



UNIVERSITI MALAYSIA TERENGGANU

**CSM3023 WEB-BASED APPLICATION DEVELOPMENT
(K1)**

**BACHELOR OF COMPUTER SCIENCE (MOBILE
COMPUTING) WITH HONORS**

**LAB 1 - INTRODUCTION TO SERVLET, JSP AND
MYSQL DATABASE**

SEMESTER II 2023/2024

Prepared for:

DR. MOHAMAD NOR HASSAN

Prepared by:

**NUR FADHILAH BINTI MOHD RAHMAT
(S67172)**



Week 1

Introduction to Servlet, JSP and MySQL Database

Web Based Application
Development



Lecturers

PUSAT PENGAJIAN INFORMATIK DAN MATEMATIK GUNAAN
(PPIMG), UNIVERSITI MALAYSIA TERENGGANU (UMT)

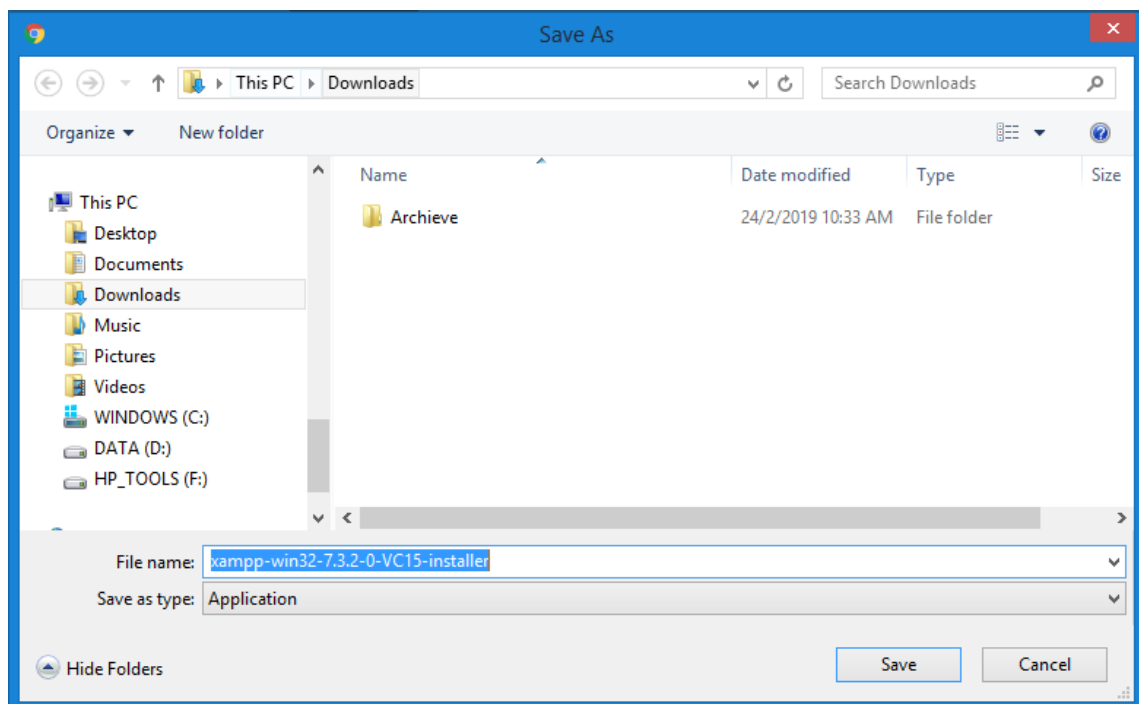
Task 1: Apache Tomcat and MySQL Installation Using XAMPP

Objective : Installation of Apache Tomcat and MySQL
Problem : To install Apache Tomcat and MySQL
Description
Estimated time : 25 minutes

1. Go to the browser and type URL
<http://www.apachefriends.org/en/xampp.html>
2. Click to XAMPP for Windows.



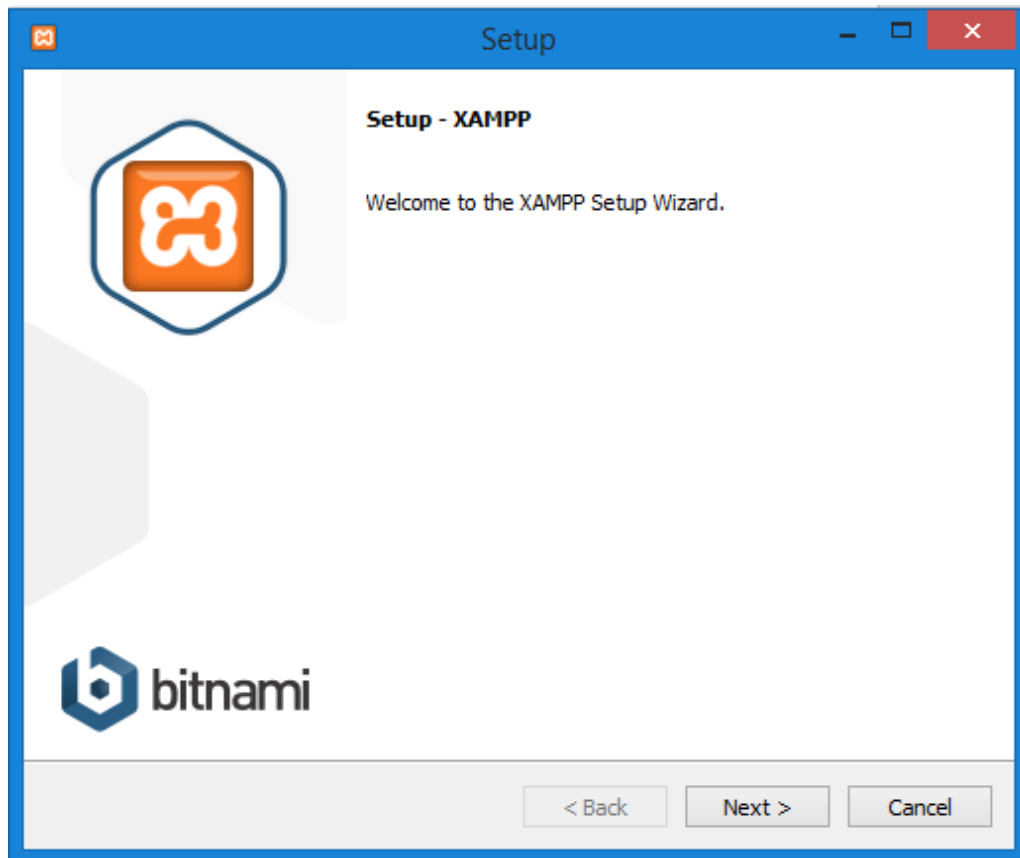
3. Save to a specific directory – for example, the Downloads folder.



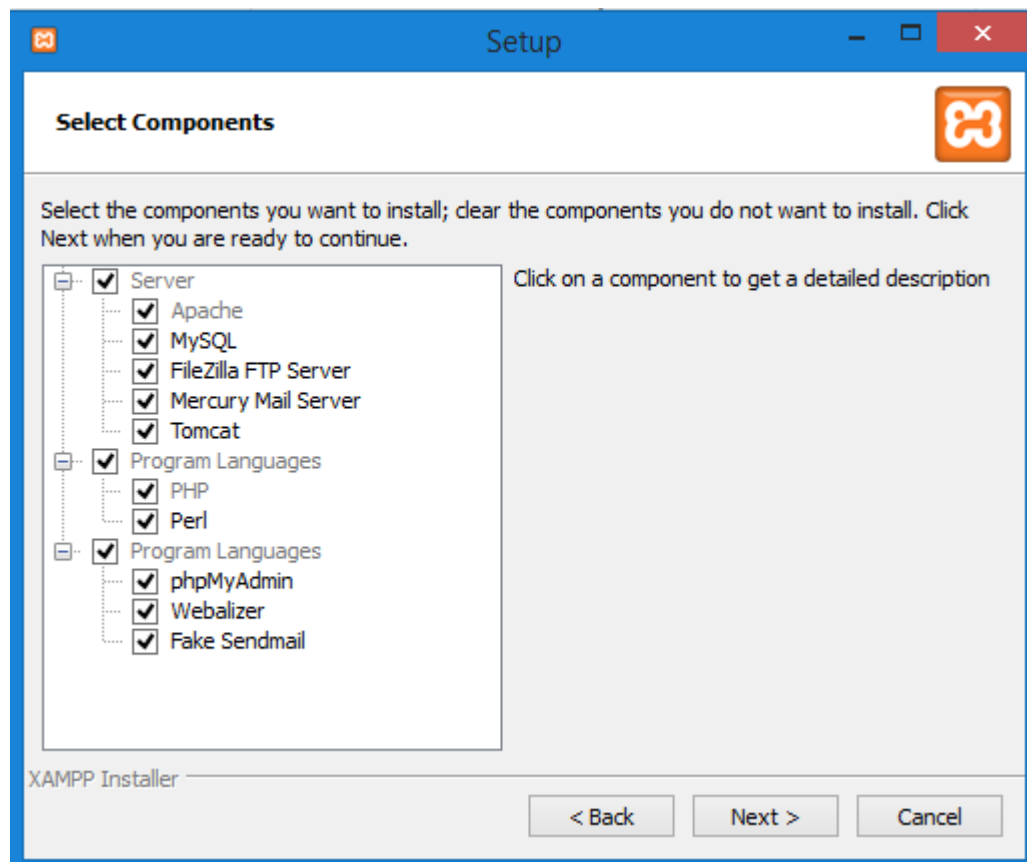
5. Run file xampp-win32-[X.X.X-VCXX]-installer.exe

Note: [X.X.X-VCXX] refers to the current version of the installer. It might differ with the version you see on the XAMPP website.

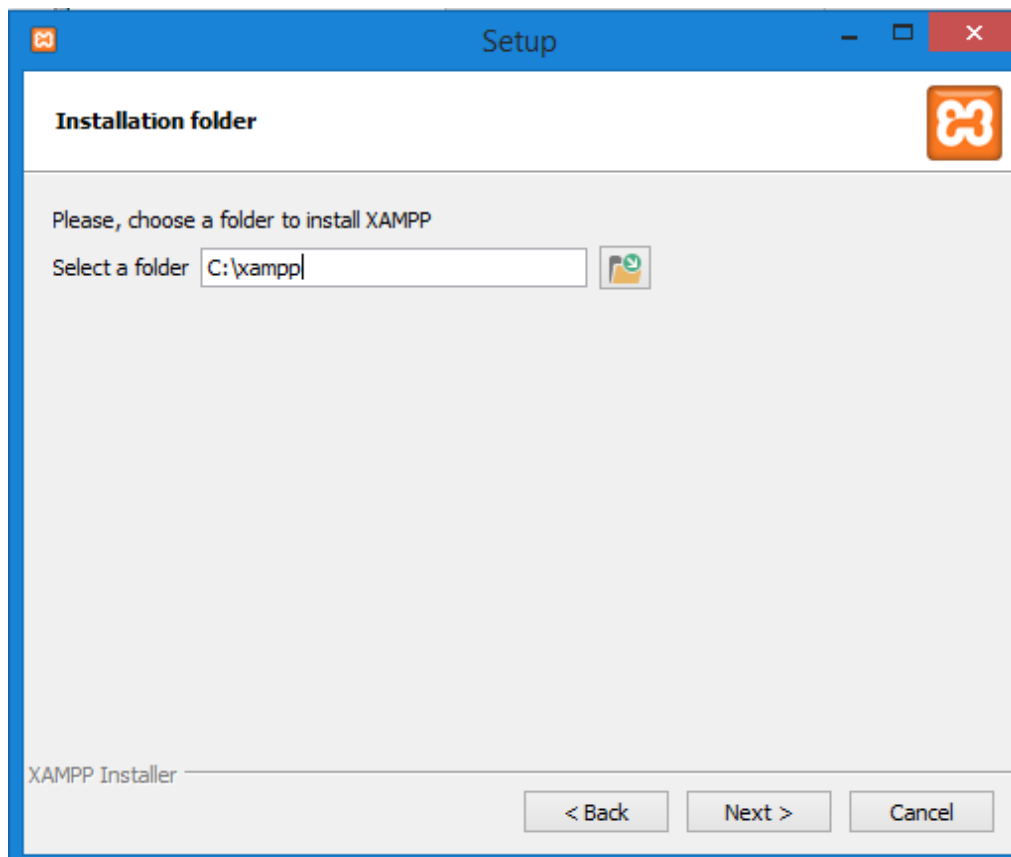
6. Click the Next button.



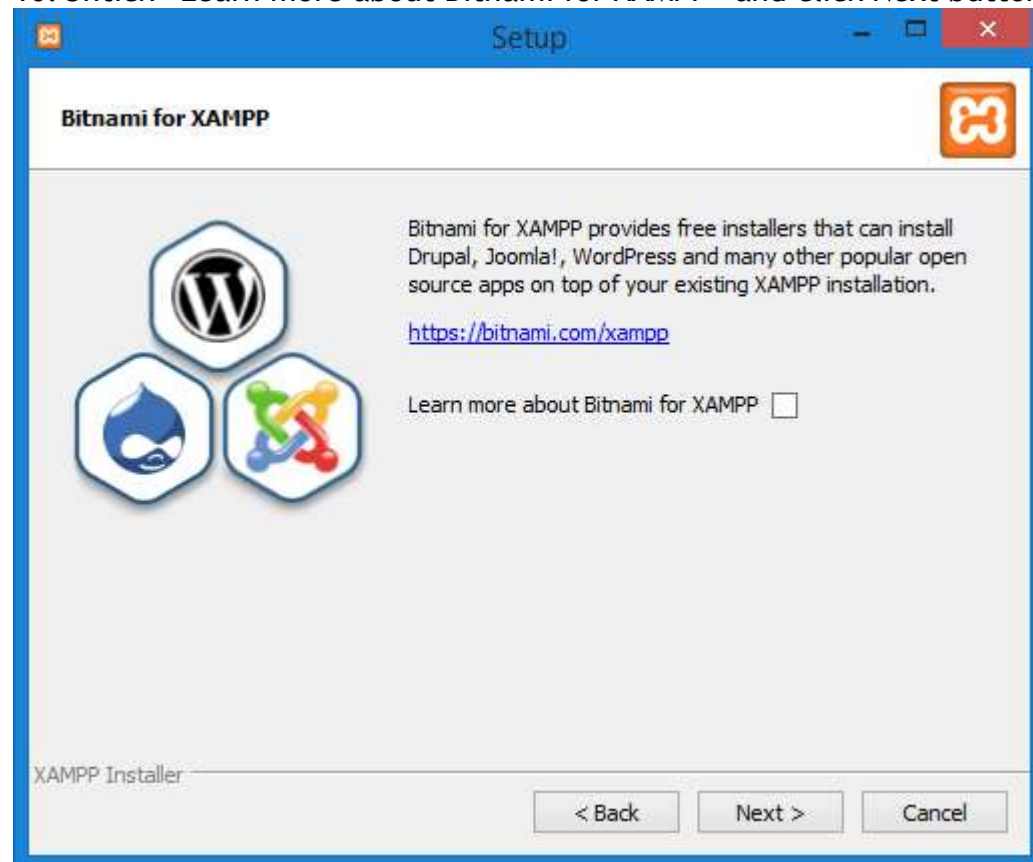
7. Please ensure you choose **Tomcat** and **MySQL**.



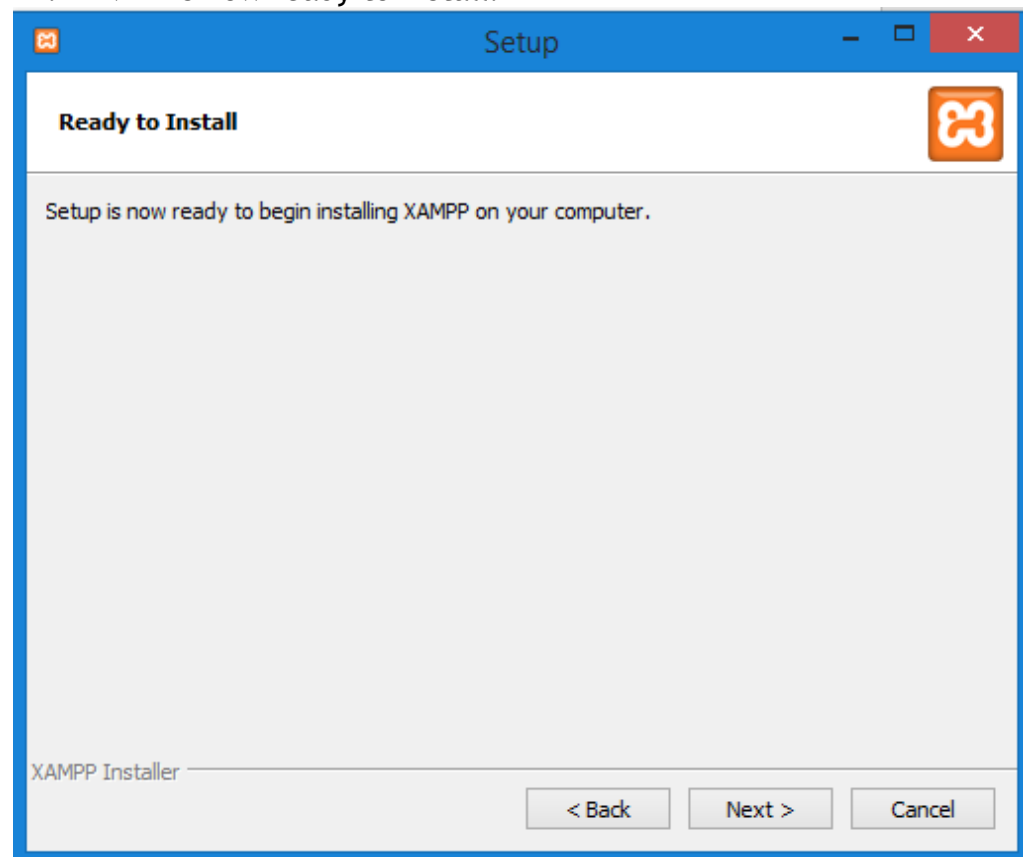
8. Choose a folder as C: \xampp and click the Next > button.



10. Untick “Learn more about Bitnami for XAMPP” and click Next button.



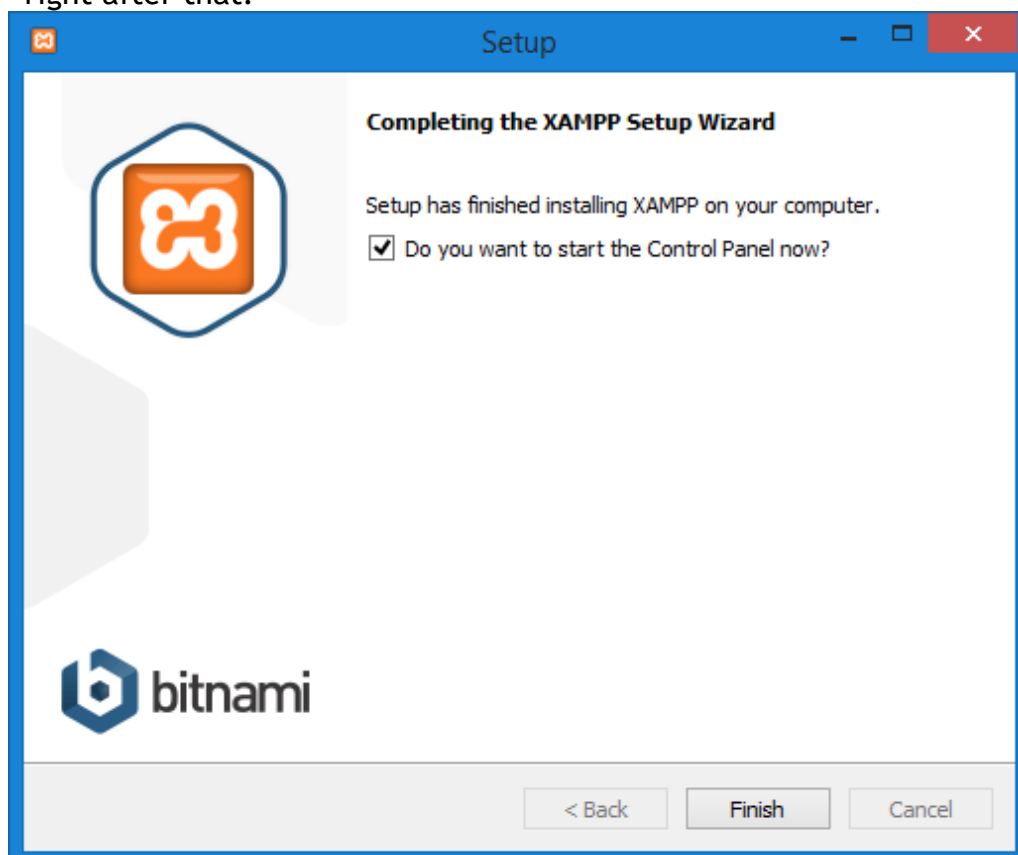
11. XAMPP is now ready to install.



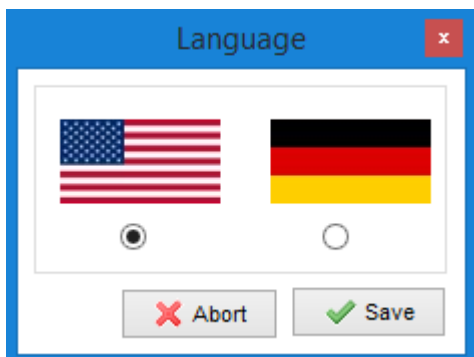
12. Wait for the installation process to complete.



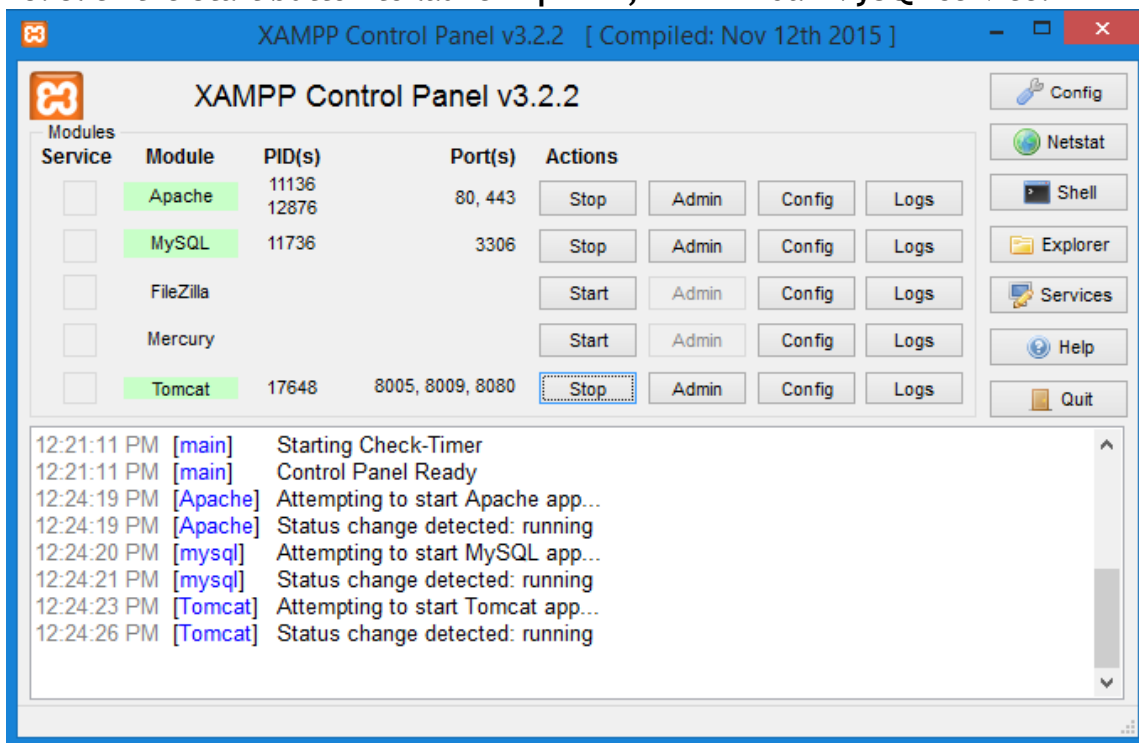
13. Click Finish button once completed and the Control Panel should start right after that.



14. Only for first time launching, select English as the default language for XAMPP. Click the Save button.



15. Click the Start button to launch **Apache**, **Tomcat** dan **MySQL** service.



Note: Why do we need Apache? We will use phpmyadmin as a tool to manage our database in the upcoming tasks. You may use other tools such as MySQL Workbench, Netbeans Database Manager etc. Do not confuse with the functions of various instruments used during the JSP development. Try to understand the context of the usage of each of them.

Reflection

What have you learnt from this exercise?

Apache is a popular open-source web server software that is widely used for serving web content over the internet.

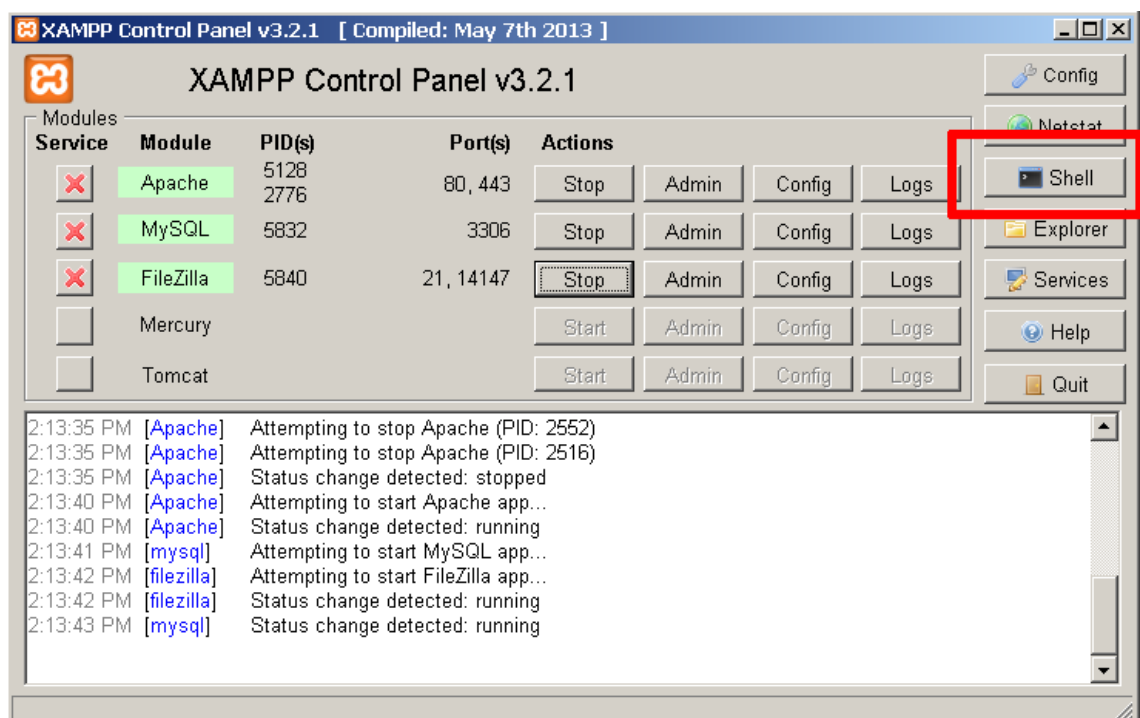
Task 2: Change the Default Root Password of MySQL Database

Objective:	Change the root password of MySQL Database
Problem Description:	To reset a new root password for MySQL Database
Estimated time:	5 minutes

By default, the MySQL installation that ships with XAMPP have an empty root password. It is a severe security risk, especially if you plan to use XAMPP in production scenarios.

To change the MySQL root password, follow these steps:

1. Ensure that the MySQL server is running.
2. Open your Windows command prompt by clicking the "Shell" button in the XAMPP control panel.

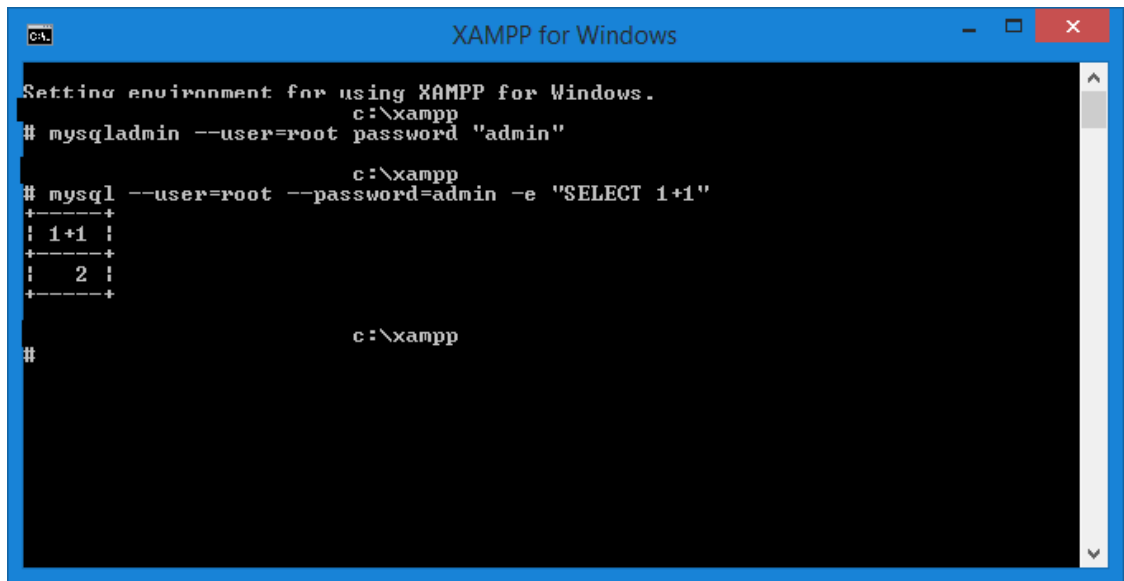


3. Use the mysqladmin command-line utility to alter the MySQL password, using the following syntax (Note: We will use "admin" as our password during the lesson):

```
mysqladmin --user=root password "admin"
```

4. To test that your password change has been accepted, by attempting to connect to the MySQL server using the MySQL command-line client in the same directory. For example, you could use the command below to connect to the server and return the results of a calculation:

```
mysql --user=root --password=admin -e "SELECT 1+1"
```



```
Setting environment for using XAMPP for Windows.
c:\xampp
# mysqladmin --user=root password "admin"
c:\xampp
# mysql --user=root --password=admin -e "SELECT 1+1"
+-----+
| 1+1 |
+-----+
| 2 |
+-----+
c:\xampp
#
```

5. To update the password in the future, you can use the following syntax:

```
mysqladmin --user=root --password=oldpassword
password "newpassword"
```

6. To test that your password change has been accepted, by attempting to connect to the MySQL server using the MySQL command-line client in the same directory. For example, you could use the command below to connect to the server and return the results of a calculation:

```
mysql --user=root --password=gue55me -e "SELECT 1+1"
```

IMPORTANT:

To avoid forgetting the password during our lesson, please set the MySQL password as **admin** ONLY..!

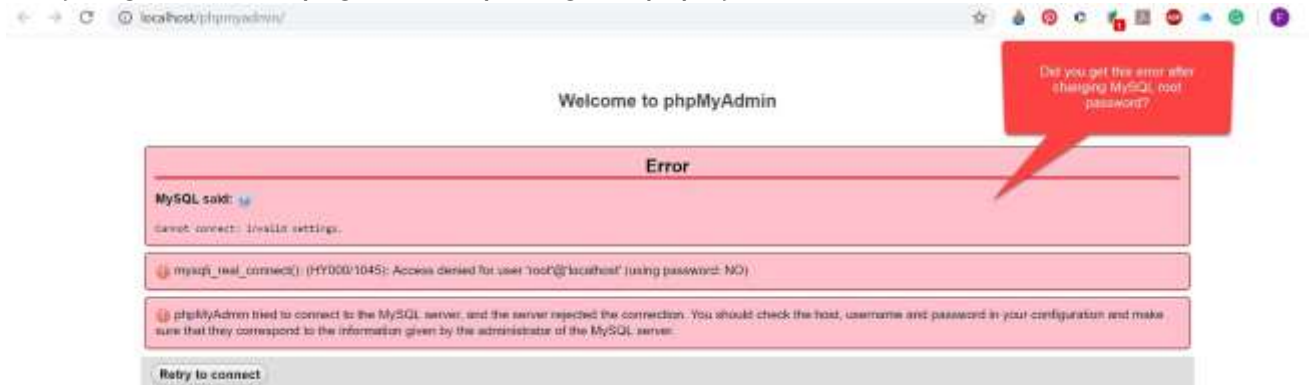
Remark: The above instructions are taken from Apache Friends Documentation. You may retrieve the documentation at <http://localhost/dashboard/docs/reset-mysql-password.html> after done with the XAMPP installation.

Reflection

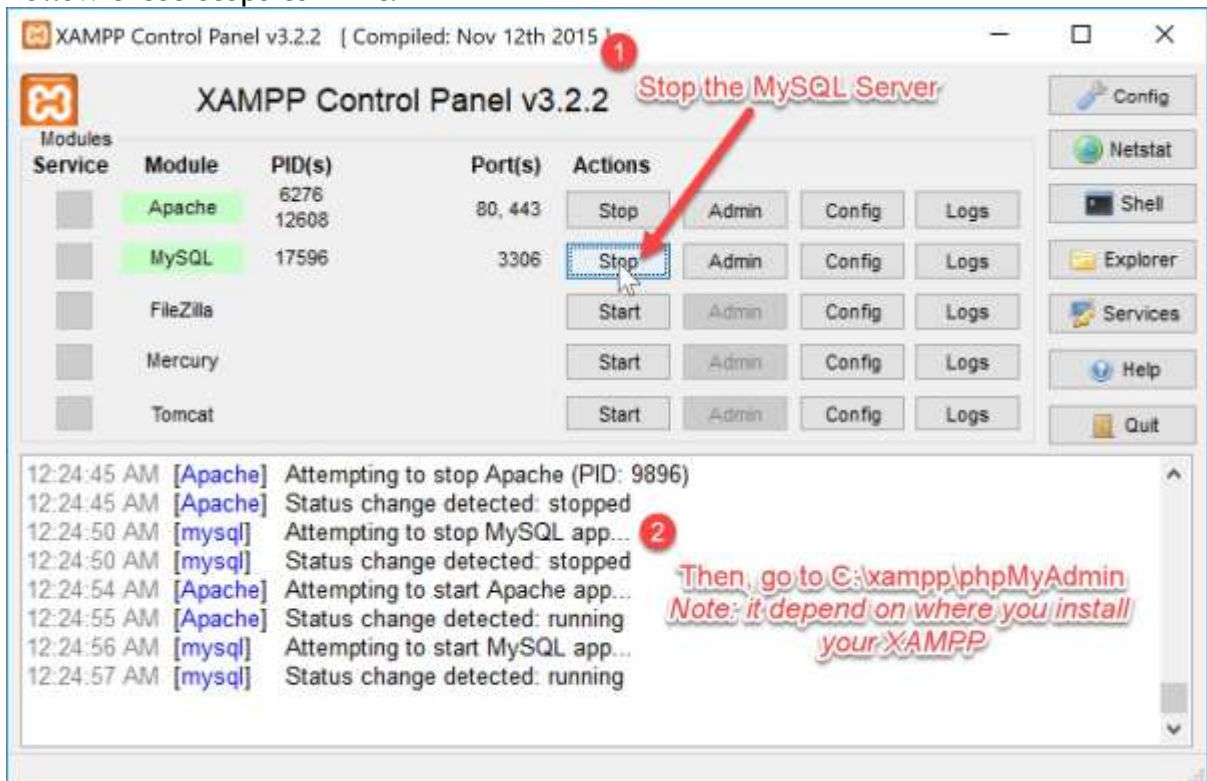
What have you learned from this exercise?

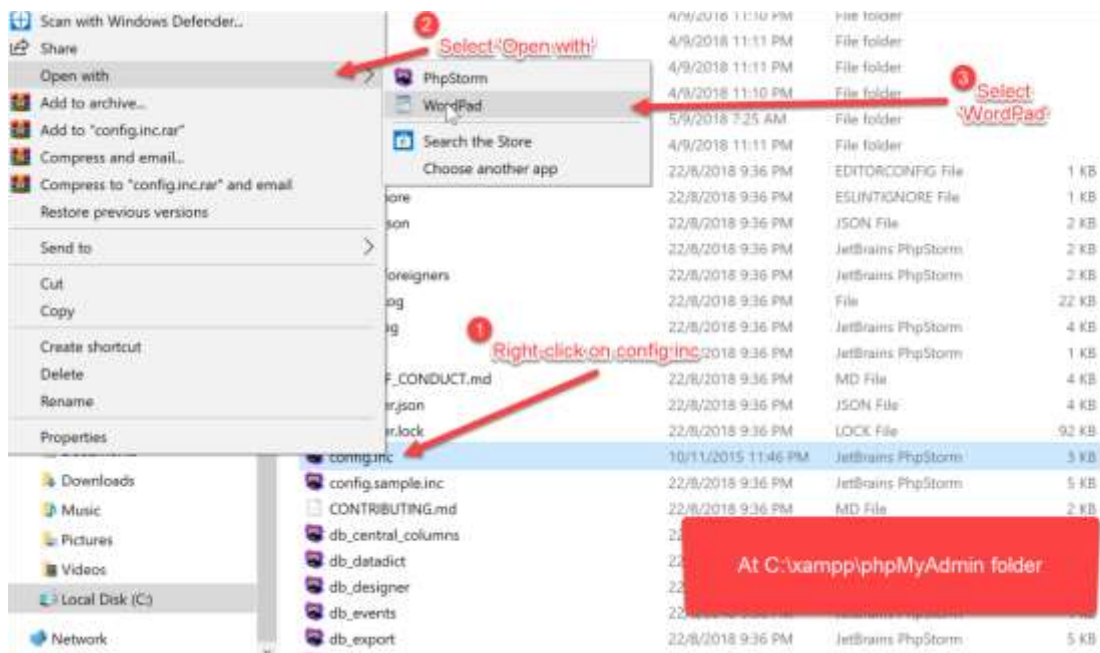
Troubleshooting Notes: Fixing phpMyAdmin Error

Did you get an error page when opening the phpMyAdmin?



Follow these steps to fix it.





```

password in
* cookie
*/
$cfg['blowfish_secret'] = 'xampp'; /* YOU SHOULD CHANGE THIS
FOR A MORE SECURE COOKIE AUTH! */

/*
* Servers configuration
*/
$i = 0;

/*
* First server
*/
$i++;

/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'admin';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = false;
$cfg['Lang'] = '';

/* Bind to the localhost ipv4 address and tcp */
$cfg['Servers'][$i]['host'] = '127.0.0.1';
$cfg['Servers'][$i]['connect_type'] = 'tcp';

/* User for advanced features */
$cfg['Servers'][$i]['controluser'] = 'pma';
$cfg['Servers'][$i]['controlpass'] = '';

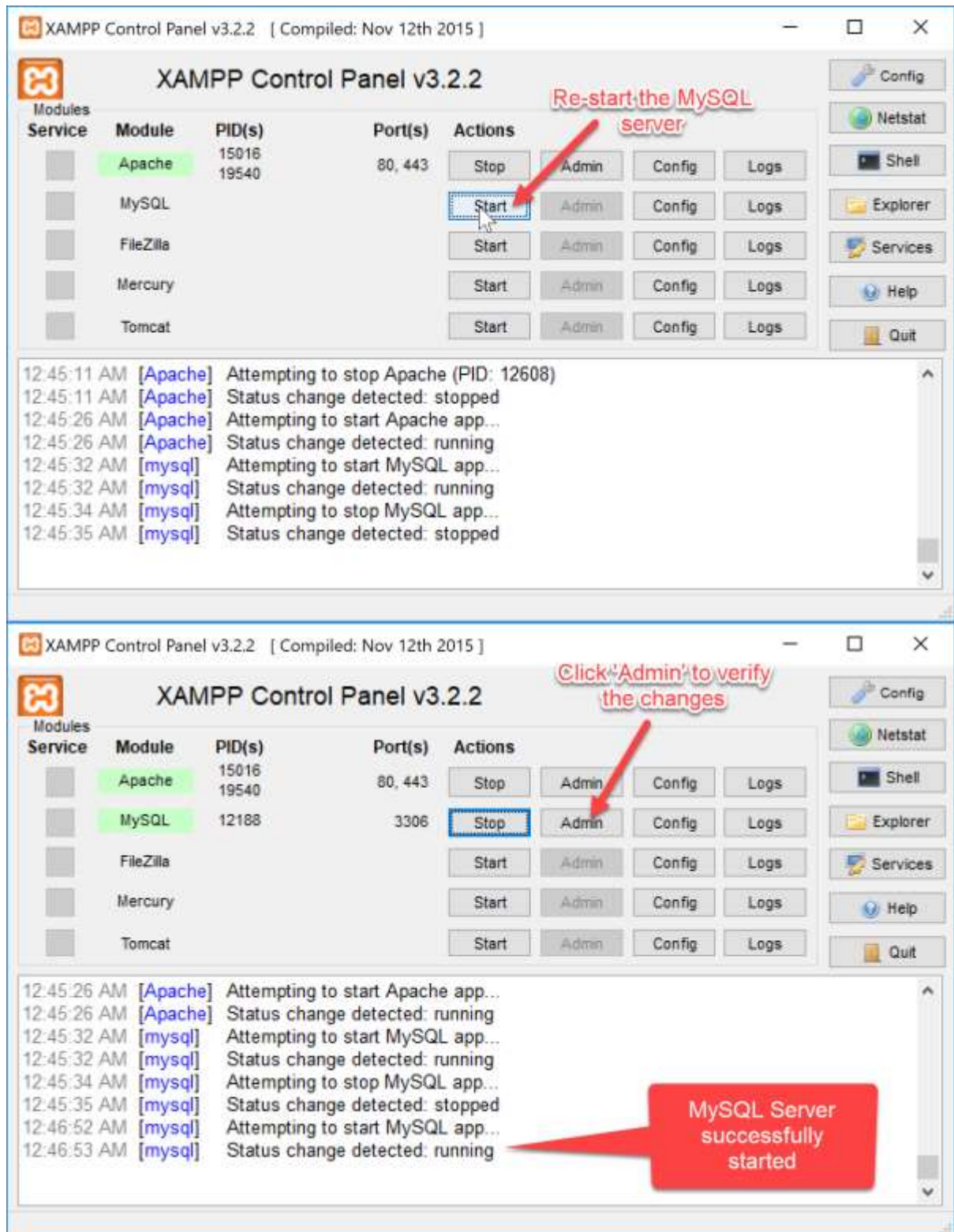
/* Advanced phpMyAdmin features */
$cfg['Servers'][$i]['pmadb'] = 'phpmyadmin';
$cfg['Servers'][$i]['bookmarktable'] = 'pma_bookmark';
$cfg['Servers'][$i]['relation'] = 'pma_relation';
$cfg['Servers'][$i]['table_info'] = 'pma_table_info';
$cfg['Servers'][$i]['table_coords'] = 'pma_table_coords';
$cfg['Servers'][$i]['pdf_pages'] = 'pma_pdf_pages';
$cfg['Servers'][$i]['column_info'] = 'pma_column_info';

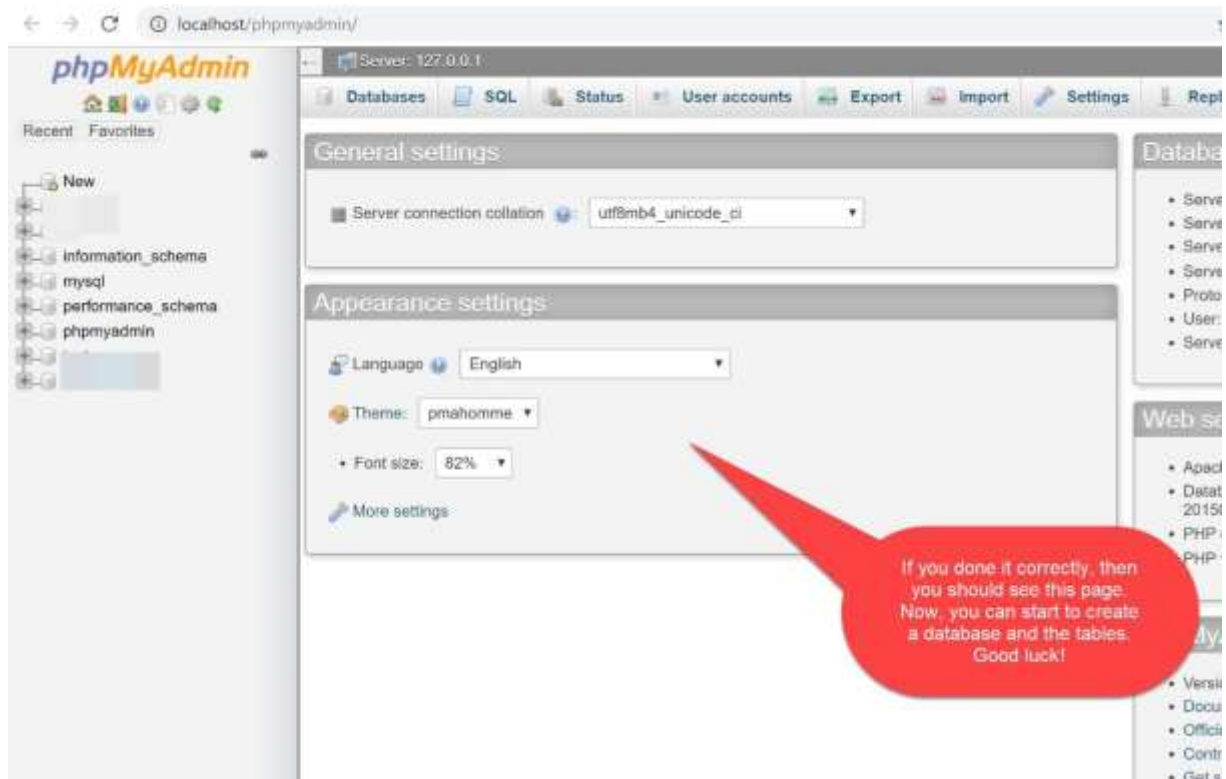
```

1 Put your root password here. In this example, 'admin' is the root password.

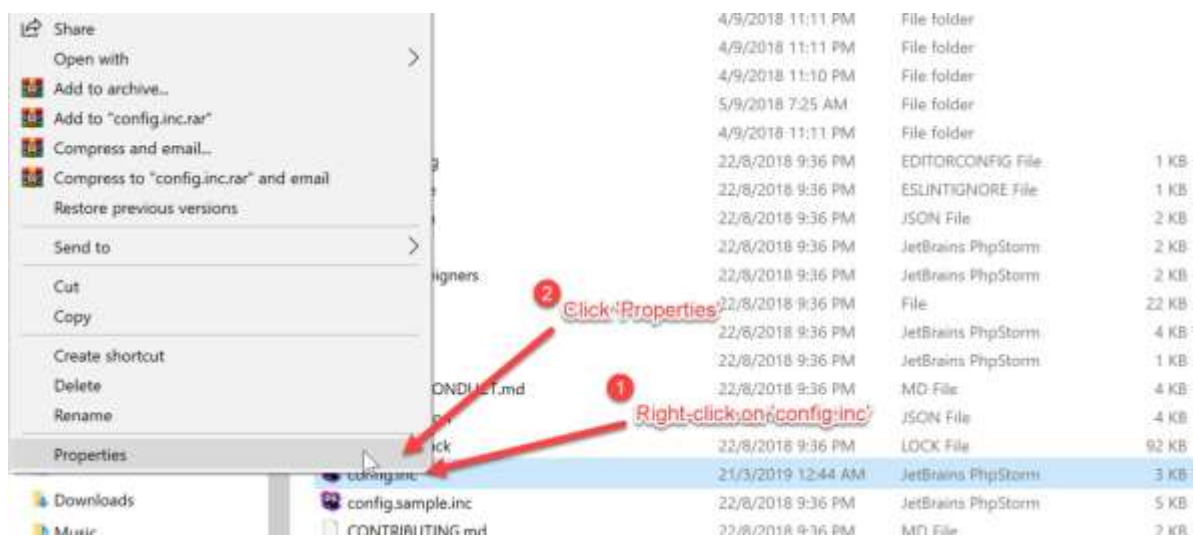
2 Change the value from 'true' to 'false'

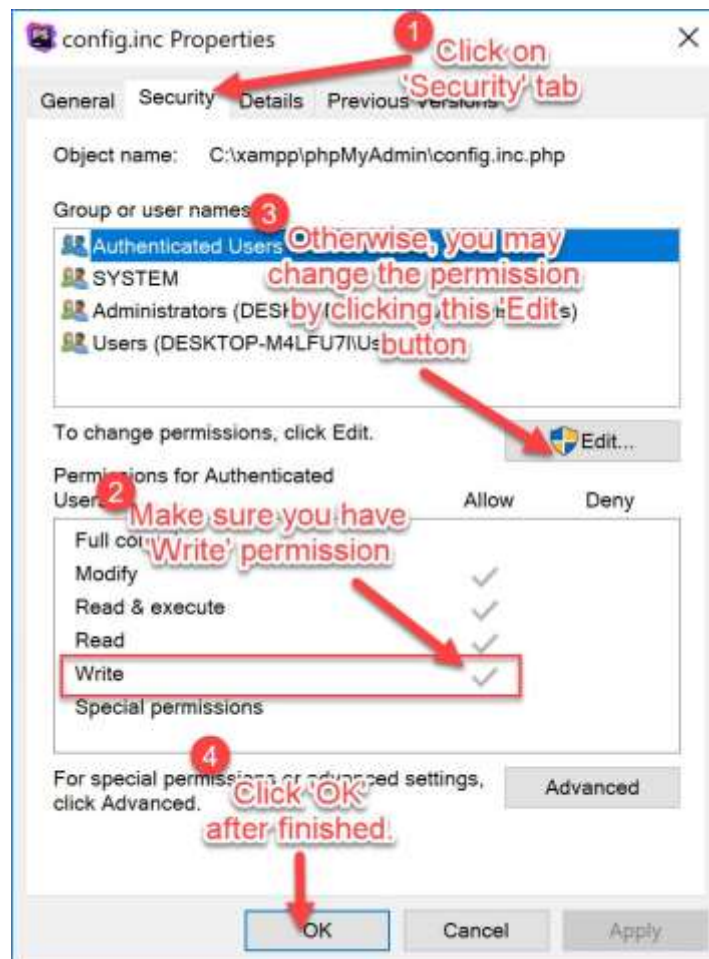
3 Save the file





If you can't edit or save the *config.inc*, follow the steps below:



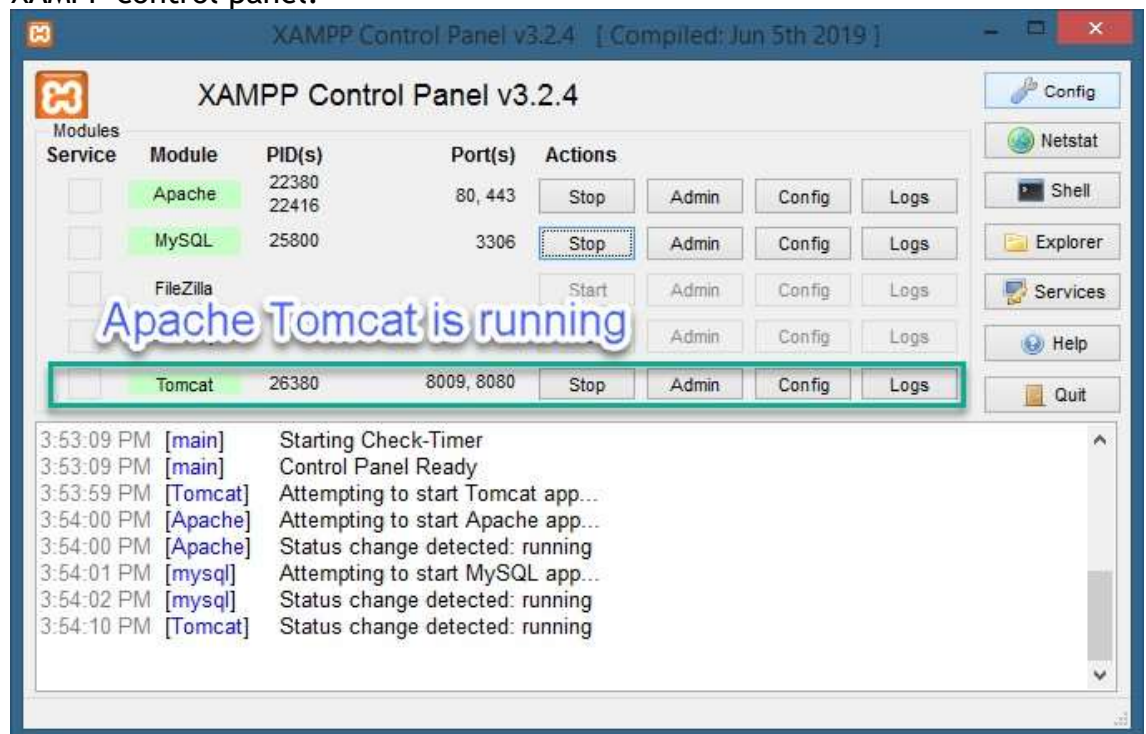


Task 3: Managing Apache Tomcat

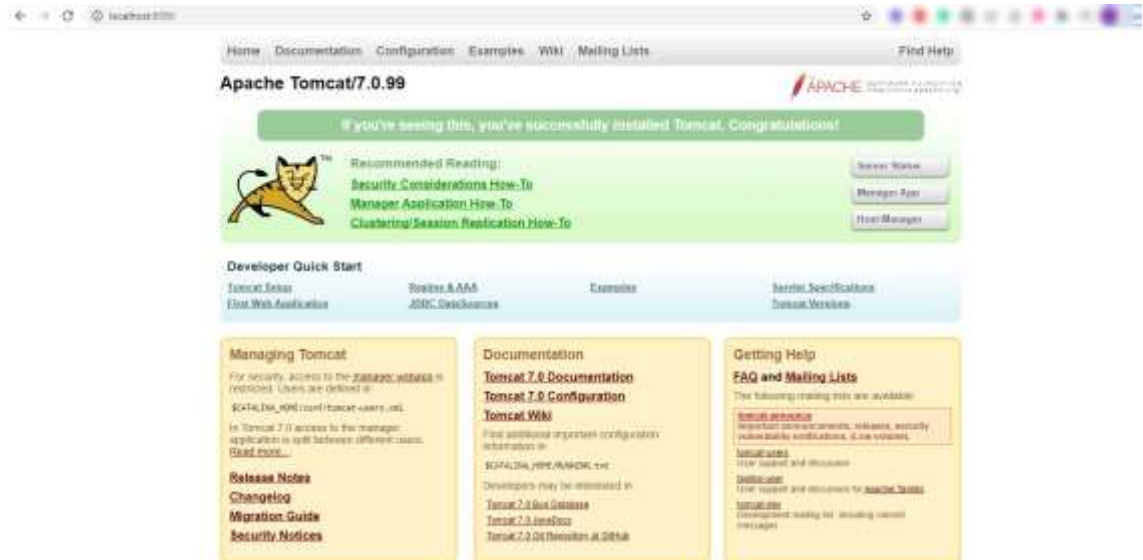
Objective:	Testing the access and add a new user to Apache Tomcat in XAMPP
Problem Description:	To configure user access for Apache Tomcat in XAMPP and test the access at localhost.
Estimated time:	30 minutes

In the previous task, we have successfully installed Apache Tomcat which is part of XAMPP software package.

1. Make sure Apache Tomcat is running. You can see that status from XAMPP control panel.

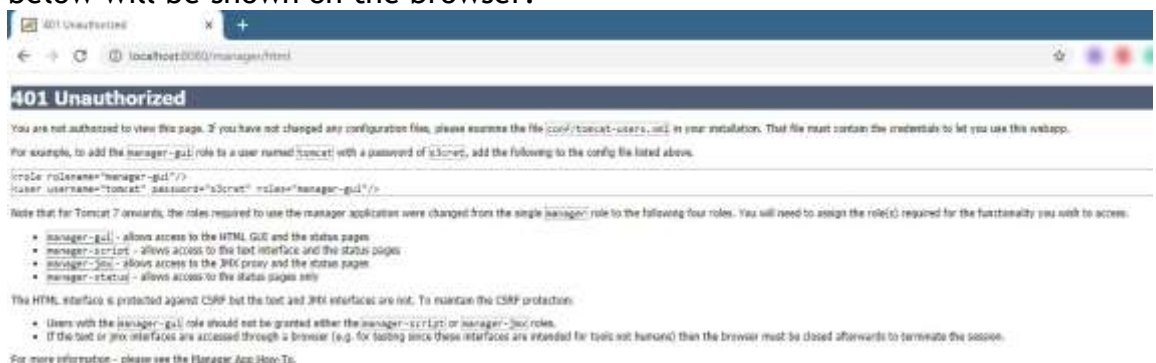


2. Open the browser, and go to <http://localhost:8080/>. If Apache Tomcat is running correctly, you will see a homepage as follows:



3. Click on the “Manager App” of Apache Tomcat/7.0.99 homepage and there will appear a page ask for username and password. Do not panic, just click “Cancel”.

4. As our Apache Tomcat currently does not has any defined user, a page below will be shown on the browser.



5. To define a new user, click **Stop** Apache Tomcat server at the XAMPP Control Panel and click “Config” > tomcat-users.xml

9. Now, **start** the Apache Tomcat and repeat Step 2 and Step 3. This time, put the username as “admin” and password as “admin”.
10. If you did the above steps correctly, you will see the screen as below:



Task 4: Netbeans 12.3 IDE Installation

Objective: Netbeans 12.3 IDE Installation

Problem Description: To setup a proper environment for Java Web Application development.

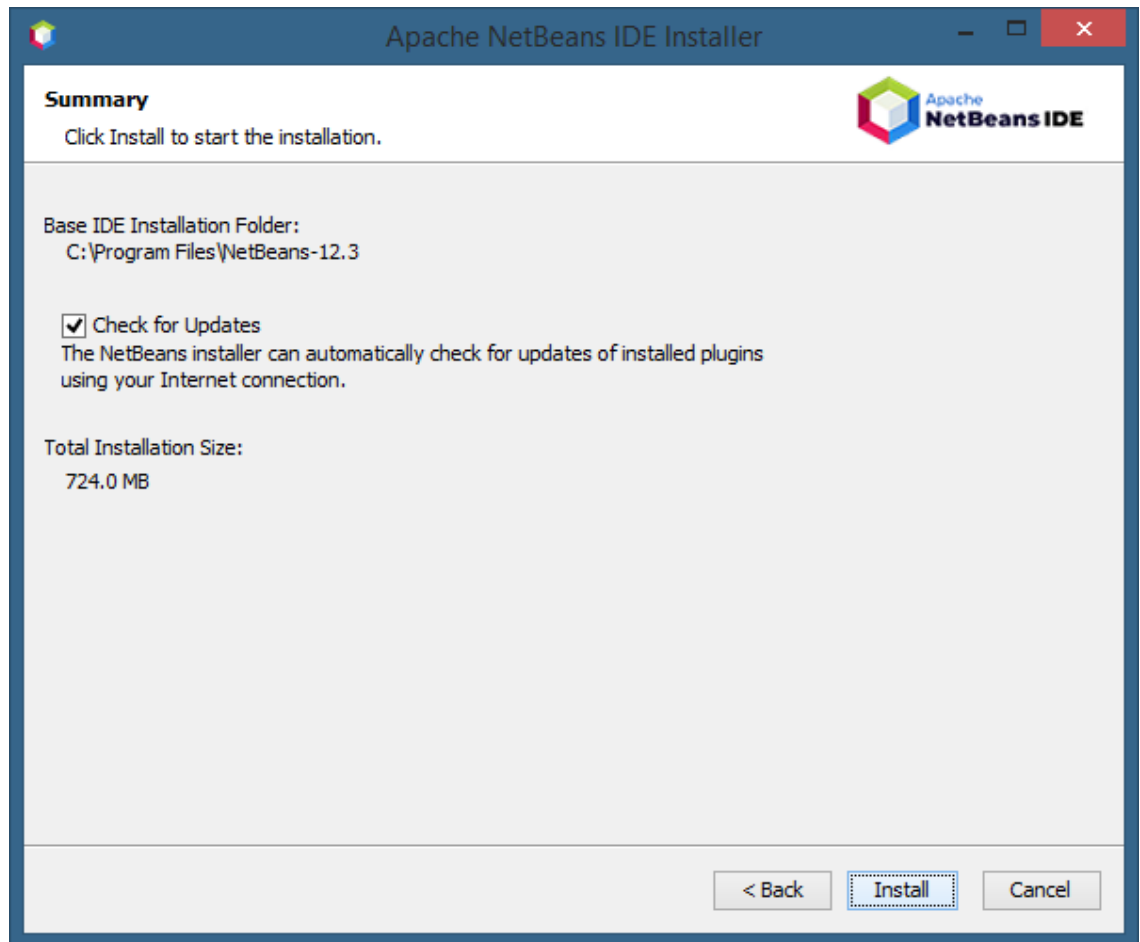
Estimated time: 30 minutes

To start the Netbeans 12.3 installation process, follow the steps below:

11. Go to <https://netbeans.apache.org/download/index.html>
12. Click “Download”. Choose the installer that appropriate with your operating system and computer architecture (x32 or x64). A correct installation program could be something like this *Apache-NetBeans-12.3-bin-windows-x64.exe*



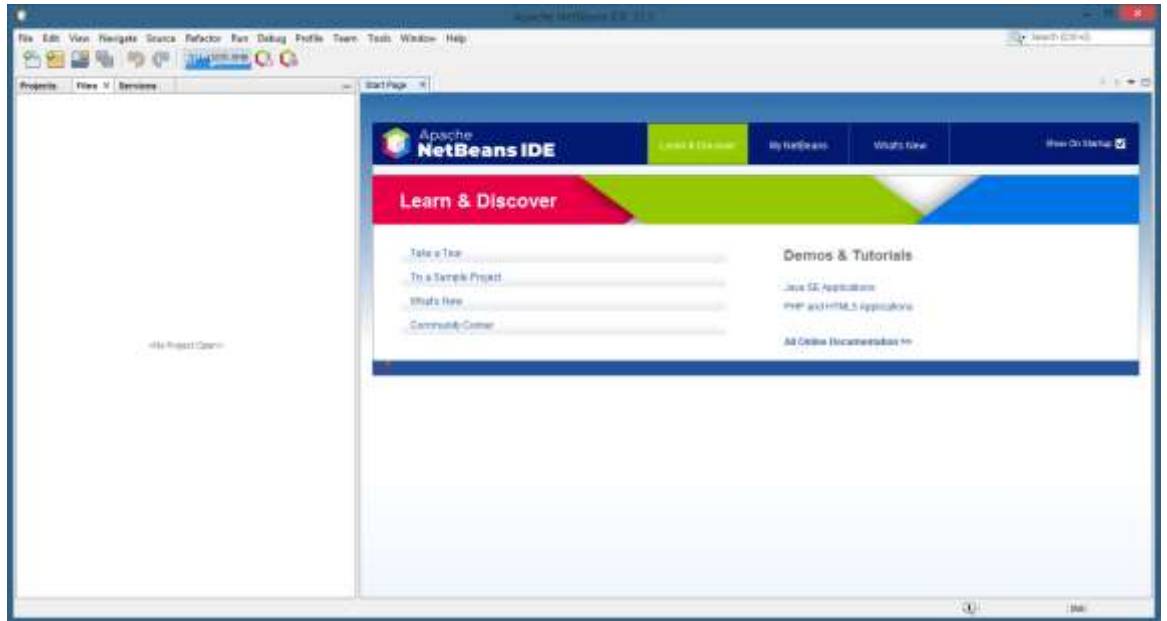
13. After the download finish, double-click on the installer and click “Next” until you reach the following screen:



14. Make sure you tick the “Check for Updates”, then click Install. Wait until the installation finish.
15. Run the Netbeans. If you have previous installation, the pop-up windows as below may appear and simply click “Yes”. This will import your previous configuration to a new one.



16. Finally, you will get the following screen.



17. Now, you are ready to start developing your first Java web application.
Well done!

Task 5: Linking Netbeans to Apache Tomcat and Writing a Simple Java Servlet

Objective:

Link Netbeans to Apache Tomcat and Writing a Simple Java Servlet

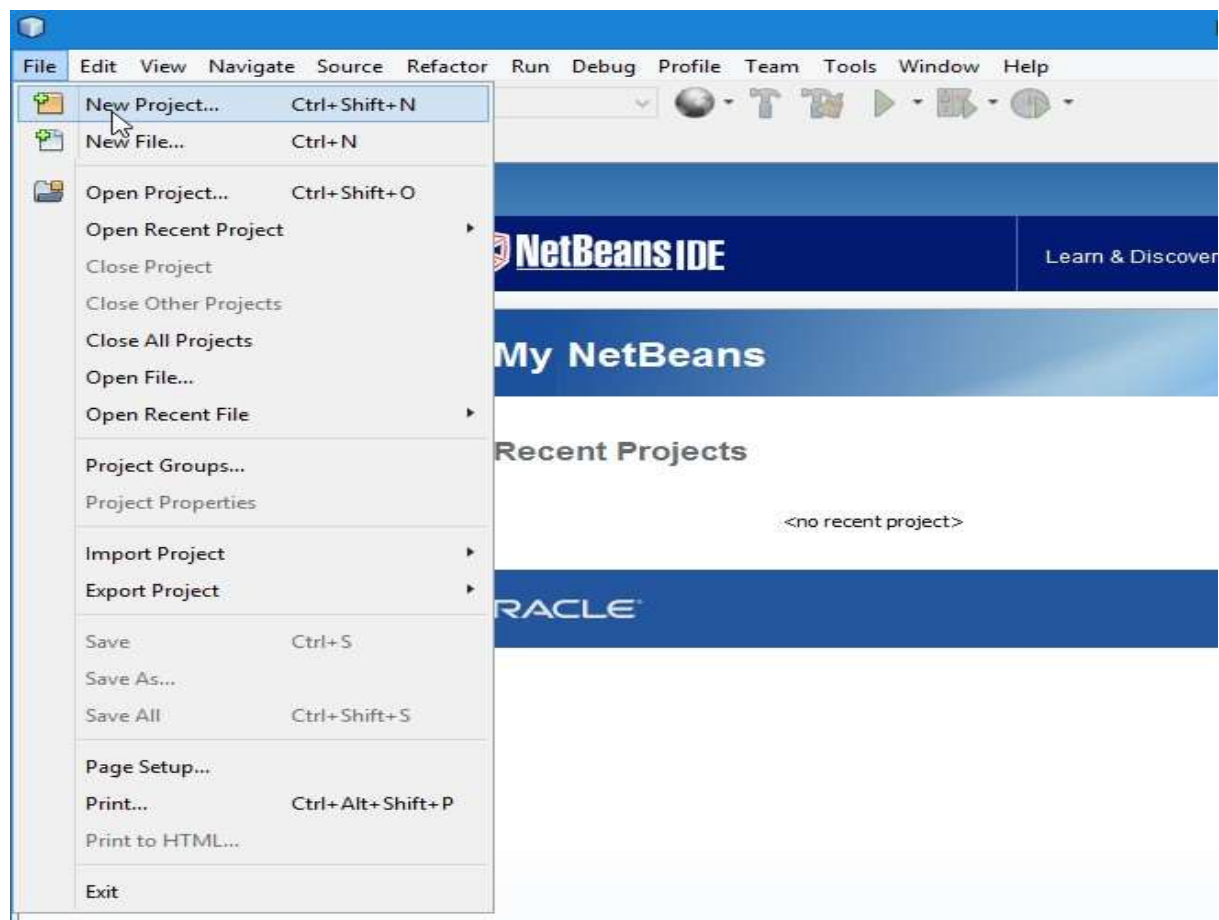
Problem**Description:**

To write a simple Java Servlet

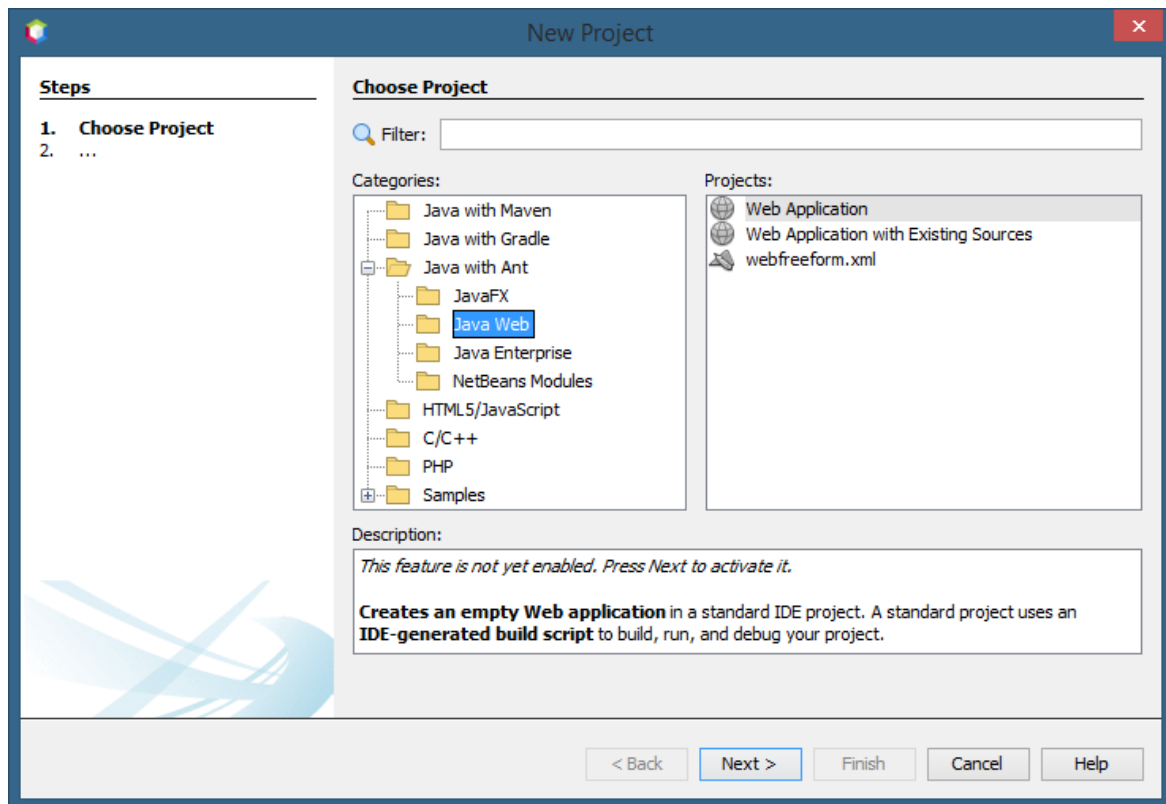
Estimated time:

15 minutes

1. Create a directory `C:\[MATRICNUMBER]`.
2. Go to `C:\[MATRICNUMBER]` Lab's directory and create sub-directory as *Lab 1 - Servlet*.
3. Open your NetBeans.
4. Go to File -> New Project



5. Select Java with Ant -> Java Web -> Web Application and click Next.



1. Type Project Name: *MyFirstServlet*.
2. Click Browse and choose Project Location: *C:\[MATRICNUMBER]\Lab 1 - Servlet*. Then click the Next button.

Steps

1. Choose Project
2. **Name and Location**
3. Server and Settings
4. Frameworks

Name and Location

Project Name:

Project Location: [Browse...](#)

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder: [Browse...](#)

Select Project Location

Look in:

Recent Items

Desktop

Documents

This PC

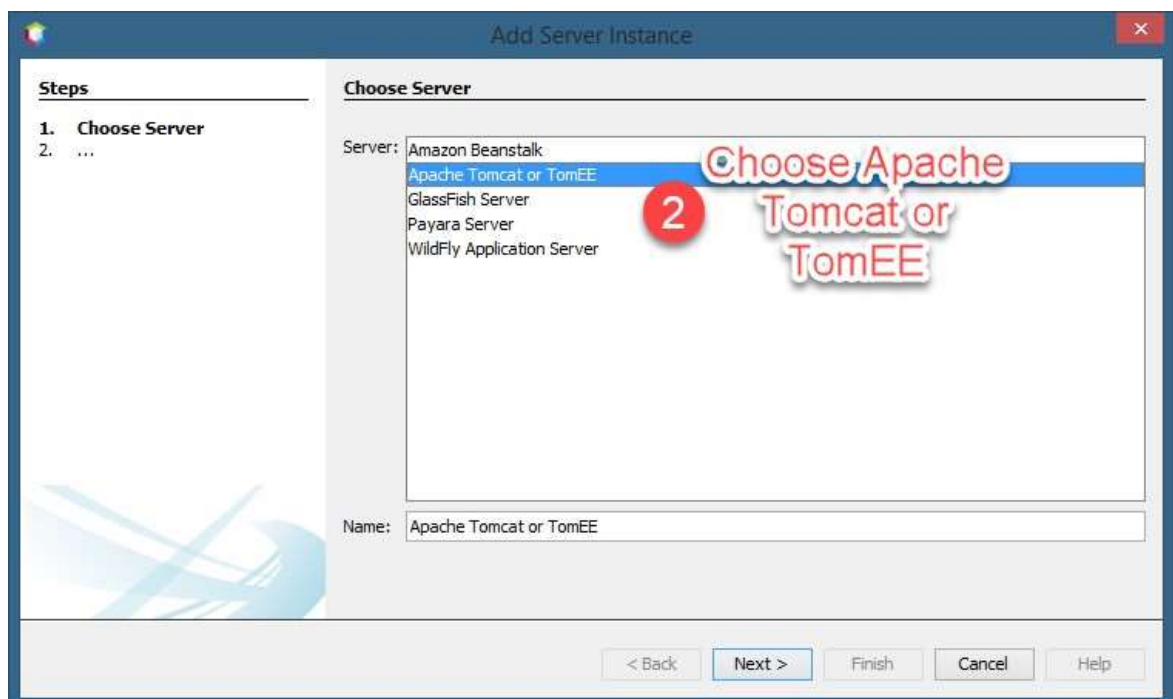
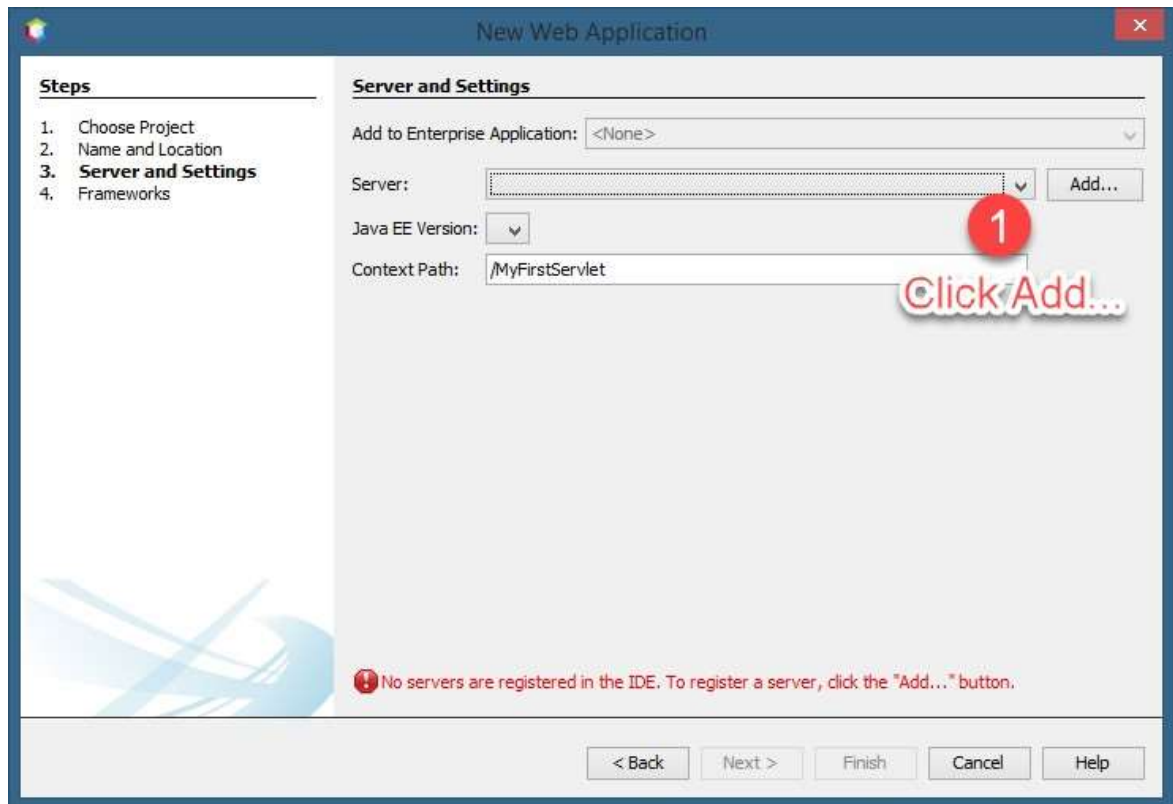
Network

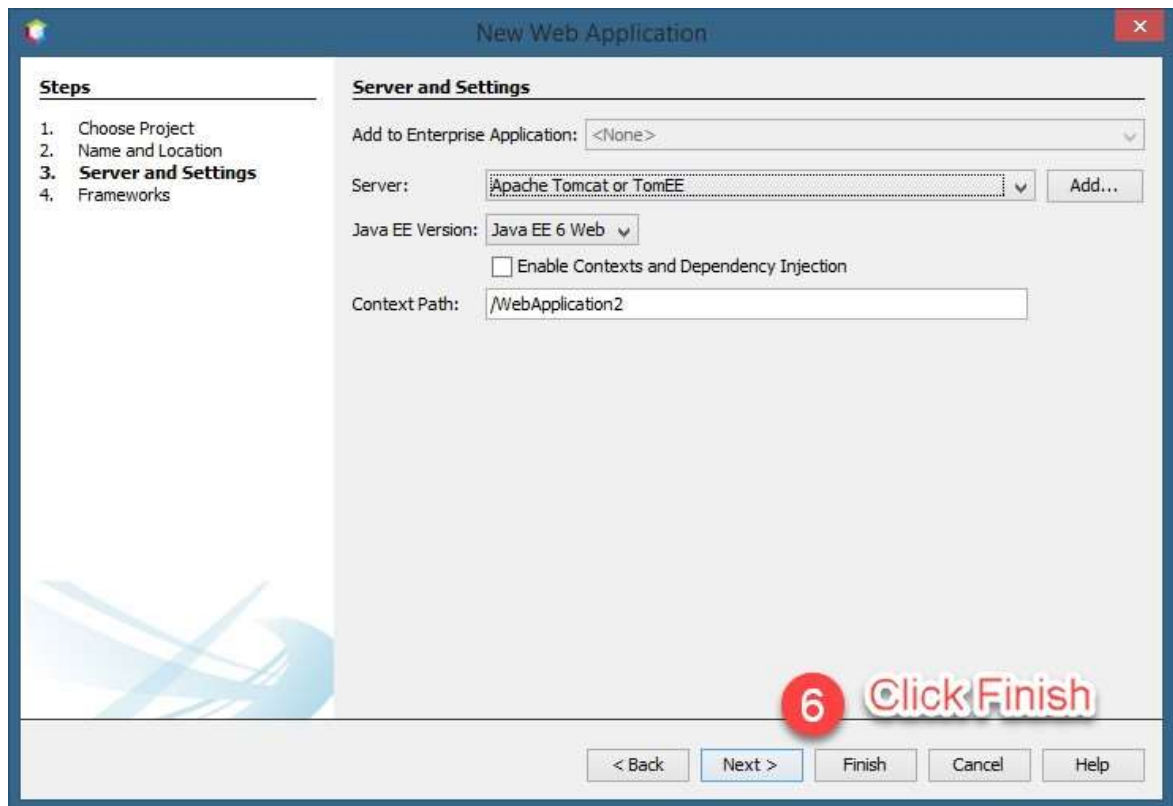
File name:

Files of type:

[Open](#) [Cancel](#)

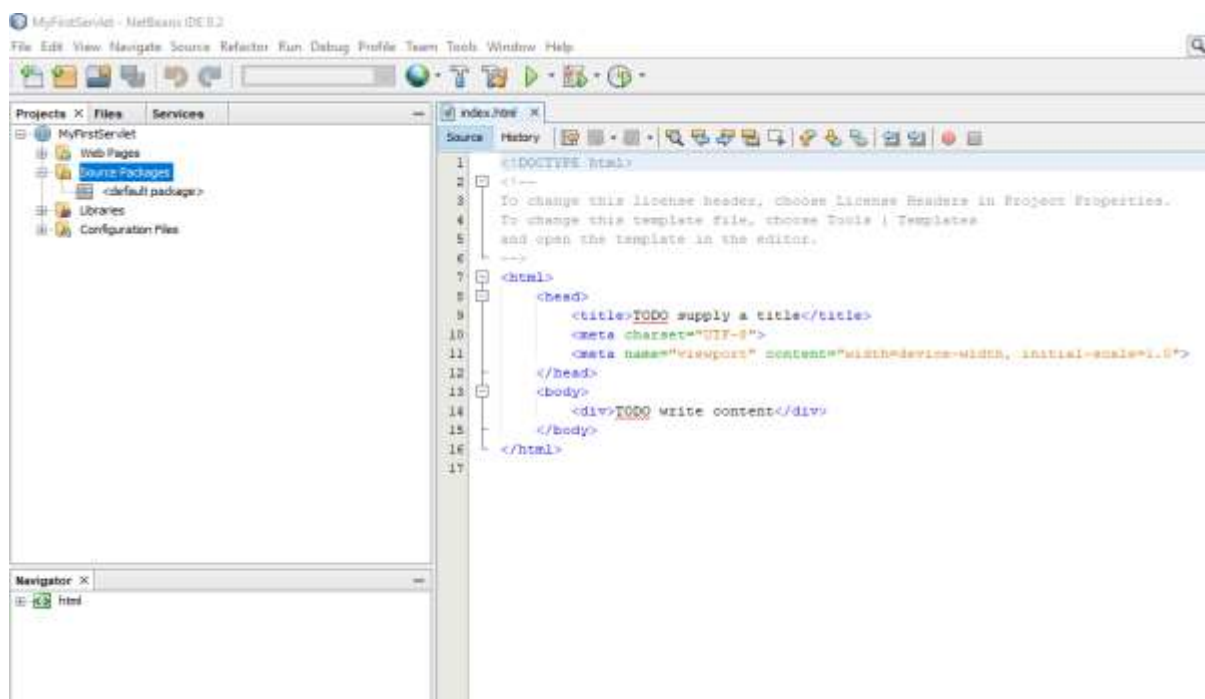
3. Before we can run our Java Web Application, we need to link it to Apache Tomcat Server. Follow the diagrams below to link Netbeans with Apache Tomcat in XAMPP.



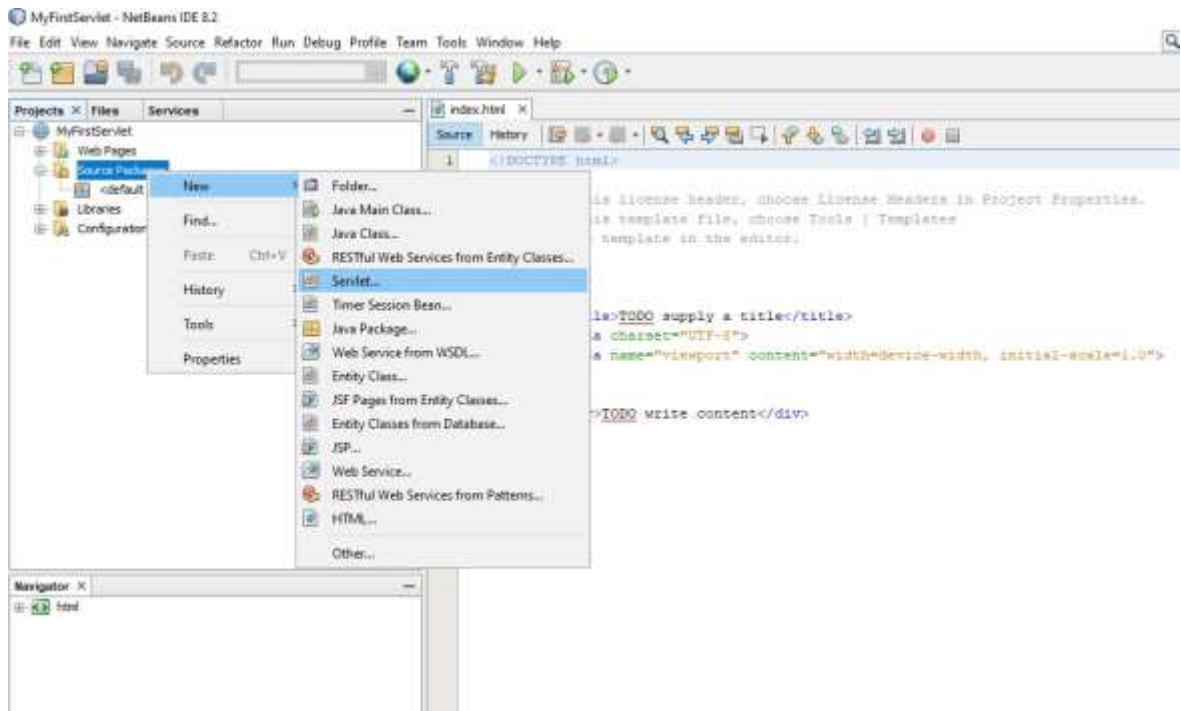


Note: the above steps only need to be ran for the first time you link the Netbeans to Apache Tomcat in XAMPP. After this, you just need to select it from the screen.

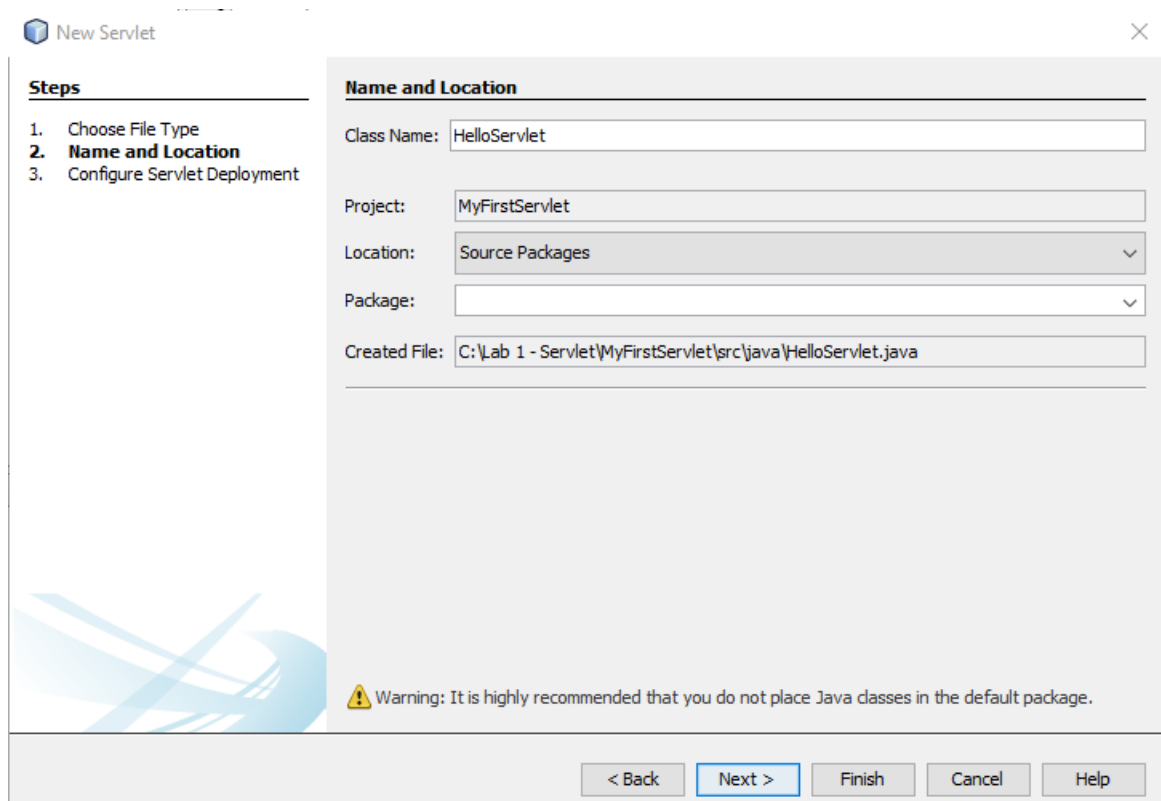
4. If everything okay, you will see the following screen:



5. Create a new Servlet file.



6. Name your servlet as *HelloServlet*.



7. Tick the “Add information to deployment descriptor (web.xml)”

New Servlet

Steps

1. Choose File Type
2. Name and Location
3. **Configure Servlet Deployment**

Configure Servlet Deployment

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

☒ Add information to deployment descriptor (web.xml):

Class Name:

Servlet Name:

URL Pattern(s):

Initialization Parameters:

Name	Value
------	-------

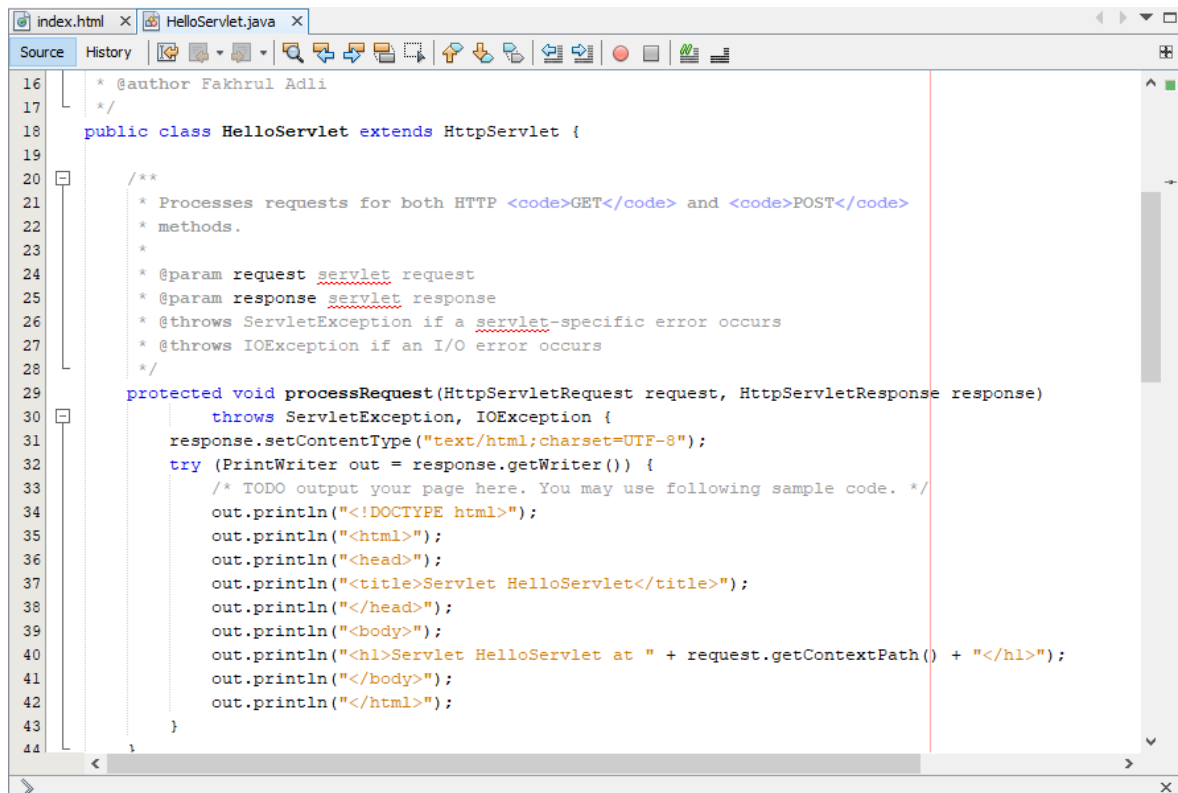
New Edit... Delete

< Back Next > Finish Cancel Help

8. NetBeans has produced a new file named *HelloServlet.java*. You may see the location of it on the left side of NetBeans editor. On the right, are the servlet codes generated by NetBeans.

```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6
7  import java.io.IOException;
8  import java.io.PrintWriter;
9  import javax.servlet.ServletException;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 /**
15 *
16 * Another Festival Ad.
17 */
18 public class HelloServlet extends HttpServlet {
19
20
21     /**
22      * Processes requests for both HTTP GET and POST
23      * methods.
24      * @param request servlet request
25      * @param response servlet response
26      * @throws ServletException if a servlet-specific error occurs
27      * @throws IOException if an I/O error occurs
28      */
29     protected void processRequest(HttpServletRequest request, HttpServletResponse response) {
```

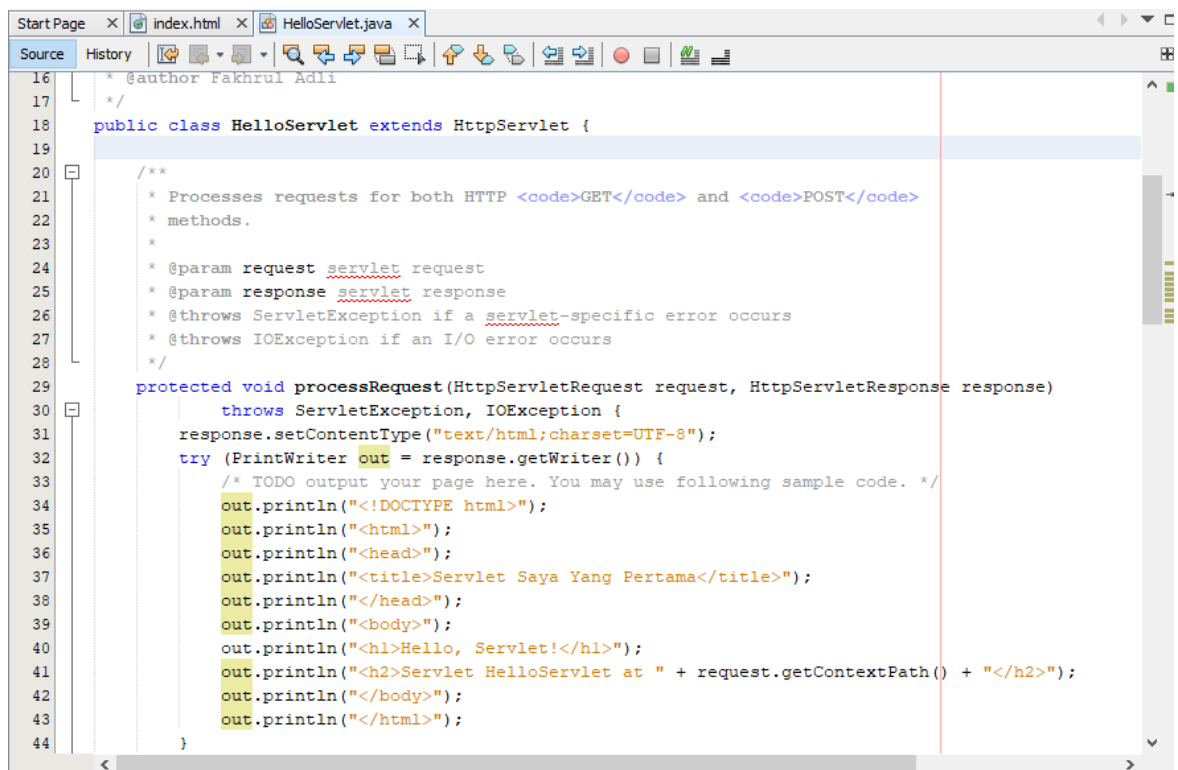
9. Browse through the servlet file until you find the processRequest() method.



The screenshot shows an IDE window with two tabs: 'index.html' and 'HelloServlet.java'. The 'HelloServlet.java' tab is active, displaying the source code of the `HelloServlet` class. The code is as follows:

```
16  * @author Fakhru Adli
17  */
18  public class HelloServlet extends HttpServlet {
19
20      /**
21       * Processes requests for both HTTP GET and POST
22       * methods.
23       *
24       * @param request servlet request
25       * @param response servlet response
26       * @throws ServletException if a servlet-specific error occurs
27       * @throws IOException if an I/O error occurs
28       */
29      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30          throws ServletException, IOException {
31          response.setContentType("text/html;charset=UTF-8");
32          try (PrintWriter out = response.getWriter()) {
33              /* TODO output your page here. You may use following sample code. */
34              out.println("<!DOCTYPE html>");
35              out.println("<html>");
36              out.println("<head>");
37              out.println("<title>Servlet HelloServlet</title>");
38              out.println("</head>");
39              out.println("<body>");
40              out.println("<h1>Servlet HelloServlet at " + request.getContextPath() + "</h1>");
41              out.println("</body>");
42              out.println("</html>");
43          }
44      }
```

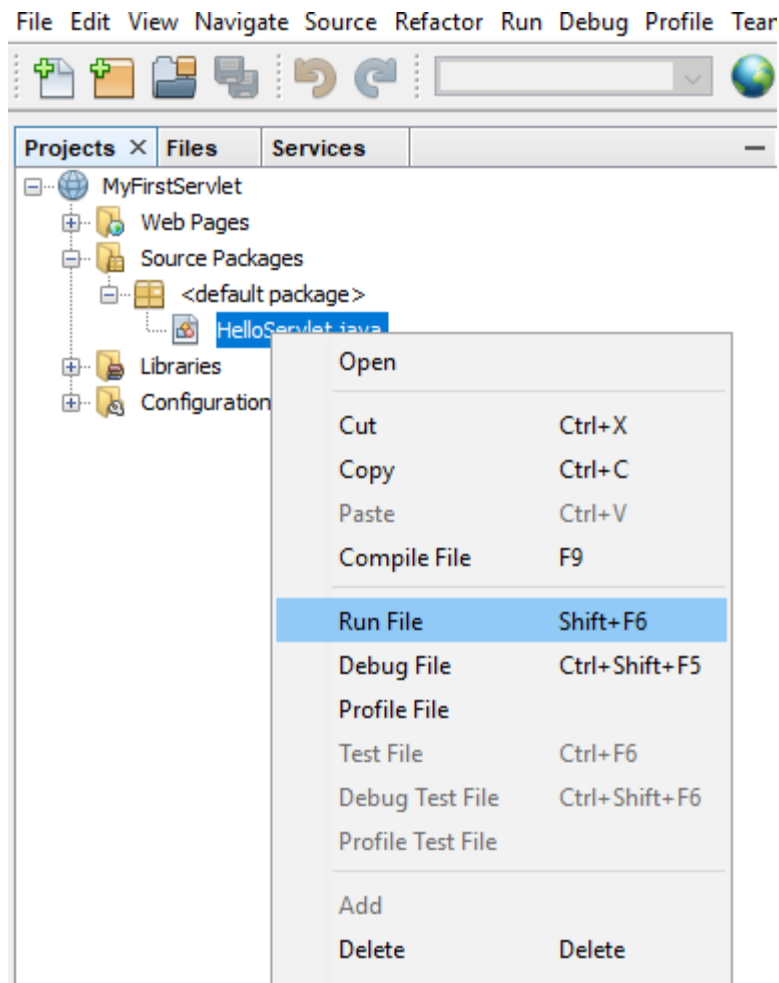
10. We are going to make a simple modification to the existing codes. Modify the them as follows. Refer to line 37, 40 and 41.



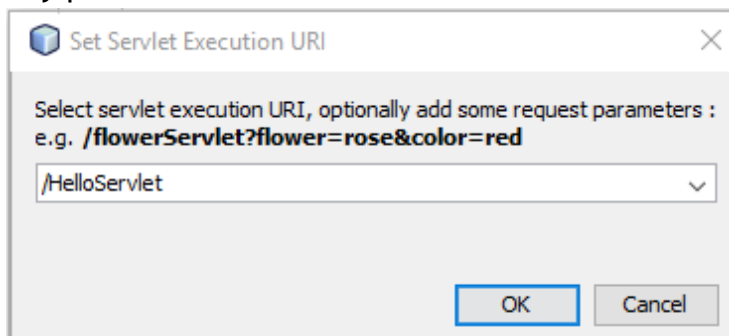
The screenshot shows the same IDE window as before, but the `processRequest` method in `HelloServlet.java` has been modified. The changes are highlighted in yellow in the original image. The modified code is as follows:

```
16  * @author Fakhru Adli
17  */
18  public class HelloServlet extends HttpServlet {
19
20      /**
21       * Processes requests for both HTTP GET and POST
22       * methods.
23       *
24       * @param request servlet request
25       * @param response servlet response
26       * @throws ServletException if a servlet-specific error occurs
27       * @throws IOException if an I/O error occurs
28       */
29      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30          throws ServletException, IOException {
31          response.setContentType("text/html;charset=UTF-8");
32          try (PrintWriter out = response.getWriter()) {
33              /* TODO output your page here. You may use following sample code. */
34              out.println("<!DOCTYPE html>");
35              out.println("<html>");
36              out.println("<head>");
37              out.println("<title>Servlet Saya Yang Pertama</title>");
38              out.println("</head>");
39              out.println("<body>");
40              out.println("<h1>Hello, Servlet!</h1>");
41              out.println("<h2>Servlet HelloServlet at " + request.getContextPath() + "</h2>");
42              out.println("</body>");
43              out.println("</html>");
44          }
45      }
```

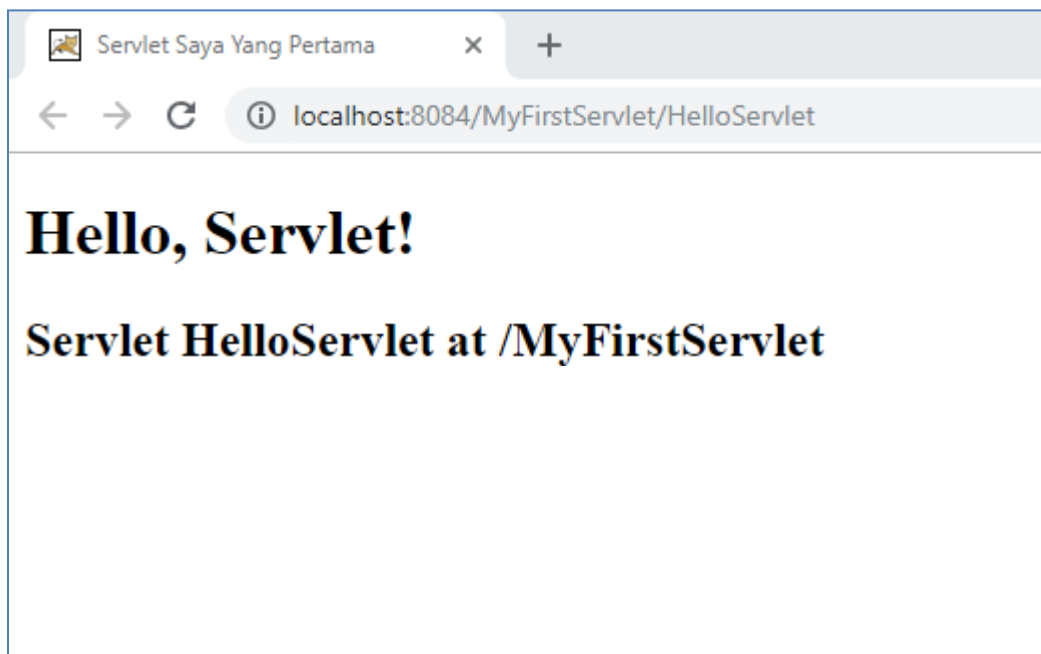

11. After finished your modification on the previous step, run *HelloServlet.java* by right-clicking on it and select *Run File*.



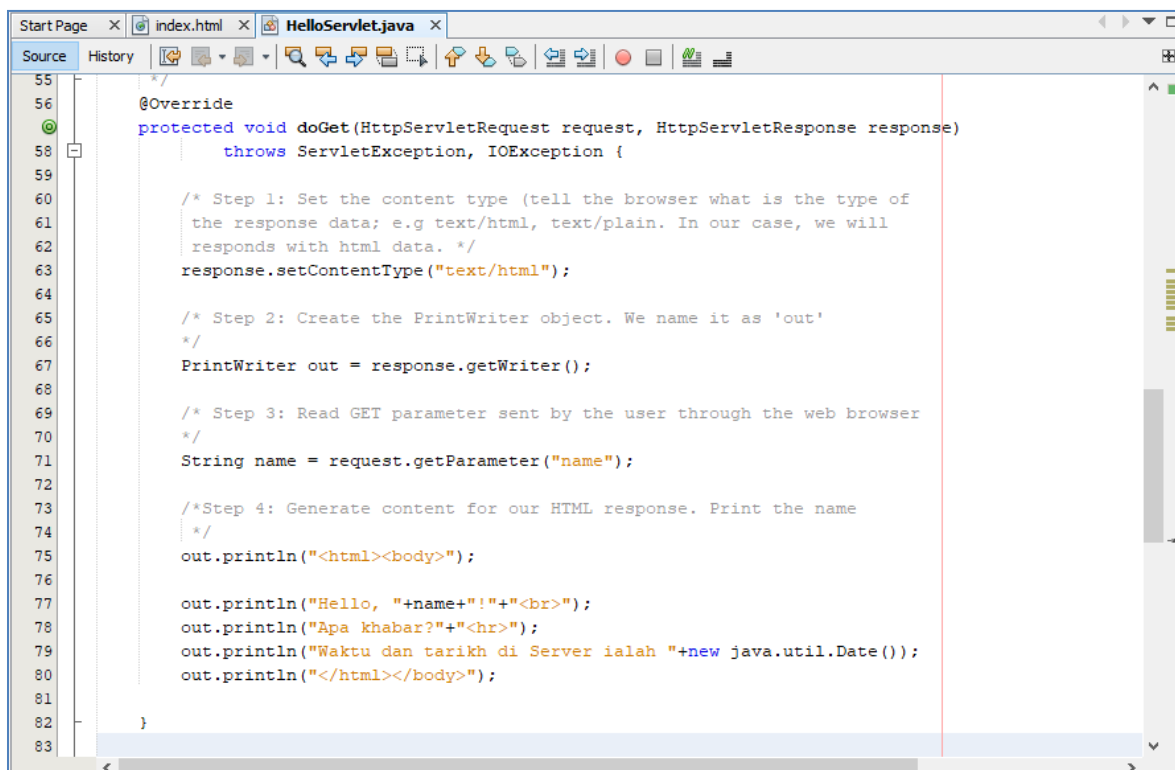
12. Before you can see the output, a dialogue box will appear. This dialogue box allows us to add some request parameter. At this moment, we will not supply any parameter. Just click OK button.



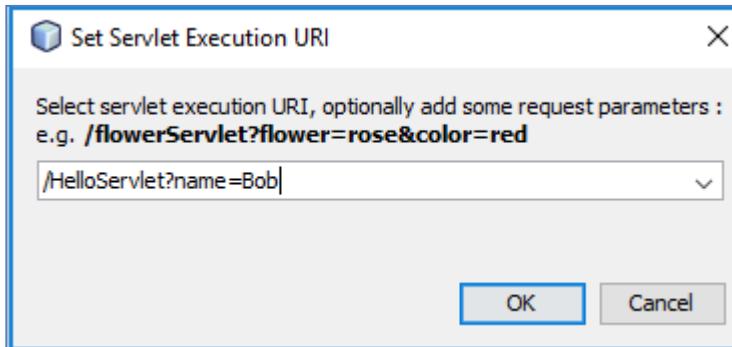
13. You will see the following output on your browser.



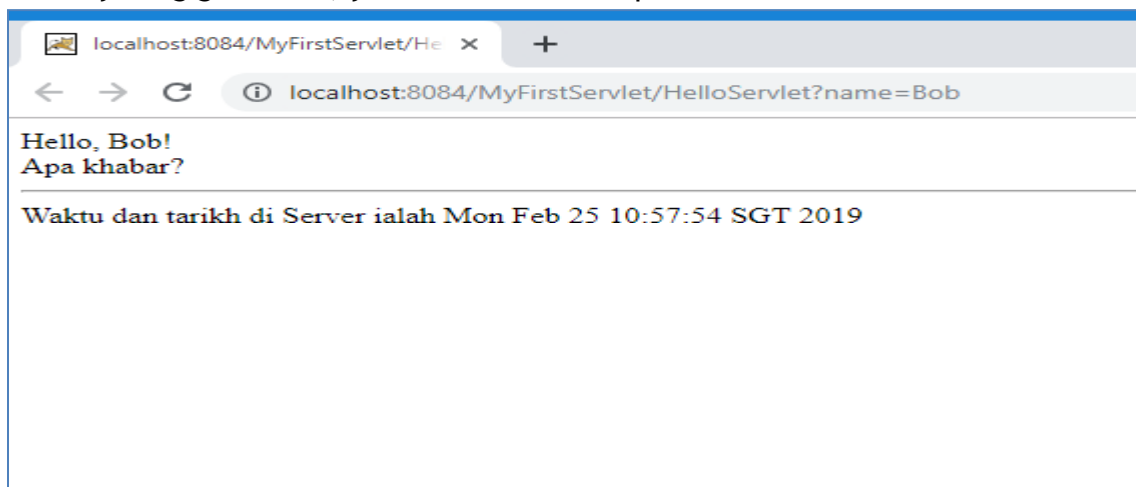
14. Now, we will make further modification to our servlet file. This time, we are going to modify the `doGet()` method. Browse through the codes in *HelloServlet.java* until you find the desired method. Type the codes as provided in the screenshot below. Read the comments carefully and try to understand what each of the codes does. **Remark: Please remove `processRequest()` method from the body of `doGet()` method.**



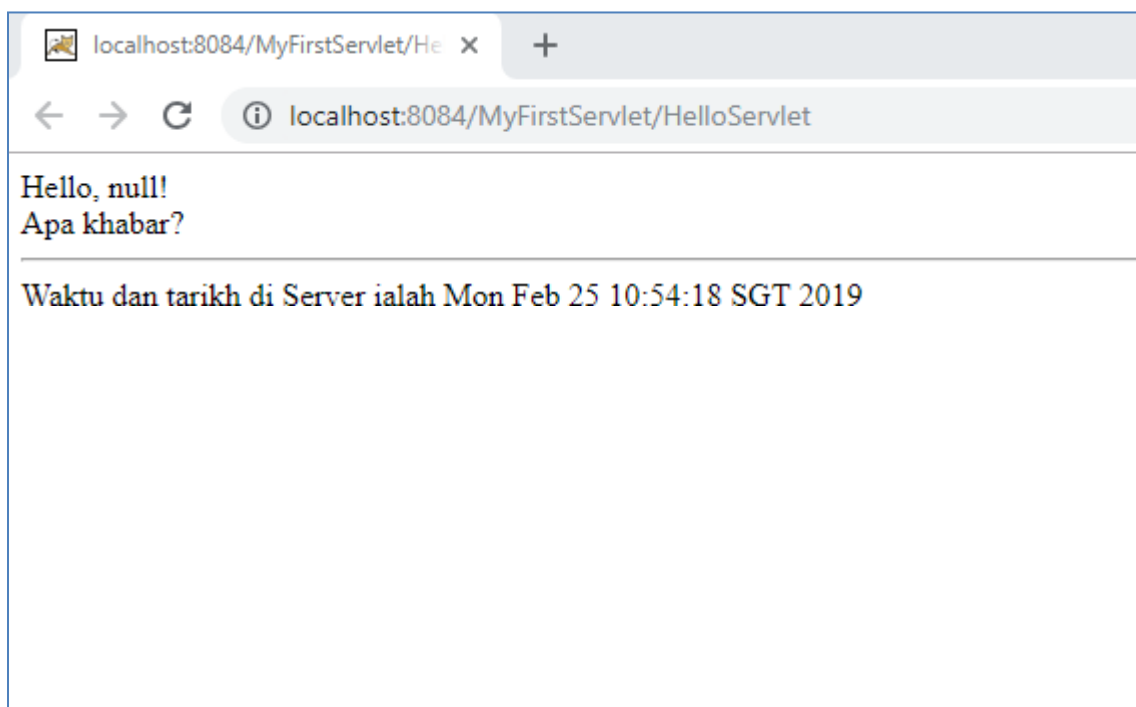
15. After finish, right click on *HelloServlet.java* and click Run. As you have seen previously, a dialogue box shows. This time, we will supply a value *Bob* to the parameter *name*. Then, click the OK button.



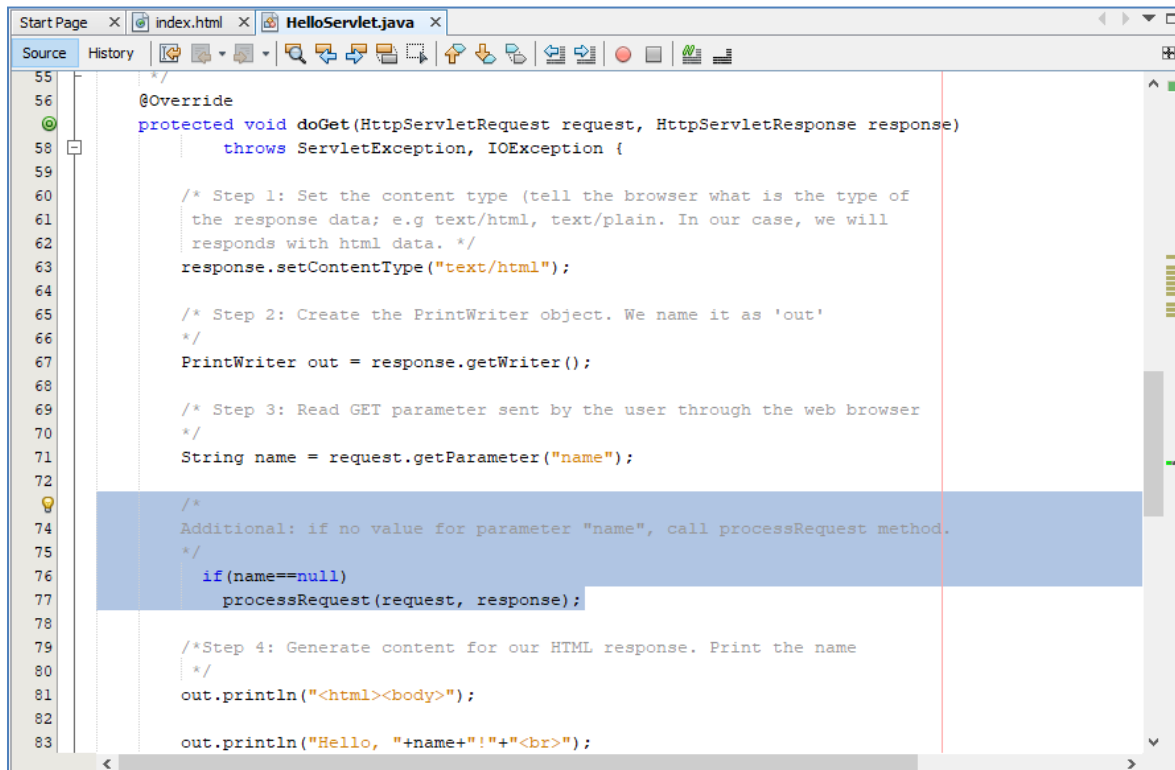
16. If everything goes well, you will see the output as below:



17. Rerun **step 20** again, this time do not supply any request parameter. What do you see from the output? Do you see something like the following screenshot? How to avoid the *null* from being displayed?

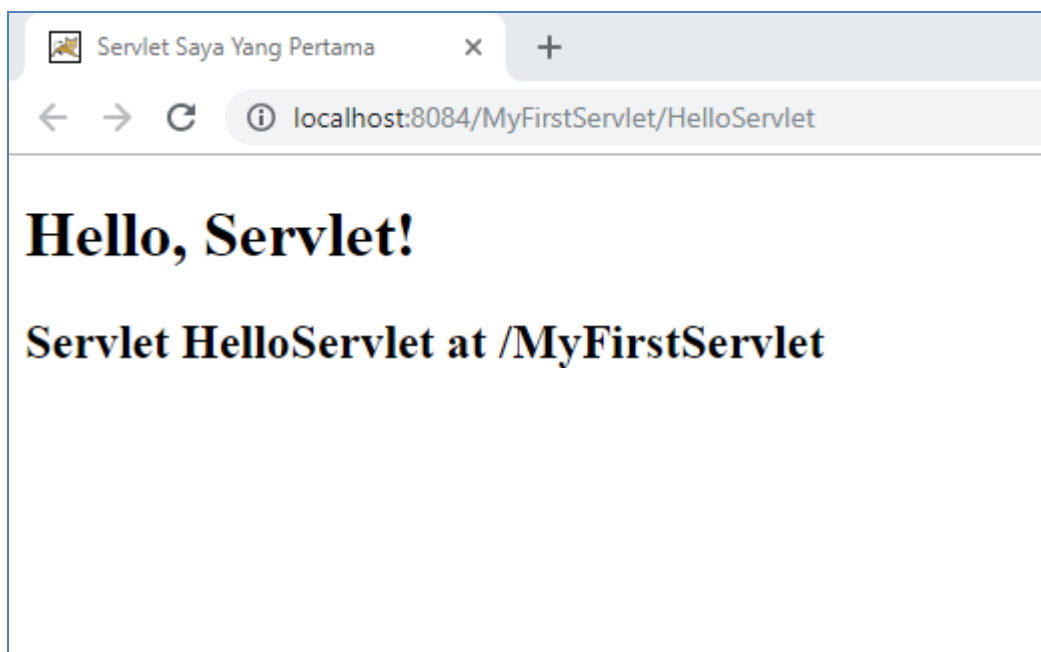


18. You can upgrade your code in `HelloServlet.java` by putting the following codes into it. By doing this, if no value supplied to the parameter *name*, the request will be passed to *processRequest()* method, and this will avoid from the *null* value appears on the browser.



```
55  */
56  @Override
57  protected void doGet(HttpServletRequest request, HttpServletResponse response)
58      throws ServletException, IOException {
59
60      /* Step 1: Set the content type (tell the browser what is the type of
61       the response data; e.g text/html, text/plain. In our case, we will
62       responds with html data. */
63      response.setContentType("text/html");
64
65      /* Step 2: Create the PrintWriter object. We name it as 'out'
66       */
67      PrintWriter out = response.getWriter();
68
69      /* Step 3: Read GET parameter sent by the user through the web browser
70       */
71      String name = request.getParameter("name");
72
73      /*
74       Additional: if no value for parameter "name", call processRequest method.
75       */
76      if(name==null)
77          processRequest(request, response);
78
79      /*Step 4: Generate content for our HTML response. Print the name
80       */
81      out.println("<html><body>");
82
83      out.println("Hello, " + name + "!" + "<br>");
```

19. So, if you rerun the file and without supplying any parameter, you will see the output as follows:



It is the same output as can be seen in **Step 18**: why?

Display output not same because name is null, so NullPointerException will be throw.

Coding:

```
6  import java.io.IOException;
7  import java.io.PrintWriter;
8  import jakarta.servlet.ServletException;
9  import jakarta.servlet.http.HttpServlet;
10 import jakarta.servlet.http.HttpServletRequest;
11 import jakarta.servlet.http.HttpServletResponse;
12
13
14 /**
15  *
16  * @author Fad Rahmat
17  */
18 public class HelloServlet extends HttpServlet {
19
20     /**
21      * Processes requests for both HTTP GET and POST
22      * methods.
23      *
24      * @param request servlet request
25      * @param response servlet response
26      * @throws ServletException if a servlet-specific error occurs
27      * @throws IOException if an I/O error occurs
28      */
29 }
```

```
30     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
31     {
32         throws ServletException, IOException {
33             response.setContentType("text/html;charset=UTF-8");
34             try (PrintWriter out = response.getWriter()) {
35                 /* TODO output your page here. You may use following sample code. */
36                 out.println("<DOCTYPE html>");
37                 out.println("<html>");
38                 out.println("<head>");
39                 out.println("<title>Servlet Page Test Param</title>");
40                 out.println("</head>");
41                 out.println("<body>");
42                 out.println("<h1>Hello, Servlet</h1>");
43                 out.println("<h2>Servlet HelloServlet at " + request.getContextPath() + "</h2>");
44                 out.println("</body>");
45                 out.println("</html>");
46             }
47         }
48     }
```

```
49
50     @Override
51     protected void doGet(HttpServletRequest request, HttpServletResponse response)
52     {
53         throws ServletException, IOException {
54             response.setContentType("text/html");
55             PrintWriter out = response.getWriter();
56             String name = request.getParameter("name");
57             if(name==null)
58                 processRequest(request, response);
59             out.println("<html><body>");
60             out.println("Hello, " + name + "!<br>");
61             out.println("Age Rahmat" + "<br>");
62             out.println("Waktu dan tanggal di Servlet ialah " + new java.util.Date());
63             out.println("</html></body>");
64         }
65     }
66
67
68
69
70
71 }
```

Output:

Hello, Fad!
Apa khabar?

Waktu dan tarikh di Servlet ialah Sun Mar 31 00:13:27 SGT 2024

Task 6: Writing a Simple JSP Program

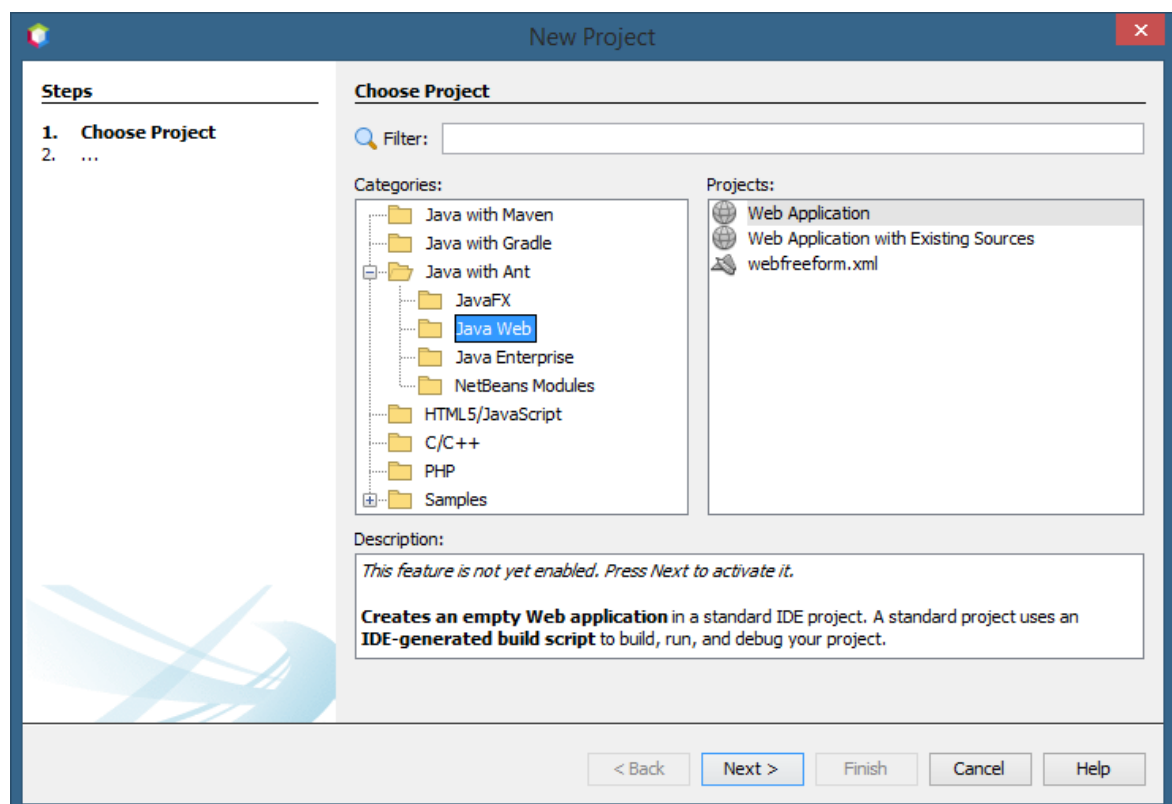
Objective : Writing a simple plain JSP program

Problem

Description : Write a simple plain JSP program to display
“Welcome to [MATRICNUMBER]..!”

Estimated time : 15 minutes

1. Create a directory C:\[MatricNumber] Lab
2. Go to C:\ [MatricNumber] Lab's directory and create sub-directory as Lab 1 - JSP.
3. Open your NetBeans.
4. Go to File -> New Project
5. Select Select Java with Ant -> Java Web -> Web Application and click Next.



6. Click the Next button.

7. Type Project Name: *Lab1*.

8. Choose Project Location: *C:\[MATRICNUMBER]\Lab 1*.

New Web Application

Steps

1. Choose Project
- 2. Name and Location**
3. Server and Settings
4. Frameworks

Name and Location

Project Name: Lab 1

Project Location: E:\CSF3107 Lab Browse...

Project Folder: E:\CSF3107 Lab\Lab 1

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder: Browse...

Different users and projects can share the same compilation libraries
(see Help for details).

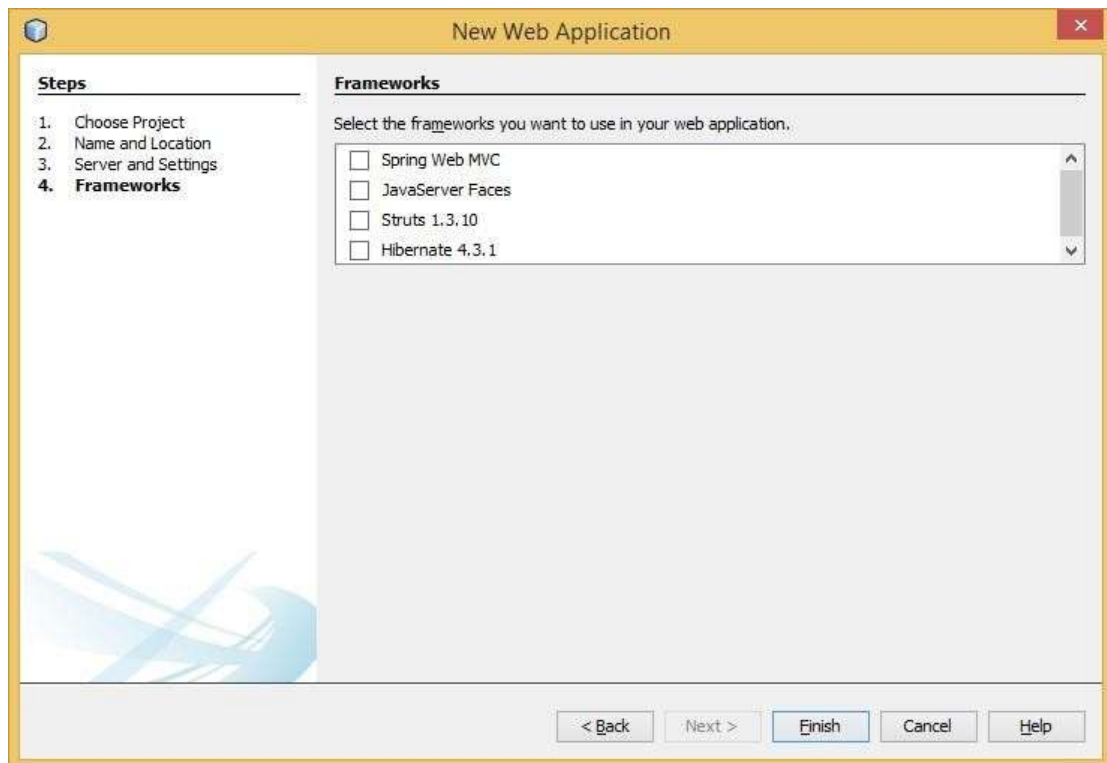
< Back Next > Finish Cancel Help

9. Click the *Next* button.

10. Select Server: *Apache Tomcat or TomEE*

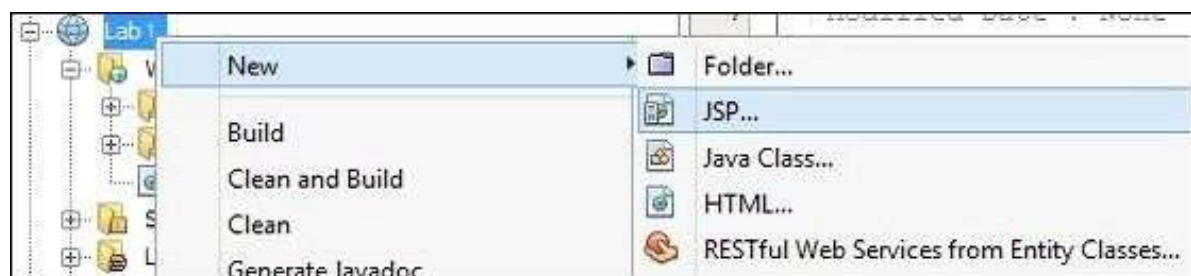
11. Select Java EE Version: *Java EE 6 Web*.

12. Click the Next button.

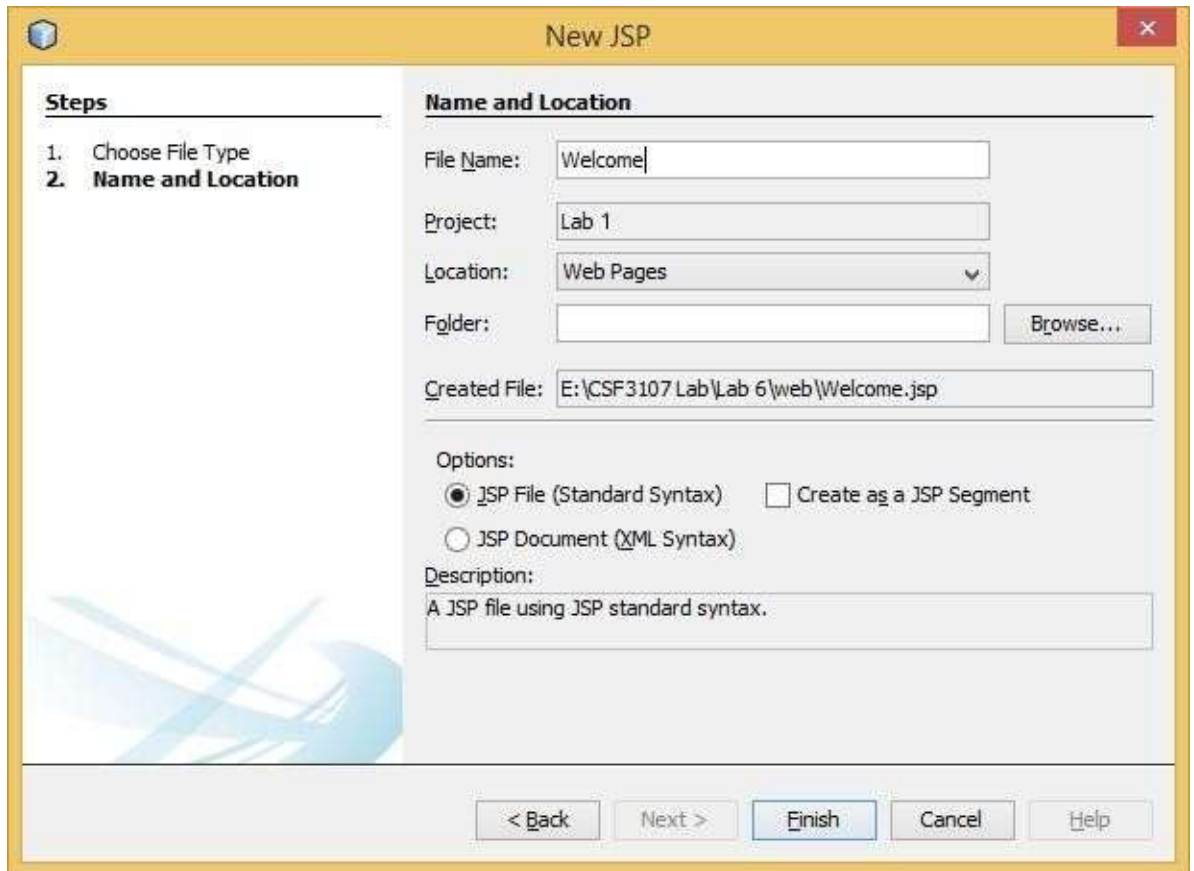


13. Click the Finish button.

14. Create a new JSP's file.



15. Type file name as *Welcome*.



16. Click the *Finish* button.

17. Type title as *[MATRICNUMBER] - Web Programming 2*

18. Type header1 as *Welcome to [MATRICNUMBER]...!*.

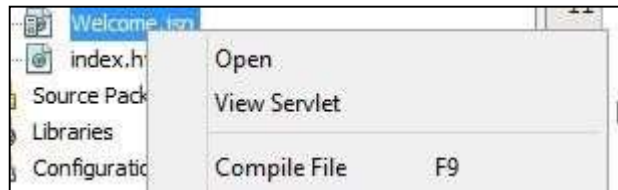


```
1 <%--
2     Document    : Welcome.jsp
3     Created on  : 29-Mar-2016, 09:46:05
4     Author     : Mohamad Nor Hassan
5 --%>
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12     <title>CSF3107 - Web Programming 2</title>
13 </head>
14 <body>
15     <h1>Welcome to CSF3107...!</h1>
16 </body>
17 </html>
```

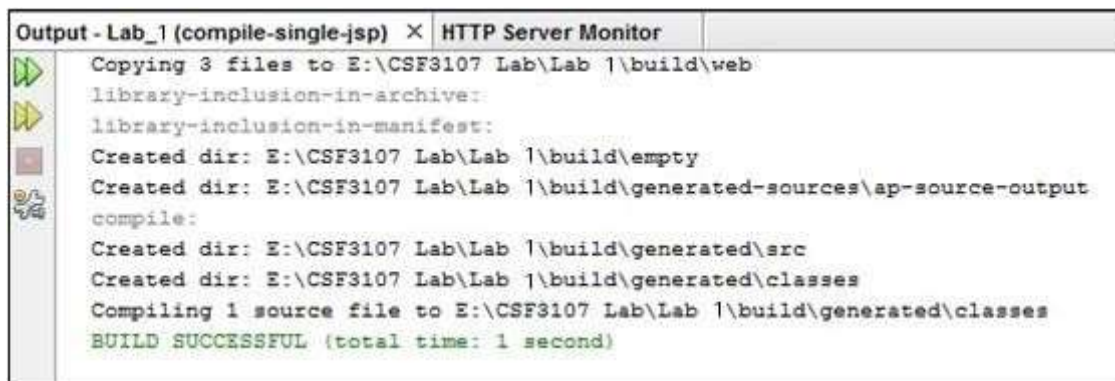
19. Click *SaveAll* icon



20. Right-click file *Welcome.jsp* and click *Compile File* (F9).

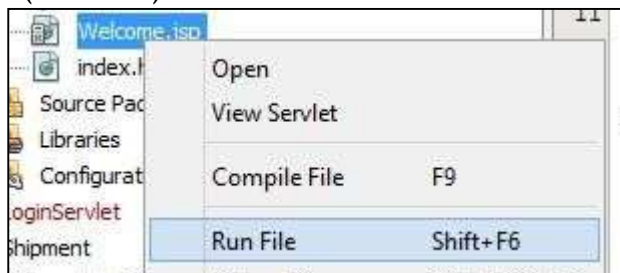


21. You will get notification message the bottom of Netbeans IDE with the green colour.



Note: Before running any JSP's files for the first time upon opening your Netbeans IDE, you need to start your web server (i.e., Apache Tomcat).
Note: Avoid these steps if Apache Tomcat already starts.

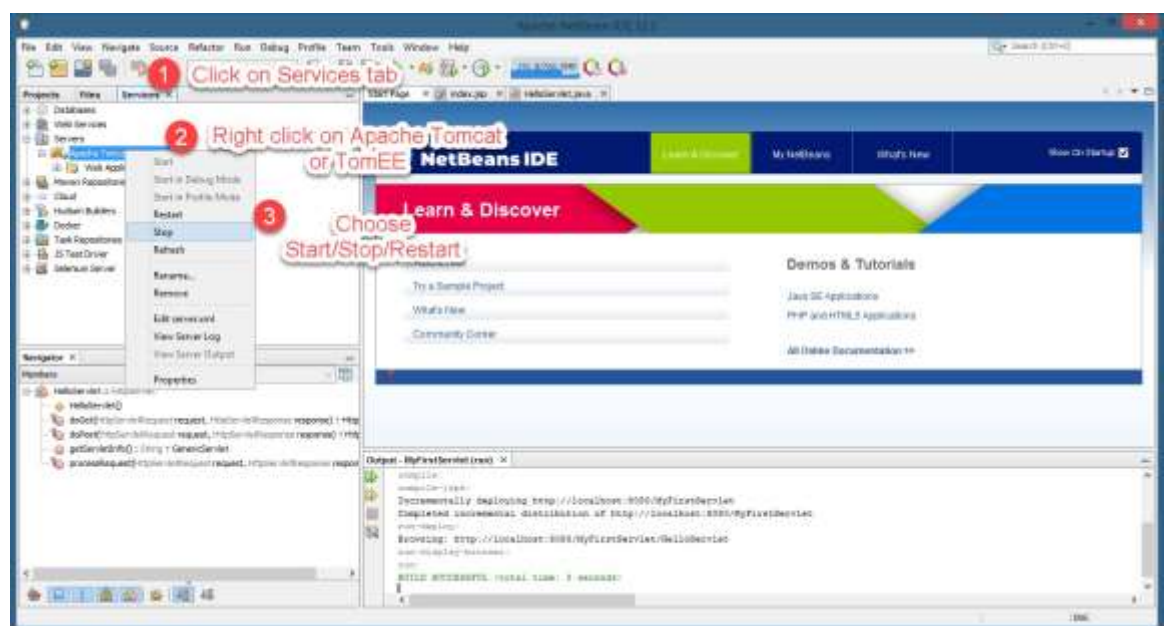
26. Go to the *Project's* tab. Then right click *Welcome.jsp* file and click *Run File* (Shift+F6).



27. The output will appear in a web browser.



Note: Beside using XAMPP to start or stop the Apache Tomcat, you may also do it directly from Netbeans as shown in the figure below.



Reflection

1. What have you learned from this exercise?

JSP is a technology used for developing dynamic web pages in Java.

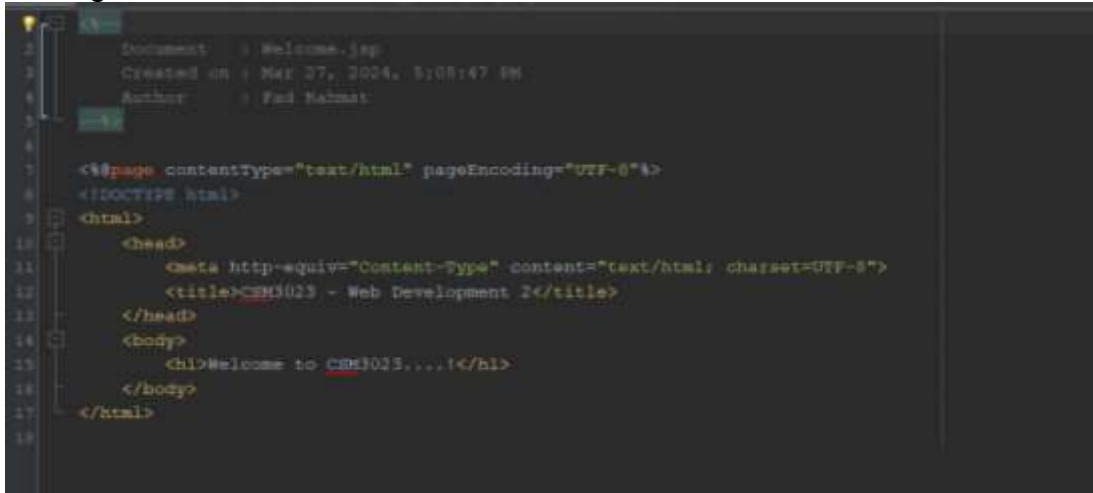
2. Explain the general concept of how the JSP's file work?

JSP files are processed by the web container, which compiles them into Java servlets, executes the embedded Java code, generates dynamic content, and sends the resulting HTML response to the client's web browser.

3. Based on your observation of the previous tasks (Task 3 and Task 4), what are the differences you can find between servlet and JSP?

Servlets primarily used for implementing server-side logic and handling HTTP requests, while JSP is used for creating dynamic web pages with embedded Java code for generating dynamic content.

Coding:



```
Document : Welcome.jsp
Created on : Mar 27, 2024, 5:08:47 PM
Author : Fad Rahmat

<%page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSM3023 - Web Development 2</title>
</head>
<body>
<h1>Welcome to CSM3023....!</h1>
</body>
</html>
```

Output:

Welcome to CSM3023....!

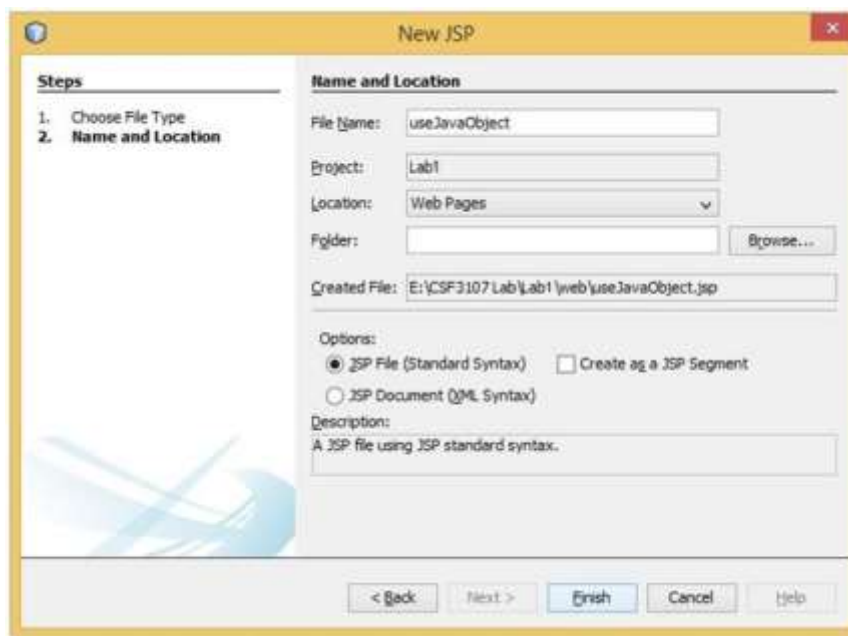
Task 7: Use Java Reference Datatype/Class Wrapper in JSP

Objective : Using Java's object in JSP page.

Problem Description : Display the current date, perform auto refresh header in JSP's page.

Estimated time : 20 minutes

1. Go to project *Lab1*.
2. Create a new JSP's file as *useJavaObject*.



3. Click the *Finish* button.
4. Change the title *Using Java's object in JSP page*.
5. Change the `<h1>` as *Display Current Date and* perform auto refresh header.

6. Add *Java util* package for Date reference.

```
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@page import="java.util.Date.*"%>
9
```

7. Write a Java Scriptlet to create Date's object and display the current date and time.

```
<body>
    <h1>Display Current Date and perform simple Mathematics operations </h1>

    <%
        Date todayDate = new Date();
        out.print("<p>Current date and time is " + todayDate.toString() + "</p>");
    %>

</body>
```

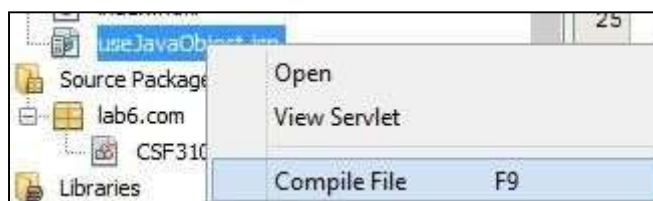
8. Continue writing a code to perform auto refresh header.

```
<%
    // Set refresh, autoload time as 5 seconds
    response.setHeader("Refresh", 5);

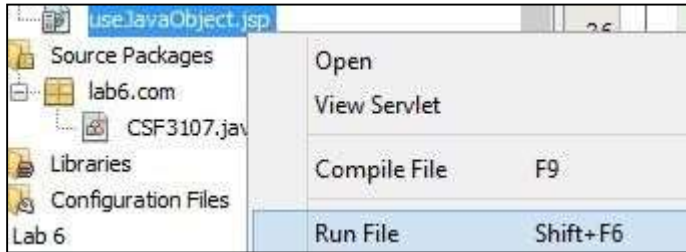
%>
```

9. Save your file.

10. Right-click *useJavaObject.jsp* and compile the program.



11. Finally, right-click *useJavaObject.jsp* and choose Run File to run the program.



12. Review the output display in the browser.



Reflection

1. What have you learnt from this exercise?

`response.setIntHeader("Refresh", 5);`, where `response` is a reference to the `HttpServletResponse` object associated with the current request.

2. What is Java Scriptlet?

is a block of Java code embedded within a JavaServer Pages (JSP) file.

3. How to use Java code in your JSP's page?

embed Java code directly within the HTML or XML markup using scriptlet tags.

Coding:

```

1  <%--
2      Document      : useJavaObject
3      Created on    : Mar 27, 2024, 5:47:56 PM
4      Author       : Fad Rahmat
5  --%>
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <%@page import="java.util.Date"%>
9  <!DOCTYPE html>
10 <html>
11 <head>
12     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13     <title> page</title>
14 </head>
15 <body>
16     <h1>Display current Date and perform simple Mathematics operations</h1>
17
18     <%
19         Date todayDate = new Date();
20         out.print("<p>Current date and time is " + todayDate.toString() + "</p>");
21     %>
22
23     <%
24         response.setIntHeader("Refresh", 5);
25     %>
26 </body>
27 </html>

```

Output:

Display current Date and perform simple Mathematics operations

Current date and time is Sun Mar 31 00:37:52 SGT 2024

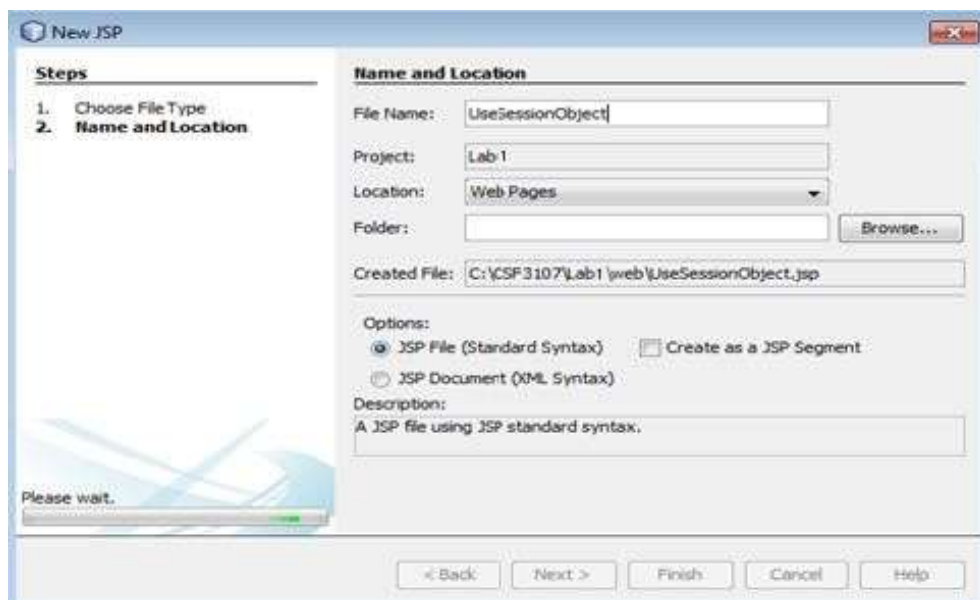
Task 8: Using JSP Implicit object in JSP page

Objective	: Using JSP Implicit object (Session) in JSP page.
Problem Description	: Using Session object, perform simple Mathematics operations in JSP's page.
Estimated time	: 30 minutes

1. Go to Project *Lab1*.
2. To create a JSP's page, right click *Lab1*-> *New* -> *JSP*.



3. Create a new JSP's file as *AttributelsSet*.



4. Click the *Finish* button.
5. Source code for *AttributelsSet.jsp* will appear.
6. Write the code below:

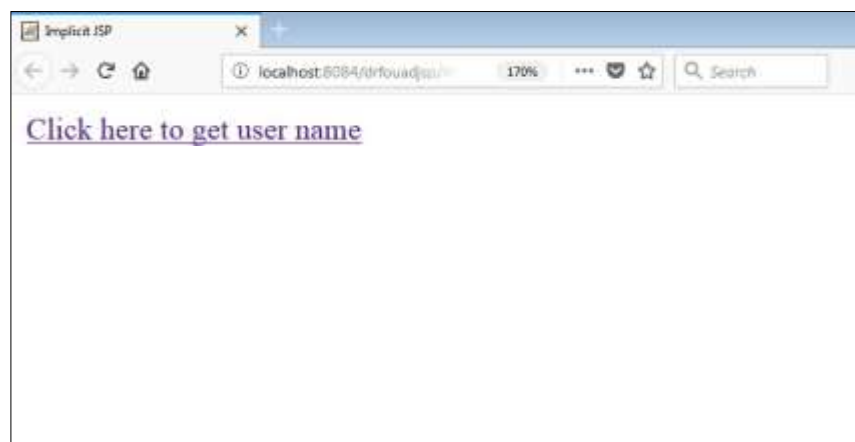
```
<%--
    Document    : jsp
    Created on   : 20-Feb-2018
    Author      : Dr. Faizah Aplop
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Implicit JSP</title>
    </head>
    <body>

        <% session.setAttribute("user", "Fouad Abdulameer");%>
        <a href="GetAttribute.jsp">Click here to get user name </a>

    </body>
</html>
```

9. Save and compile *AttributelsSet.jsp* file.
10. Run the *AttributelsSet.jsp* file, and you should get the interface as below:



11. Repeat step 1 and step 2.
12. Key-in File Name: *GetAttribute*.

13. Click the *Finish* button.

14. Source code for *GetAttribute.jsp* will appear.

15. Write the *GetAttribute* code.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Implicit JSP</title>
  </head>
  <body>

    <%
      String name = (String) session.getAttribute("user");
      out.println("User Name is: " + name);
    %>

  </body>
</html>
```

17. Compile *GetAttribute.jsp* file.

18. Run the *AttributeIsSet.jsp* file and click on the link.

19. Add math package to perform simple Mathematics operation in your page.

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

20. Create a new JSP's file as *MathematicsOperations* to perform addition, multiplication and find the square roots of the number.

```
<%
  int num1 = 25;
  int num2 = 10;
  int addition_output;
  int multiply_output;
  double squareroot = 0.00;

  java.util.Formatter myFormat = new java.util.Formatter();

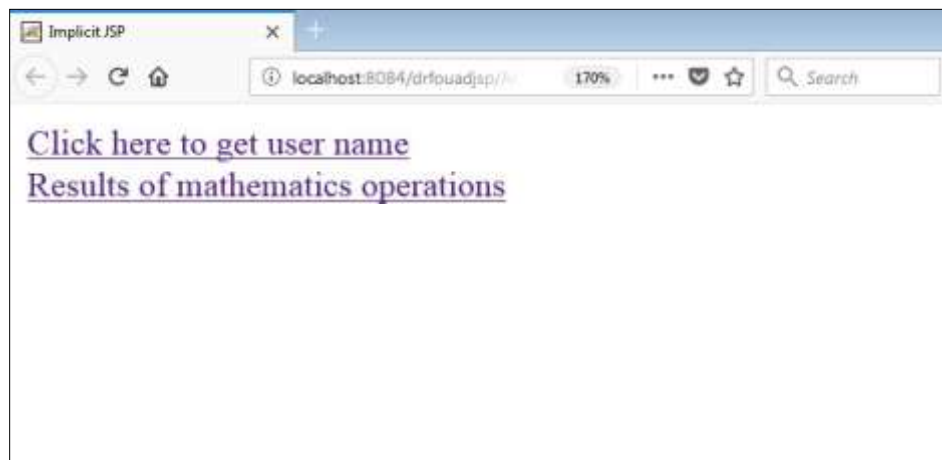
  //perform basic arithmetics operations,,,
  addition_output = num1 + num2;
  multiply_output = num1 * num2;

  //Find square root for variable num1..
  squareroot = (double) (Math.sqrt(num1));

  out.print("<p>Addition num1 and num2 is " + addition_output + "</p>");
  out.print("<p>Multiplication num1 and num2 is " + multiply_output + "</p>");

  out.print("<p></p>");
  out.print("<p>Square root of " + num1 + " is " + myFormat.format("%.2f", squareroot) + "</p>");
%>
```

21. You should get the interface as below:



22. Review the outputs display in the browser.

Reflection

1. How do you want to submit specific information from one form to next form?

store the specific information in session variables when the first form is submitted, and retrieve it from the session in the subsequent form.

2. What happened if the field name you specify in `request.getParameter("field_name")` in the second page is different from the field name you defined in the first page?

the `getParameter()` method will return null because it cannot find a parameter with the specified name.

Coding:

AttributelsSet.:

```

1 <!--
2 Document : AttributeIsSet.
3 Created on : Mar 30, 2024, 11:08:03 AM
4 Author : Fad Rahmat
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>implicit JSP</title>
13 </head>
14 <body>
15 <% session.setAttribute("user", "Fouad Abdulameer");%>
16 <a href="GetAttribute.jsp">Click here to get user name </a>
17 </body>
18 </html>
19

```

GetAttribute.:

```

1 <!--
2 Document : GetAttribute.
3 Created on : Mar 30, 2024, 11:17:18 PM
4 Author : Fad Rahmat
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>implicit JSP</title>
13 </head>
14 <body>
15 <%
16     String name = (String) session.getAttribute("user");
17     out.println("User Name is: " + name);
18 %>
19 </body>
20 </html>
21

```

MathematicsOperations:

```

1 <!--
2 Document : MathematicsOperations
3 Created on : Mar 31, 2024, 11:08:39 AM
4 Author : Fad Rahmat
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@page import="java.math.*"%>
9 <!DOCTYPE html>
10 <html>
11 <head>
12 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13 <title>implicit JSP</title>
14 </head>
15 <body>
16

```

```

13  <%
14      int num1 = 25,
15          num2 = 10,
16          addition_output,
17          multiply_output;
18      double squareroot = 0.00;
19
20      java.util.Formatter myFormat = new java.util.Formatter();
21
22      // Define basic arithmetic operations
23      addition_output = num1 + num2;
24      multiply_output = num1 * num2;
25
26      // Find square root for variable num1
27      squareroot = (double)(Math.sqrt(num1));
28
29      out.print("Addition num1 and num2 is " + addition_output + "<br/>");
30      out.print("Multiplication num1 and num2 is " + multiply_output + "<br/>");
31      out.print("Square root of " + num1 + " is " + myFormat.format("%.2f", squareroot) + "<br/>");
32
33  %>
34
35  </body>
36  </html>
37

```

Output:

[Click here to get user name](#)

[Click here to get result of Mathematics Operations](#)

User Name is: Fouad Abdulameer

Addition num1 and num2 is 35.

Multiplication num1 and num2 is 250.

Square root of 25 is 5.00.

Task 9: Populate Array values into HTML's Table

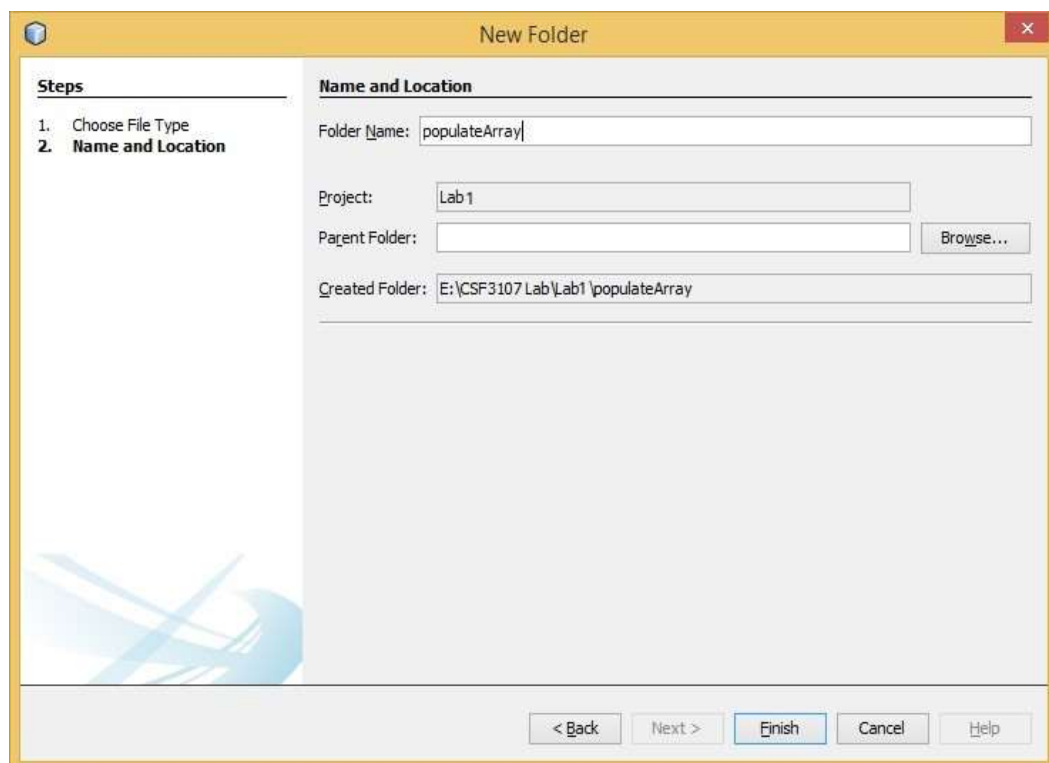
Objective : Read Java array and populate it into HTML's table.

Problem : i. Create a 2D array that store sales data.

Description ii. Then, read an array and populate into HTML's table.

Estimated time : 50 minutes

1. Go to Project *Lab1*.
2. To create a JSP's page, right click *Lab1* -> *New* -> *JSP*.
3. Key-in File Name: *populateArray*.



4. Click the *Finish* button.

5. Prepare standard HTML's markup for page *populateArray.jsp*.
6. Write a Java Scriplet and store the following information into an array;

	Jan	Feb	Mac
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

7. Read the array and populate its value into HTML's table.
8. Save and compile *populateArray.jsp* file.
9. Run the *populateArray.jsp* file and sample of output shows as below:



Salesman	Jan	Feb	Mac
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

©2016-Mohamad Nor

Reflection

1. Write a sample syntax to declare 2D Java array.

```
int[][] twoDArray = new int[3][4];
```

2. Define a sequence of steps on how you accomplish Task 7.

```
<%
List<String[]> salesmans = new ArrayList<>();
salesmans.add(new String[]{"Salesman 1", "2500", "2100", "2200"});
salesmans.add(new String[]{"Salesman 2", "2000", "1900", "2400"});
salesmans.add(new String[]{"Salesman 3", "1800", "2200", "2450"});%>

<% for (String[] salesman : salesmans) { %>
    <tr>
        <td><%= salesman[0] %></td>
        <td><%= salesman[1] %></td>
        <td><%= salesman[2] %></td>
```



```

        <td><%= salesman[3] %></td>
    </tr>
<% } %>

```

3. What is the difference between HTML's page and JSP's page?

HTML pages are used for presenting static content to users and are processed entirely on the client-side, while JSP pages allow for the generation of dynamic content and server-side processing of data and logic through the integration of Java code.

Coding:

```

1  <!--
2      Document   : populateArray
3      Created on : Mar 31, 2024, 1:15:59 AM
4      Author    : Fad Rahmat
5  -->
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10 <head>
11     <meta charset="UTF-8">
12     <title>populate Array</title>
13     <style>
14         table {
15             border-collapse: collapse;
16             width: 50%;
17             margin: 20px auto;
18             float: left;
19
20             th, td {
21                 padding: 10px;
22                 border: 1px solid #ddd;
23                 text-align: left;
24             }
25             th {
26                 background-color: #f2f2f2;
27             }
28         }
29     </style>
30 </head>
31 <body>

```

```

31 <h1>Read Java array and populate it into HTML's table</h1>
32
33 <%@ page import="java.util.ArrayList" %>
34 <%@ page import="java.util.List" %>
35 <%@ page import="java.util.Arrays" %>
36
37 <%
38 List<String[]> salesmans = new ArrayList<>();
39 salesmans.add(new String[]{"Salesman 1", "2500", "2100", "2200"});
40 salesmans.add(new String[]{"Salesman 2", "2000", "1900", "2400"});
41 salesmans.add(new String[]{"Salesman 3", "1800", "2200", "2450"});
42 %>
43
44 <table border="1">
45 <tr>
46 <th>Salesman</th>
47 <th>Jan</th>
48 <th>Feb</th>
49 <th>Mar</th>
50 </tr>
51 <% for (String[] salesman : salesmans) { %>
52 <tr>
53 <td><%= salesman[0] %></td>
54 <td><%= salesman[1] %></td>
55 <td><%= salesman[2] %></td>
56 <td><%= salesman[3] %></td>
57 </tr>
58 <% } %>
59 </table>
60
61 </body>
62 </html>

```

Output:

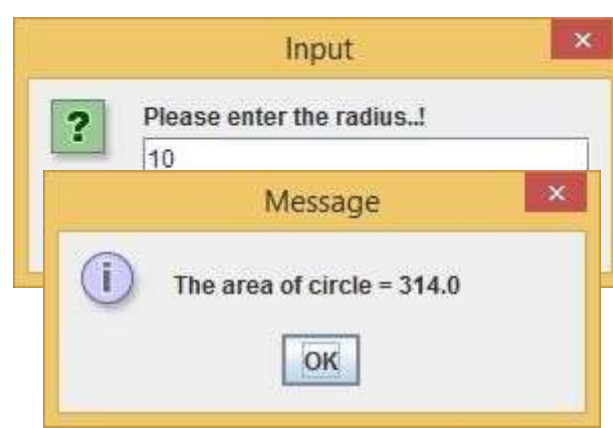
localhost:8080/Lab1/populateArray.jsp

Read Java array and populate it into HTML's table

Salesman	Jan	Feb	Mar
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

Exercise

1. The following program is developed using Java Standard Edition.



Convert this program into Web-Based application where the user can dynamically key-in radius and submit the request to get the area of a circle. You must ensure to accept only number as a radius.

Coding:

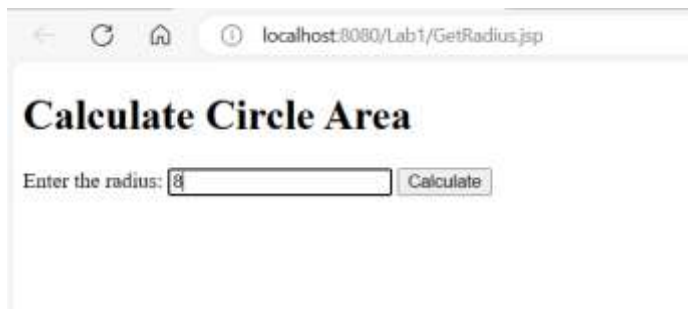
Get radius:

```
1  <!--
2  Document : GetRadius
3  Created on : Mar 31, 2024, 2:04:43 AM
4  Author : Padi Rahmat
5  -->
6
7  <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10 <head>
11 <meta charset="UTF-8">
12 <title>Calculate Circle Area</title>
13 </head>
14 <body>
15 <h1>Calculate Circle Area</h1>
16 <form method="post" action="calculateArea.jsp">
17 <label for="radius">Enter the radius:</label>
18 <input type="text" id="radius" name="radius" pattern="[0-9]*)" title="Please enter a number" required>
19 <button type="submit">Calculate</button>
20 </form>
21 </body>
22 </html>
```

Calculate area:

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2  <%@ page import="java.util.Scanner" %>
3  <%@ page import="java.text.DecimalFormat" %>
4  <!DOCTYPE html>
5  <html>
6  <head>
7  <meta charset="UTF-8">
8  <title>Area Result</title>
9  </head>
10 <body>
11 <h1>Area Result</h1>
12 <!-- Retrieve the radius value from the form -->
13 <% String radiusStr = request.getParameter("radius"); %>
14
15 <!-- Check if the radius value is provided and is a valid number -->
16 <% if (radiusStr != null && radiusStr.matches("\\d+")) { %>
17 <!-- Convert the radius string to an integer -->
18 <% int radius = Integer.parseInt(radiusStr); %>
19
20 <!-- Calculate the area -->
21 <% double area = Math.PI * radius * radius; %>
22
23 <!-- Format the area value -->
24 <% DecimalFormat df = new DecimalFormat("#.##"); %>
25
26 <!-- Display the result -->
27 <p>The area of the circle with radius <%= radius %> is <%= df.format(area) %>.</p>
28 <% } else { %>
29 <!-- Display an error message if the radius value is missing or not a valid number -->
30 <p>Please enter a valid number for the radius.</p>
31 <% } %>
32 </body>
33 </html>
```

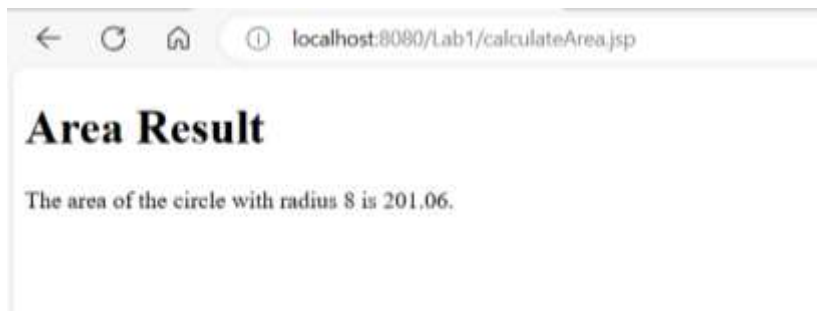
Output:



localhost:8080/Lab1/GetRadius.jsp

Calculate Circle Area

Enter the radius:

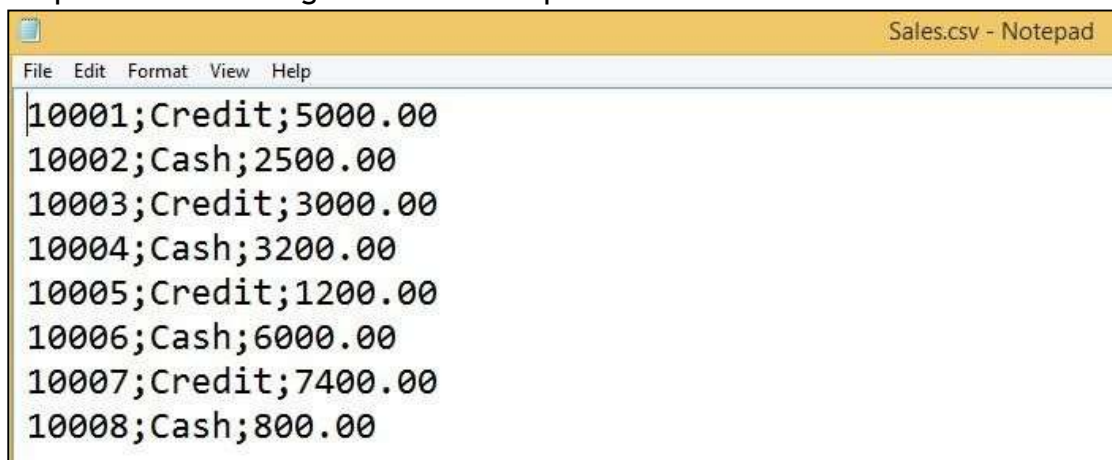


localhost:8080/Lab1/calculateArea.jsp

Area Result

The area of the circle with radius 8 is 201.06.

2. Prepare the following file in the Notepad and save it as *Sales.csv*.



Sales.csv - Notepad

File Edit Format View Help

```
10001;Credit;5000.00
10002;Cash;2500.00
10003;Credit;3000.00
10004;Cash;3200.00
10005;Credit;1200.00
10006;Cash;6000.00
10007;Credit;7400.00
10008;Cash;800.00
```

Create a simple JSP's page to read Sales.csv file. Upon reading Sales.csv's file, calculate the discount and populate HTML's table in the JSP's page.

Customer	Cust. Type	Purchase	Discount
10001	Credit	5000	0.00
10002	Cash	2500	250.00
10003	Credit	3000	0.00
10004	Cash	3200	320.00
10005	Credit	1200	0.00
10006	Cash	6000	600.00
10007	Credit	7400	0.00
10008	Cash	800	80.00

Cash customers are eligible to get a 10% discount.

Coding:

```

1  <!--
2  Document   : Sales.jsp
3  Created on : Mar 11, 2024, 2:15:43 AM
4  Author    : Fad Rahmat
5  -->
6
7  <%page contentType="text/html" pageEncoding="UTF-8"%>
8  <% page import="java.io.*, java.util.*" %>
9  <%(OUTER html)
10 <html>
11 <head>
12 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"%>
13 <title>Sales jsp</title>
14 <style>
15 table, th, td {
16     border: 1px solid black;
17     border-collapse: collapse;
18     padding: 5px 10px;
19     text-align: center;
20 }
21
22 th {
23     background-color: lightgray;
24 }
25 </style>
26 </head>
27 <body>
28
29 <!--Read CSV files and populate it into HTML's table</!%

```

```

30
31 <%
32 // Path to the CSV file
33 String filePath = "C:/Users/Fad Rahmat/OneDrive/Desktop/Sales.csv";
34
35 // List to hold the data from the CSV file
36 List<String[]> csvData = new ArrayList<>();
37
38 // Read the CSV file and populate the list
39 try {
40     BufferedReader br = new BufferedReader(new FileReader(filePath));
41     String line;
42     while ((line = br.readLine()) != null) {
43         String[] parts = line.split(",");
44         csvData.add(parts);
45     }
46     br.close();
47 } catch (IOException e) {
48     out.println("Error reading CSV file: " + e.getMessage());
49 }
50 %>

```

```

31 <table>
32 <thead>
33 <tr>
34 <th>Customer</th>
35 <th>Cust. Type</th>
36 <th>Purchase</th>
37 <th>Discount</th>
38 </tr>
39 </thead>
40 <tbody>
41 <% for (int i = 0; i < csvData.size(); i++) { %>
42 <tr>
43 <%
44 String[] rowData = csvData.get(i);
45 String customer = rowData[0];
46 String custType = rowData[1];
47 int purchase = Integer.parseInt(rowData[2]);
48 double discount = 0.0;
49 if (custType.equalsIgnoreCase("Cash")) {
50     discount = purchase * 0.1; // 10% discount for cash customers
51 }
52 %>
53 <td><%= customer %></td>
54 <td><%= custType %></td>
55 <td><%= purchase %></td>
56 <td><%= String.format("%.2f", discount) %></td>
57 </tr>
58 <% } %>
59 </tbody>
60 </table>
61
62 </body>
63 </html>

```

Output:

localhost:8080/Lab1/Sales.csv.jsp

Read CSV files and populate it into HTML's table

Customer	Cust. Type	Purchase	Discount
10001	Credit	5000	0.00
10002	Cash	2500	250.00
10003	Credit	3000	0.00
10004	Cash	3200	320.00
10005	Credit	1200	0.00
10006	Cash	6000	600.00
10007	Credit	7400	0.00
10008	Cash	800	80.00