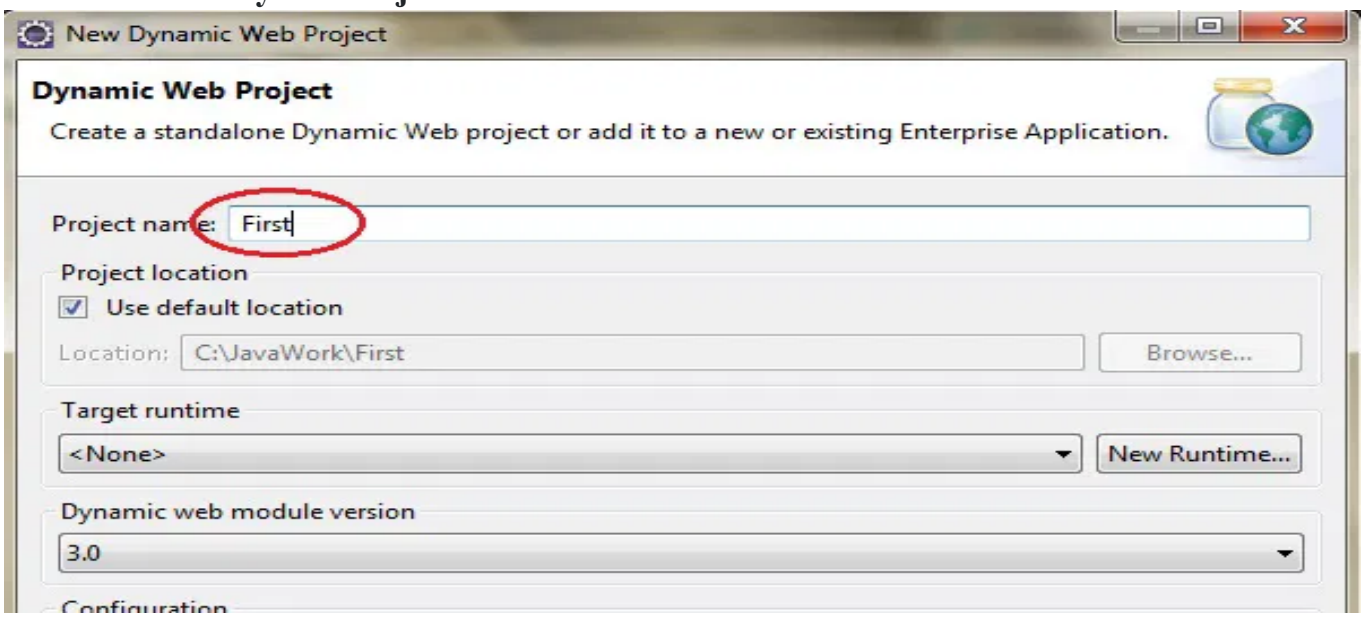
Steps to create Servlet using Eclipse IDE

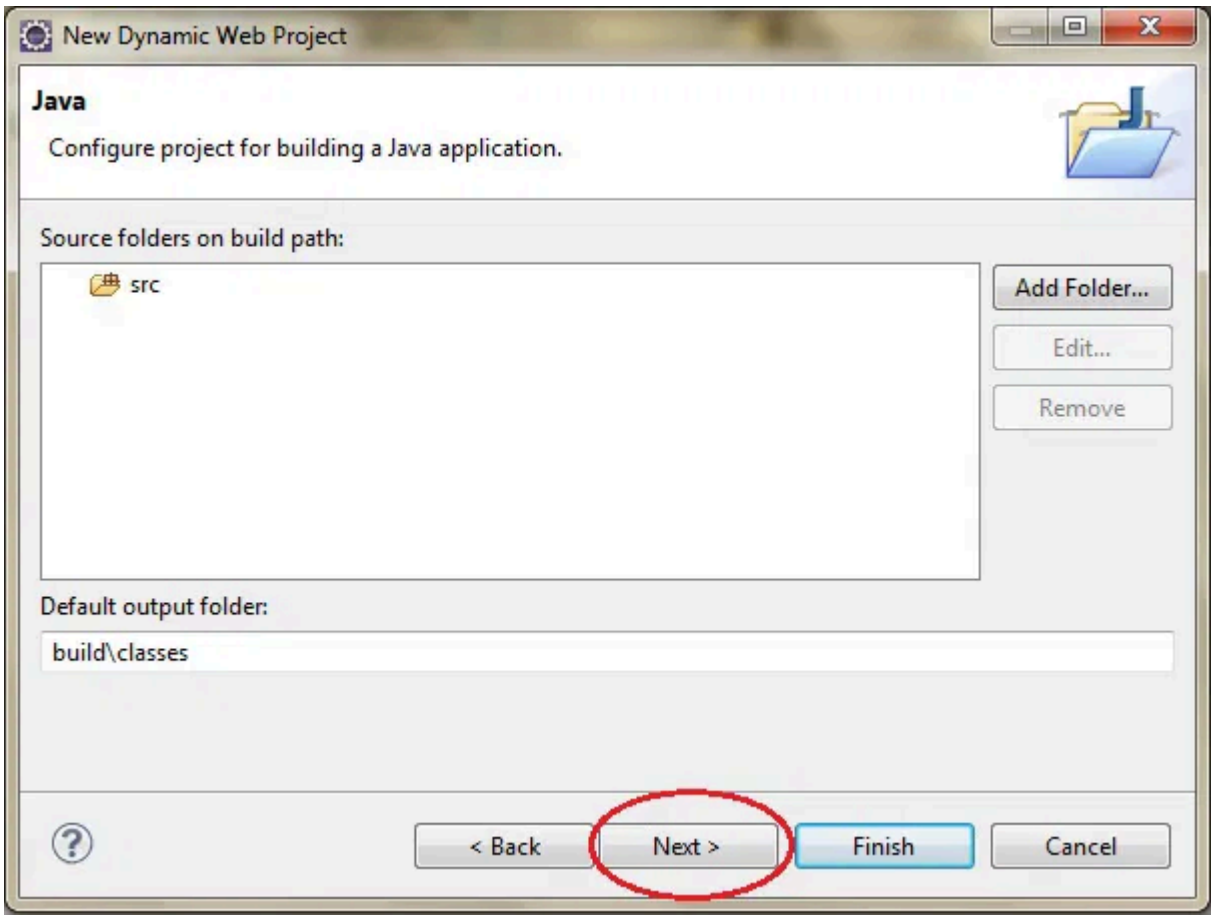
To create a Servlet application in Eclipse IDE you will need to follow the following steps:

1. Goto File -> New -> Dynamic Web Project

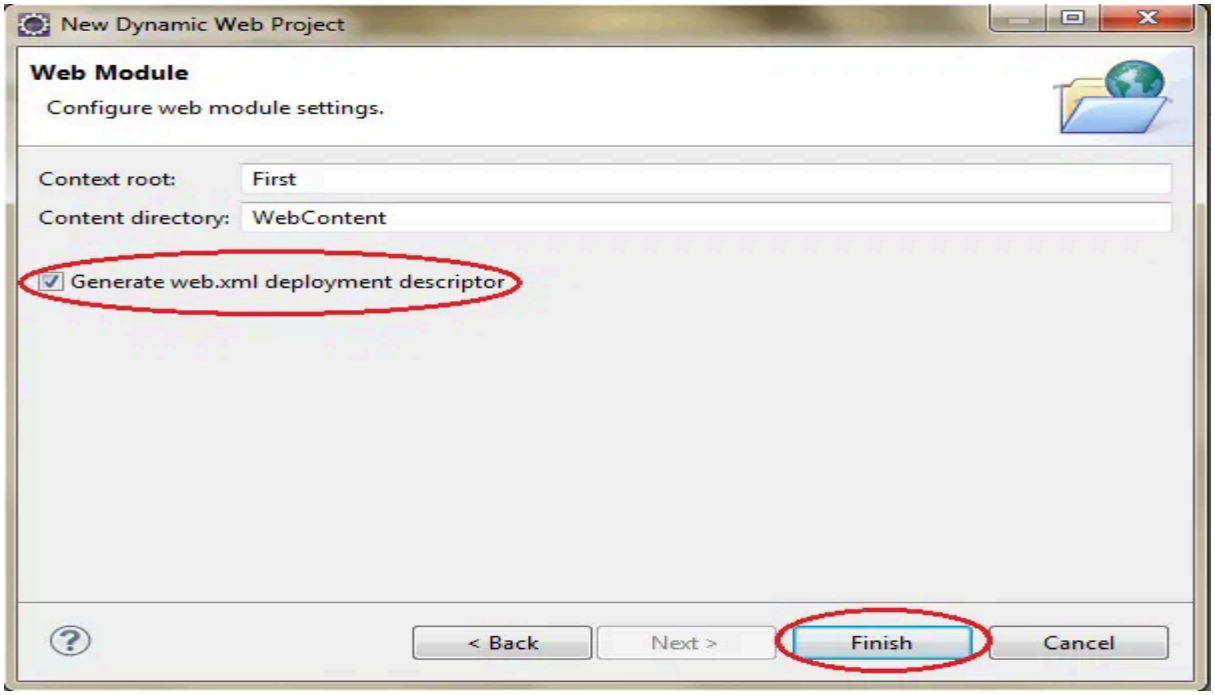


Give a Name to your Project and click Next

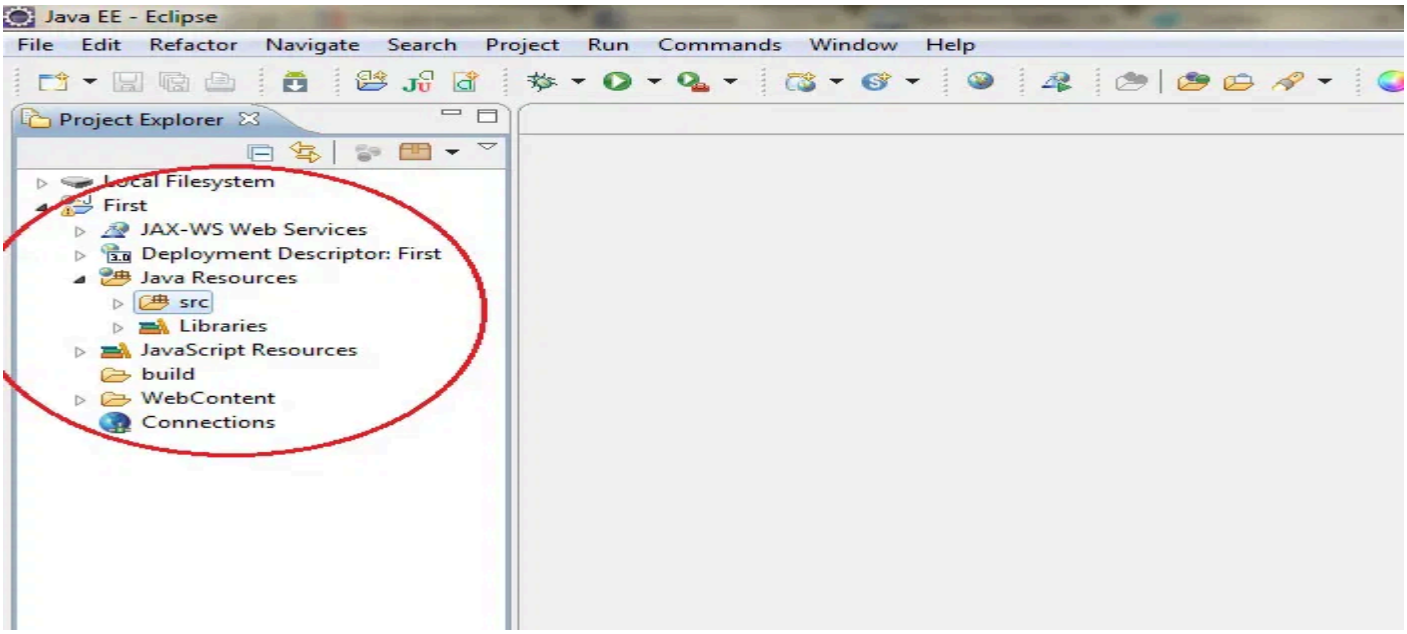




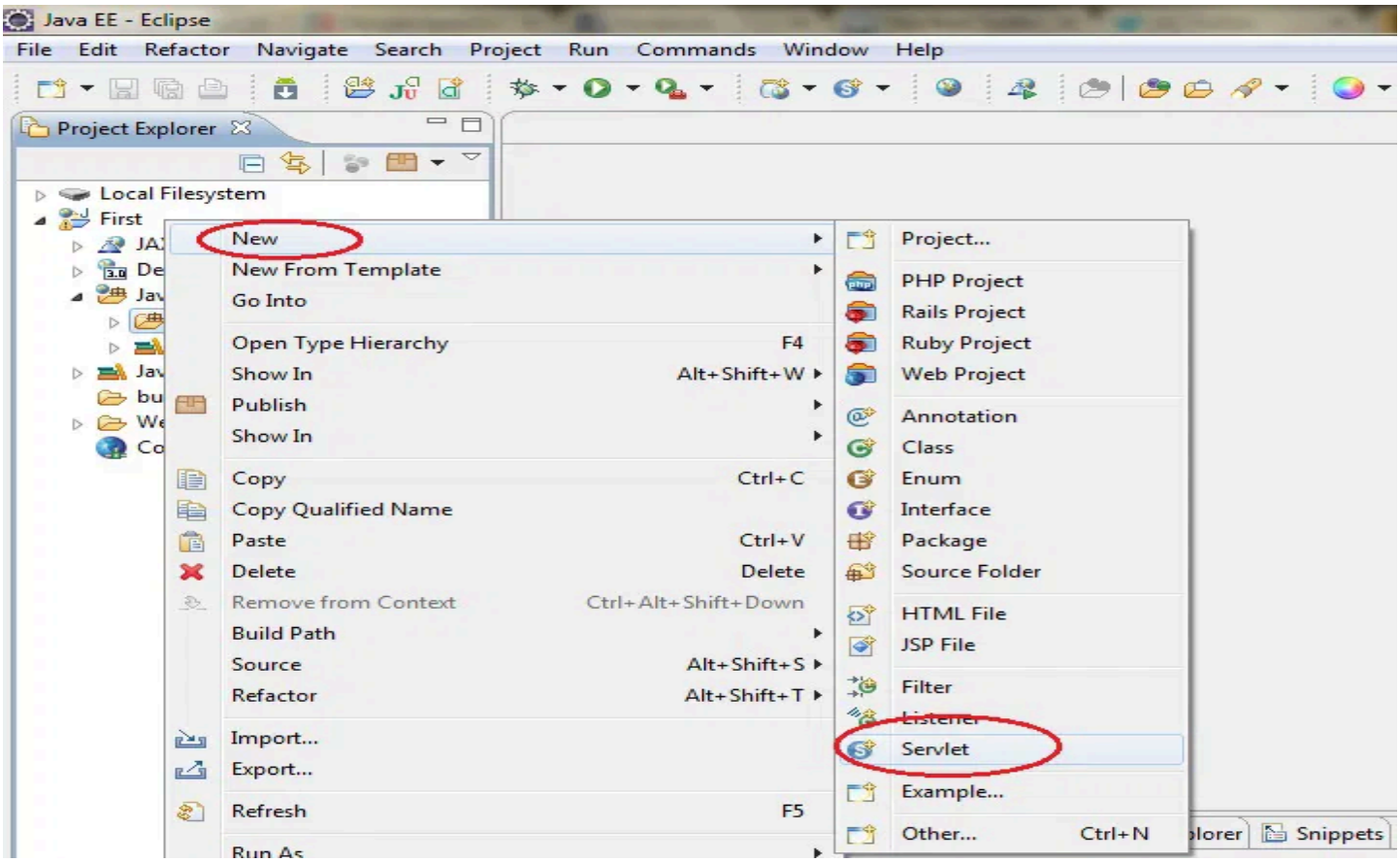
Check Generate web.xml Deployment Descriptor and click Finish



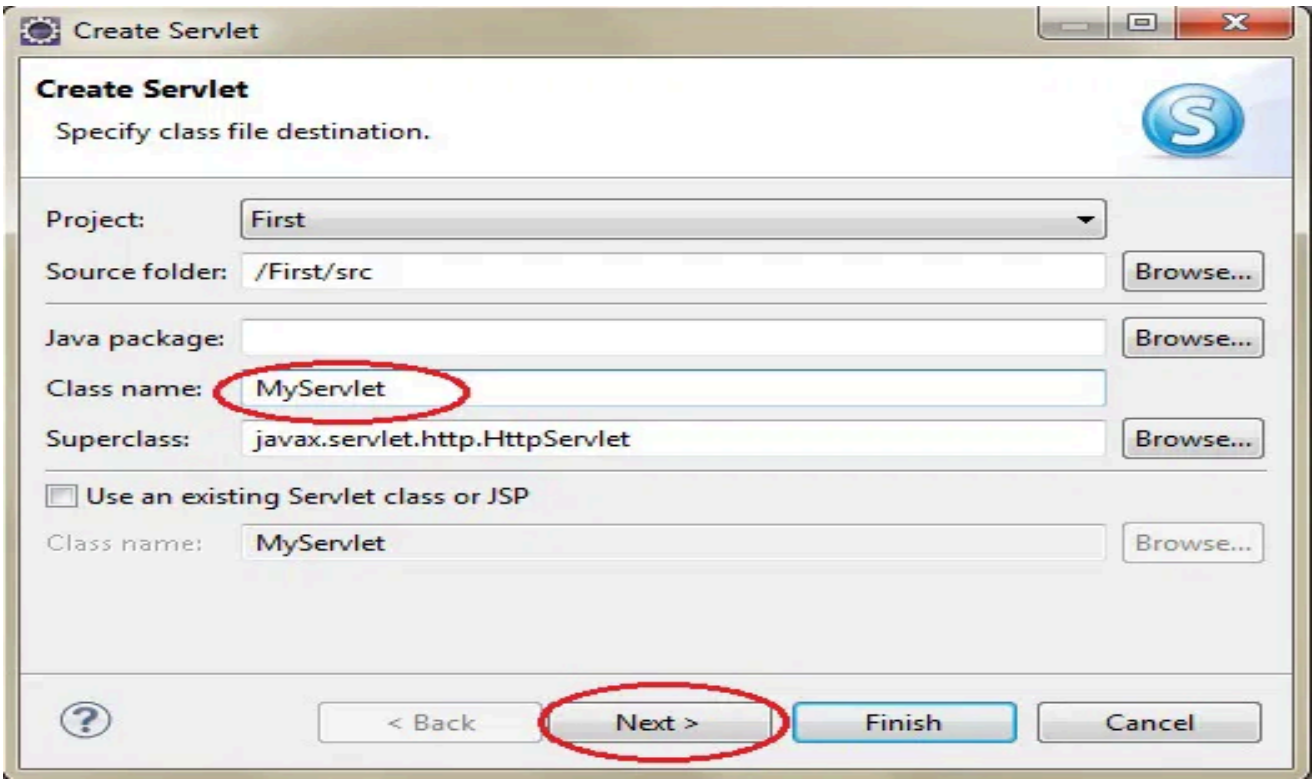
Now, the complete directory structure of your Project will be automatically created by Eclipse IDE.

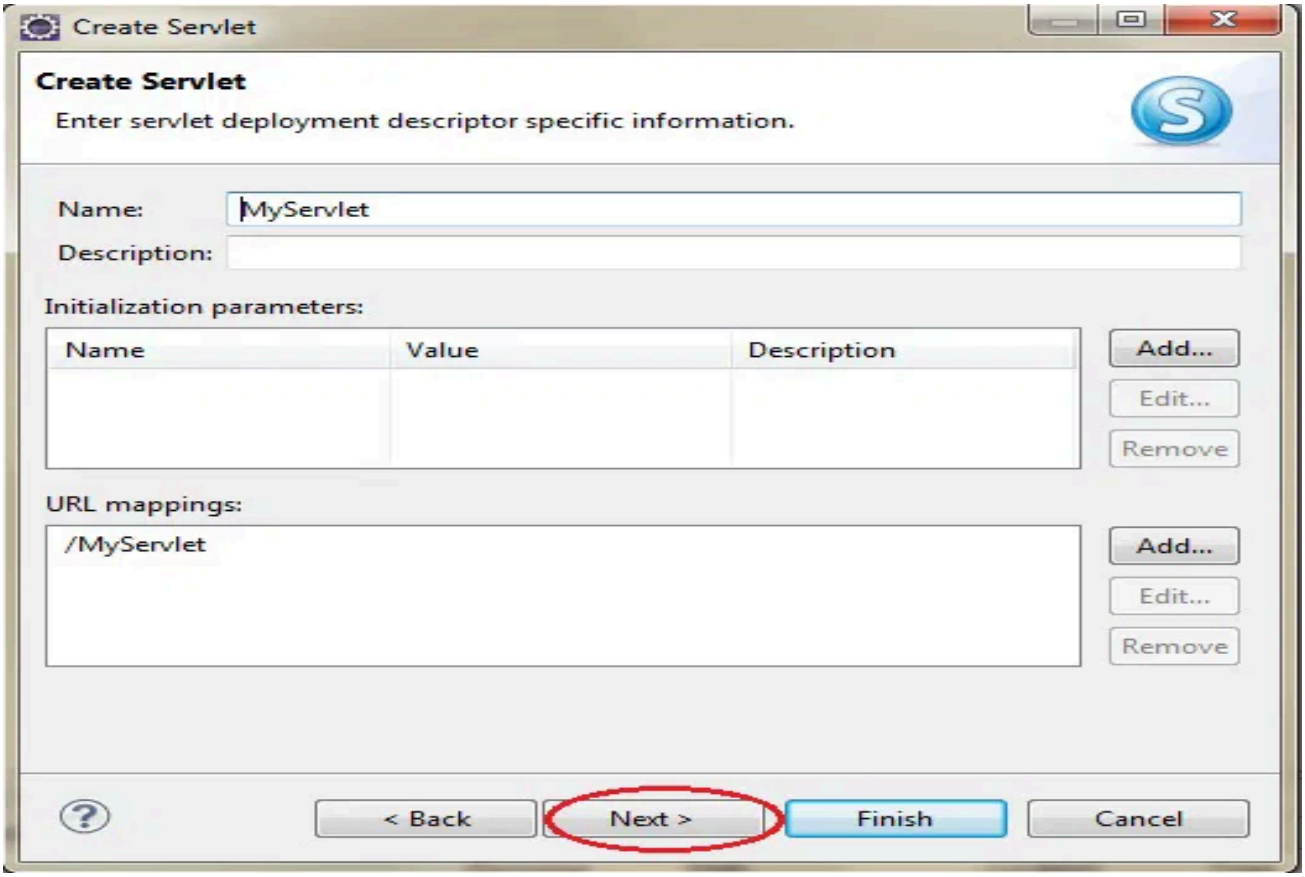


Click on First project, go to Java Resources -> src. Right click on src select New -> Servlet

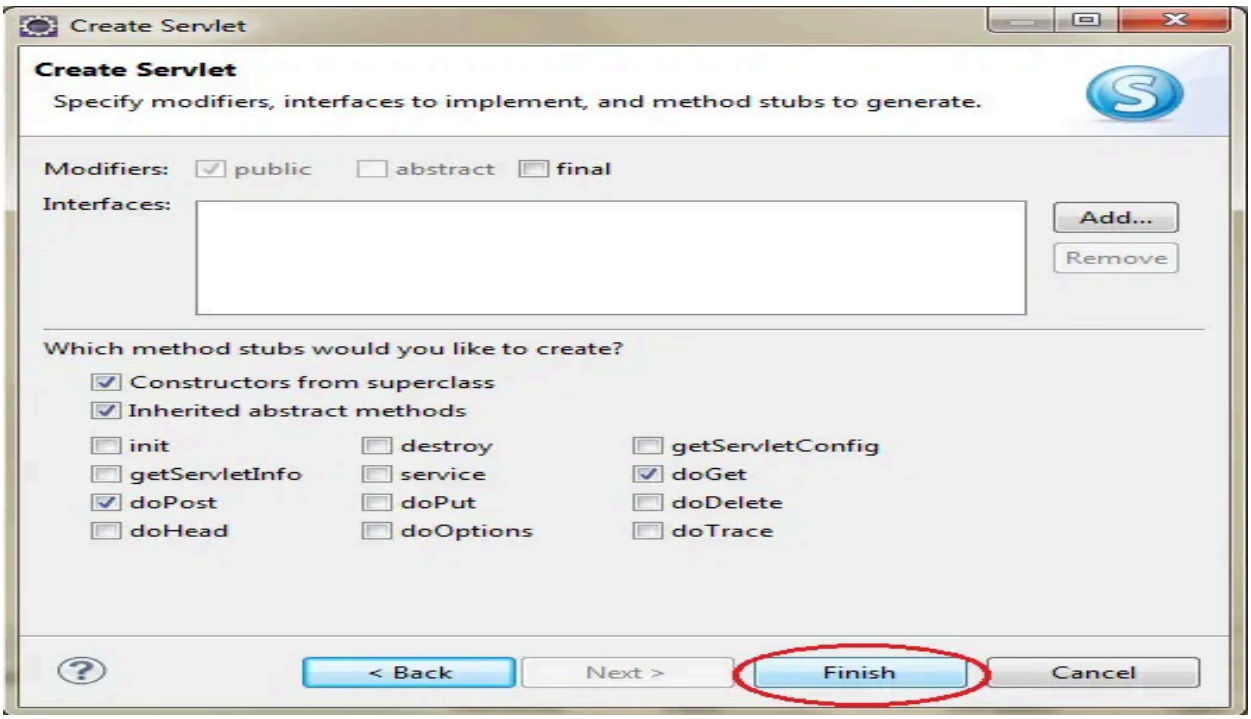


Give Servlet class name and click Next

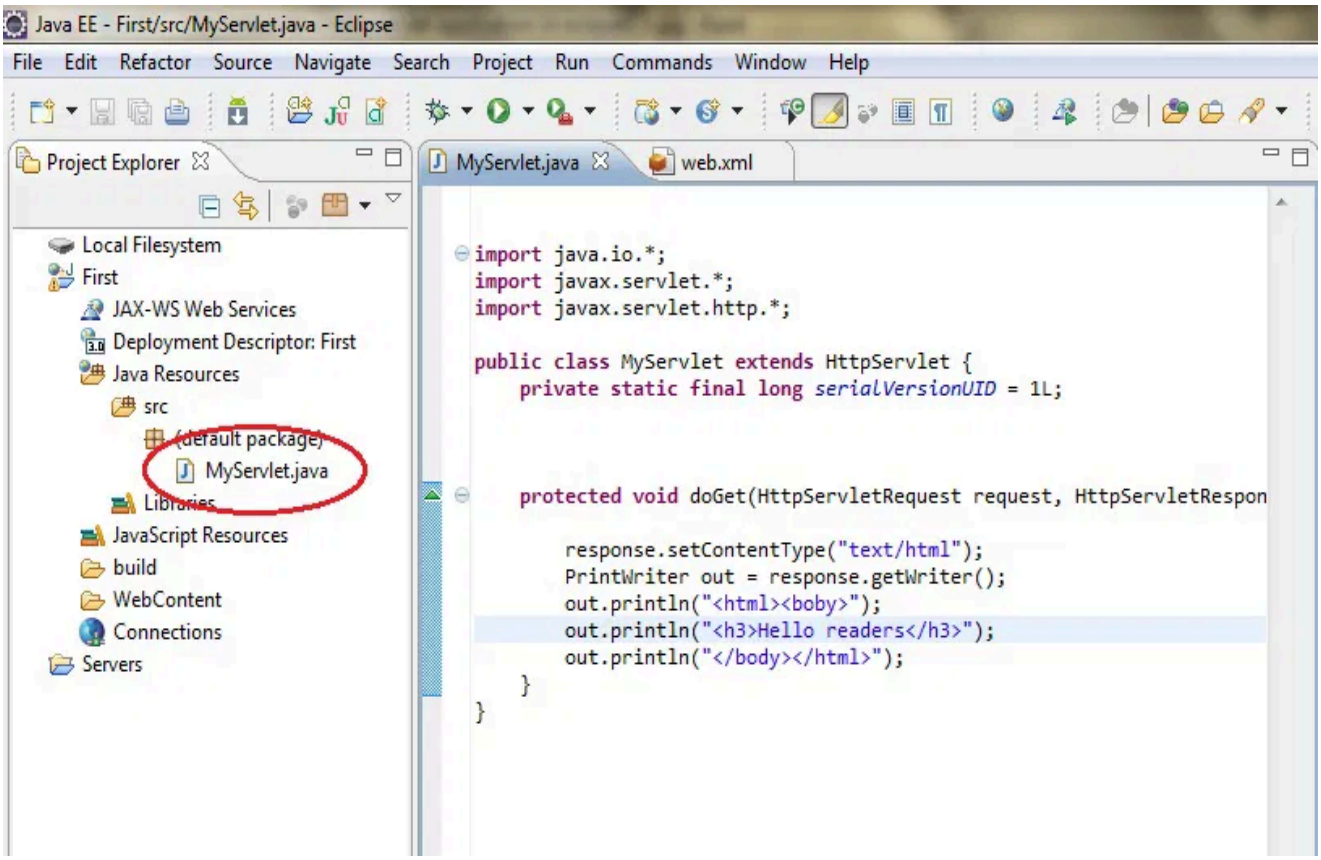




Leave everything else to default and click Finish



Now your Servlet is created, write some code inside it



Practical : 1

Aim : Design and develop standalone application / Desktop application to perform CRUD operations.

Theory :

SQL is a database language designed for the retrieval and management of data in a relational database. SQL is the standard language for database management. All the RDBMS systems like MySQL, MS Access, Oracle, Sybase, Postgres, and SQL Server use SQL as their standard database language. SQL programming language uses various commands for different operations.

Why Use SQL?

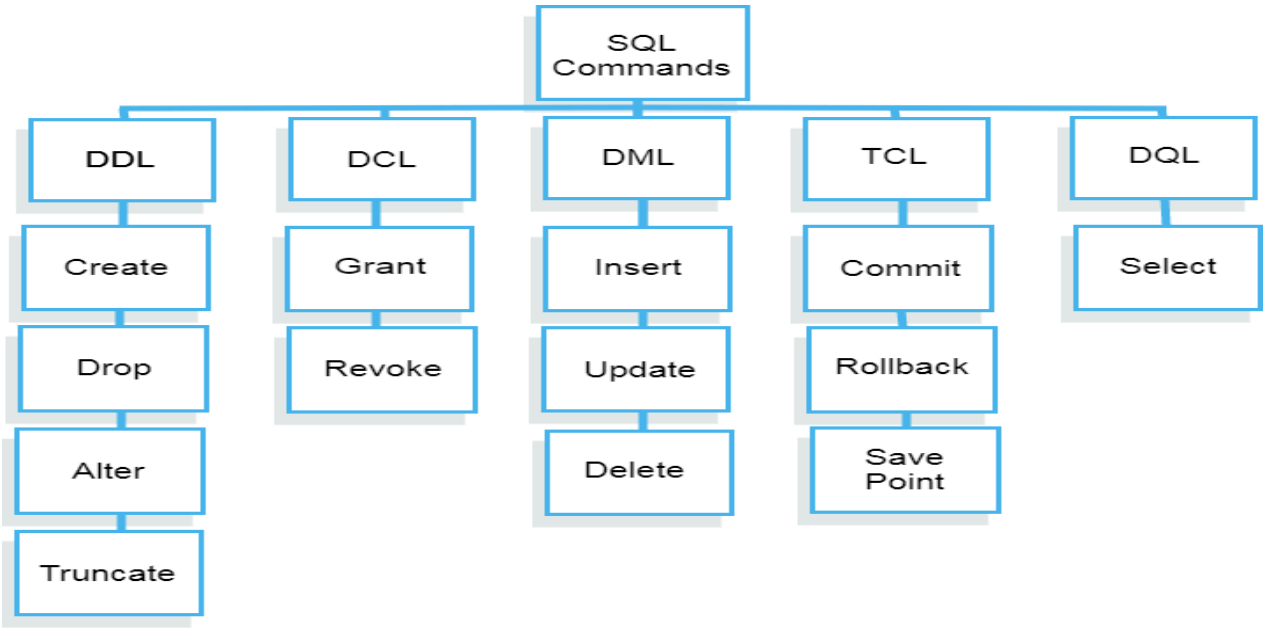
Here, are important reasons for using SQL

- It helps users to access data in the Database system.
- It helps you to describe the data.
- It allows you to define the data in a database and manipulate that specific data.
- With the help of SQL commands in java, you can create and drop databases and tables.
- SQL offers you to use the function in a database, create a view, and stored procedure.
- You can set permissions on tables, procedures, and views.

Types of SQL

Here are five types of widely used SQL queries.

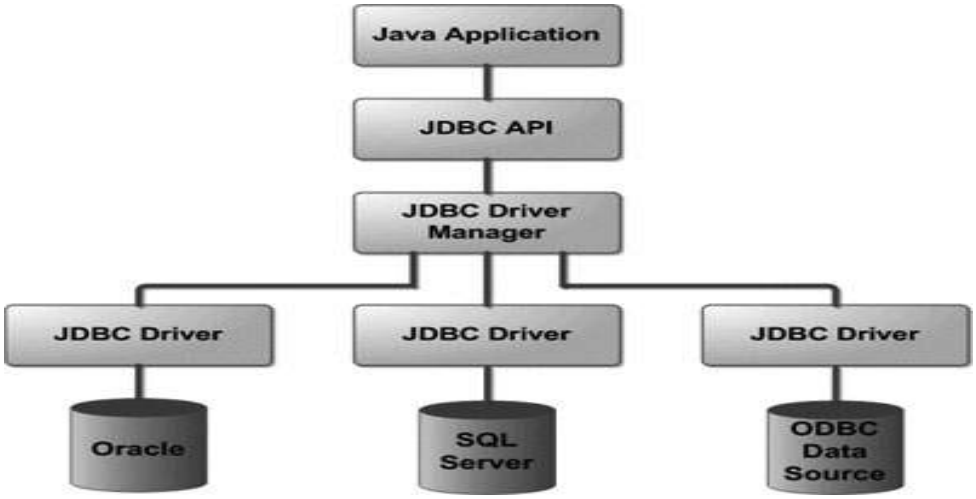
- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language(DCL)
- Transaction Control Language(TCL)
- Data Query Language (DQL)



What is JDBC?

JDBC (Java Database Connectivity) is a Java API that enables Java applications to interact with databases for: Storing data, Retrieving data , Updating data , Deleting data
It provides a standard interface to connect Java programs with different databases like MySQL, Oracle, PostgreSQL, etc.

JDBC Architecture:



JDBC Architecture Components:

- Java Application
- JDBC API
- JDBC Driver Manager
- JDBC Driver
- Database

Types of JDBC Drivers:

Type	Name	Description
Type 1	JDBC-ODBC Bridge	Uses ODBC, slow, deprecated
Type 2	Native-API Driver	Uses DB native libraries
Type 3	Network Protocol Driver	Uses middleware server
Type 4	Thin Driver	Pure Java, most commonly used

Steps to Use JDBC

Step 1: Load the Driver
Class.forName("com.mysql.cj.jdbc.Driver");

Step 2: Establish Connection
Connection con =
DriverManager.getConnection(jdbc:mysql://localhost:3306/dbname","username","password");

Step 3: Create Statement
Types:
Statement: Statement stmt = con.createStatement();

PreparedStatement:
PreparedStatement ps = con.prepareStatement("INSERT INTO student VALUES (?, ?, ?)");

CallableStatement:
CallableStatement cs = con.prepareCall("{call getStudent(?)}");

Step 4: Execute SQL Query
executeQuery() → SELECT
executeUpdate() → INSERT, UPDATE, DELETE
execute() → Any SQL

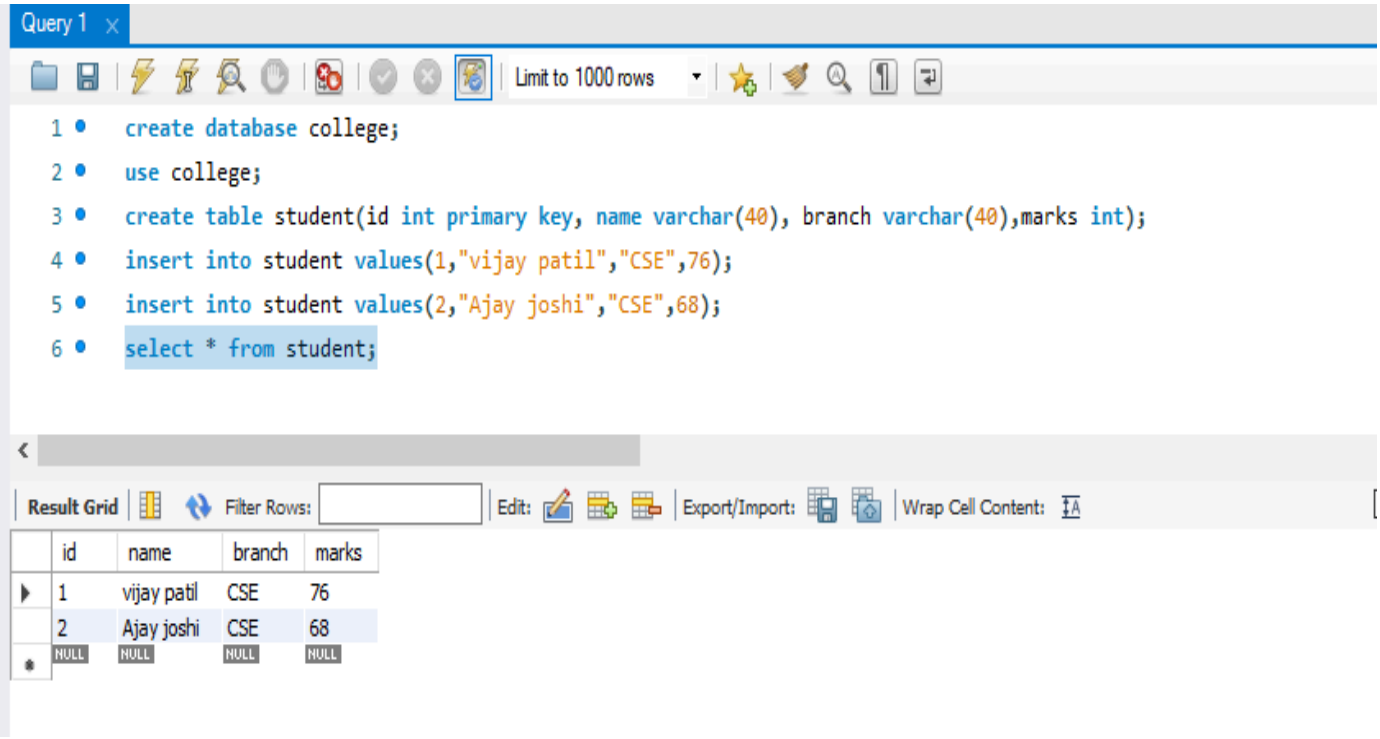
Step 5: Process ResultSet
```
while(rs.next()) {
   System.out.println(rs.getString("name"));
}
```

Step 6: Close Connection
con.close();

Conclusion:

Problem Statement:

- **Create Database with name: College**
- **Create Table Student(ID , Name , Branch , Marks)**
- **Insert Records in Student Table.**



1. **Write a Java program using JDBC to:**
   - **Connect to a MySQL database**
   - **Display a message "Database Connected Successfully"**
   - **Handle connection errors properly.**
   - **Fetch and display all records from the Student table**
   - **Insert student details (ID, Name, Branch, Marks) into a database table**
   - **Update marks of a student using Student ID**
   - **Delete a student record based on Student ID**

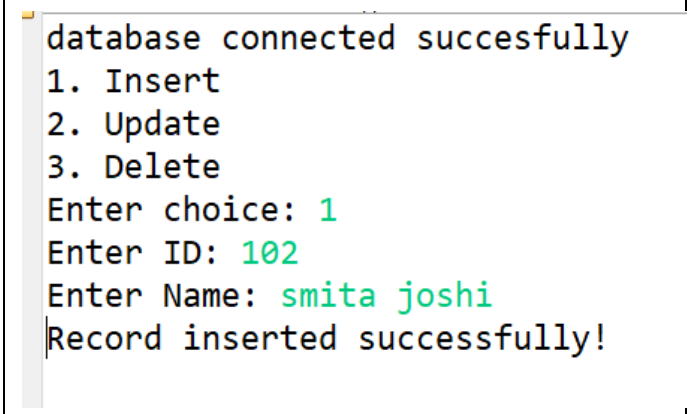
Program:

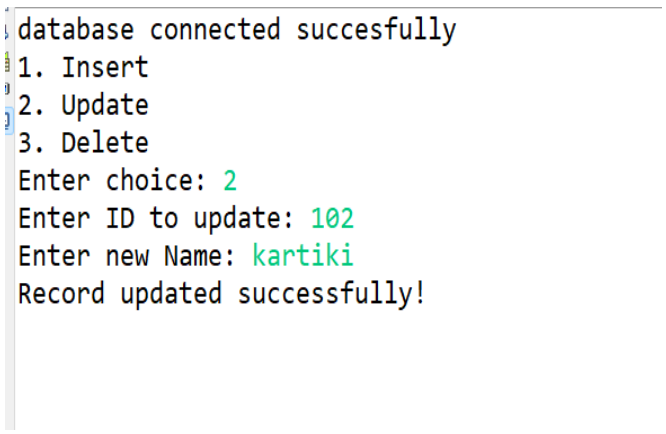
```
import java.sql.*;
import java.util.Scanner;

public class StudentJDBC {

   public static void main(String[] args) {
      String url = "jdbc:mysql://localhost:3306/college";
      String user = "root";
```

```
String password = "root";   // change as per your DB
Scanner sc = new Scanner(System.in);
try {
   // 1. Load Driver
   Class.forName("com.mysql.cj.jdbc.Driver");

   // 2. Create Connection
   Connection con = DriverManager.getConnection(url, user, password);
   System.out.println("Connected to database");

   System.out.println("1. Insert");
   System.out.println("2. Update");
   System.out.println("3. Delete");
   System.out.println("4. Display");

   System.out.print("Enter choice: ");
   int choice = sc.nextInt();

   // ---------------- INSERT ----------------
   if (choice == 1) {
      String sql = "INSERT INTO student VALUES (?, ?, ?, ?)";
      PreparedStatement ps = con.prepareStatement(sql);

      System.out.print("Enter ID: ");
      ps.setInt(1, sc.nextInt());

      System.out.print("Enter Name: ");
      ps.setString(2, sc.next());

      System.out.print("Enter Branch: ");
      ps.setString(3, sc.next());

      System.out.print("Enter Marks: ");
      ps.setInt(4, sc.nextInt());

      int rows = ps.executeUpdate();
      System.out.println(rows + " record inserted");
   }

   // ---------------- UPDATE ----------------
```

```
        else if (choice == 2) {
            String sql = "UPDATE student SET marks=? WHERE id=?";
            PreparedStatement ps = con.prepareStatement(sql);

            System.out.print("Enter new marks: ");
            ps.setInt(1, sc.nextInt());

            System.out.print("Enter student ID: ");
            ps.setInt(2, sc.nextInt());

            int rows = ps.executeUpdate();
            System.out.println(rows + " record updated");
        }
        // ---------------- DELETE ----------------
        else if (choice == 3) {
            String sql = "DELETE FROM student WHERE id=?";
            PreparedStatement ps = con.prepareStatement(sql);

            System.out.print("Enter student ID to delete: ");
            ps.setInt(1, sc.nextInt());

            int rows = ps.executeUpdate();
            System.out.println(rows + " record deleted");
        }
// ---------------- DISPLAY RECORDS ----------------
      else if (choice == 4) {
            String sql = "select * from student";
           Statement st  = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
                 // 6. Display Records
        System.out.println("ID\tName\tBranch\tMarks");
        System.out.println("-------------------------------");
        while (rs.next()) {
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String branch = rs.getString("branch");
            int marks = rs.getInt("marks");
            System.out.println(id + "\t" + name + "\t" + branch + "\t" + marks);
        }
        // 7. Close Connection
```

```
        con.close();
        }
        else {
           System.out.println("Invalid choice");
        }
        // Close connection
        con.close();

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





Output:

Output:

****Add screenshots of your executed program with all options.**
Specify Rollno. And Name the right corner of each printout page.

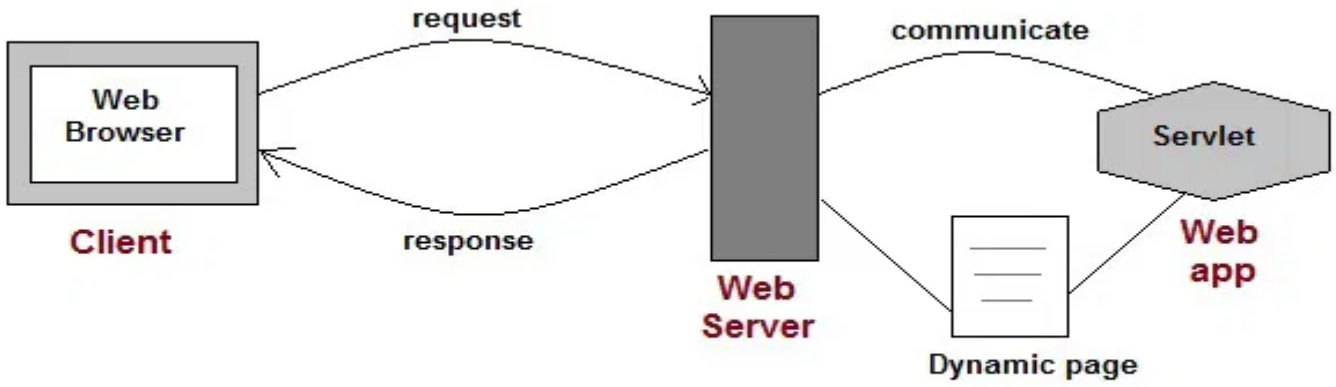
Practical : 2

Aim : Write a program to create a simple servlet for 1. Demonstration of Servlet Life Cycle 2. Form processing (Student Information) 3. Printing request header information.

Theory :

Java Servlets are programs that run on a Web or Application server and act as a middle layer between a requests coming from a Web browser or other HTTP client and databases or applications on the HTTP server.

Servlets Architecture



Servlets Tasks

Servlets perform the following major tasks −

- Read the explicit data sent by the clients (browsers). This includes an HTML form on a Web page.
- Read the implicit HTTP request data sent by the clients (browsers).
- Process the data and generate the results. This process may require talking to a database, executing an RMI or CORBA call, invoking a Web service, or computing the response directly.
- Send the explicit data (i.e., the document) to the clients (browsers). This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), Excel, etc.
- Send the implicit HTTP response to the clients (browsers). This includes telling the browsers or other clients what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other such tasks.

Servlet API consists of two important packages that encapsulates all the important classes and interface, namely :

- javax.servlet
- javax.servlet.http

Some Important Classes and Interface of javax.servlet.http

CLASSES and INTERFACES	
HttpServlet	**HttpServletRequest**
HttpServletResponse	**HttpSessionAttributeListener**
HttpSession	**HttpSessionListener**
Cookie	**HttpSessionEvent**

A servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet.

- The servlet is initialized by calling the **init()** method.
- The servlet calls **service()** method to process a client's request.
- The servlet is terminated by calling the **destroy()** method.
- Finally, servlet is garbage collected by the garbage collector of the JVM.

Problem Statement:

Write a program to create a simple servlet for

1. Demonstration of Servlet Life Cycle
2. Printing request header information
3. Form processing (Student Information)

1. Demonstration of Servlet Life Cycle

```
package p1;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet("/Servlet_lifecycle")

public class Servlet_lifecycle extends HttpServlet
 {
        private static final long serialVersionUID = 1L;

     public Servlet_lifecycle()
      {
         super();
      }

    public void init(ServletConfig config) throws ServletException
    {
              System.out.println("init() method is initialized");
    }

    public void destroy()
    {
              // TODO Auto-generated method stub
    }

   protected void service(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
    {
              System.out.println("service() method is initialized");
```

```
            doGet(request,response);
    }

   protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
    {
        response.setContentType("text/html");
         PrintWriter out = response.getWriter();
         out.println("<html><body>");
        out.println("<center>");
        out.println("<font color='blue' size=24px>");
        out.println("<p>doGet() method called</p>");
        out.println("</font>");
       out.println("</center>");
       out.println("</body></html>");

   }
}
```

Screenshots:


```
1 package p1;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import javax.servlet.*;
6 import javax.servlet.annotation.WebServlet;
7 import javax.servlet.http.*;
8
9 @WebServlet("/Servlet_lifecycle")
10 public class Servlet_lifecycle extends HttpServlet {
11     private static final long serialVersionUID = 1L;
12
```

```
INFO: Reloading Context with name [/Practical2] has started
Jan 10, 2026 3:05:04 PM org.apache.catalina.core.StandardContext reload
INFO: Reloading Context with name [/Practical2] is completed
init() method is initialized
service() method is initialized
service() method is initialized
Jan 10, 2026 3:15:45 PM org.apache.catalina.core.StandardContext reload
INFO: Reloading Context with name [/Practical2] has started
Jan 10, 2026 3:15:45 PM org.apache.catalina.core.StandardContext reload
INFO: Reloading Context with name [/Practical2] is completed
```

2. Printing request header information

```
package p1;
import java.io.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.*;
import java.util.Enumeration;

@WebServlet("/Header_info")
public class Header_info extends HttpServlet
{
        private static final long serialVersionUID = 1L;
       public Header_info()
       {
            super();
       }
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
      {
            response.setContentType("text/html");
             PrintWriter out = response.getWriter();
             out.println("<h2>HTTP Header Details</h2>");
             Enumeration<String> headers = request.getHeaderNames();
              while (headers.hasMoreElements())
               {
                     String name = headers.nextElement();
                     out.println("<b>" + name + " :</b> " +
                     request.getHeader(name) + "<br>");
                }

        }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
      {
              doGet(request, response);
        }
}
```

Screenshot:

3.Form processing (Student Information)

student.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h2>Student Registration</h2>
  <form action="Student_data" method="post">
     Roll No: <input type="text" name="roll"><br><br>
     Name: <input type="text" name="name"><br><br>
    Branch: <input type="text" name="branch"><br><br>
     marks: <input type="text" name="marks"><br><br>
     <input type="submit" value="Submit">
  </form>
</body>
</html>
```

Student_Data.java

```
package p1;
import java.io.*;
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("/Student_data")
public class Student_data extends HttpServlet {
        private static final long serialVersionUID = 1L;

  public Student_data() {
    super();
    // TODO Auto-generated constructor stub
  }

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                // TODO Auto-generated method stub

        }

protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
 {
             response.setContentType("text/html");
              PrintWriter out = response.getWriter();
              // Reading form data
              String roll = request.getParameter("roll");
              String name = request.getParameter("name");
              String branch = request.getParameter("branch");
              String marks = request.getParameter("marks");

              out.println("<html><body>");
              out.println("<h2>Student Details</h2>");
              out.println("<table border='1'>");
              out.println("<tr><td>Roll No</td><td>" + roll + "</td></tr>");
```

```
            out.println("<tr><td>Name</td><td>" + name + "</td></tr>");
            out.println("<tr><td>Course</td><td>" + branch + "</td></tr>");
            out.println("<tr><td>Age</td><td>" + marks + "</td></tr>");
            out.println("</table>");
            out.println("</body></html>");

        }
}
```