



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 4 2024/2025

Prepared for:

DR. MOHAMAD NOR HASSAN

Prepared by:

NUR ARINA BINTI ABDUL MALEK (S65361)



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)

Revision History

Revision Date	Previous Revision Date	Summary of Changes	Changes Marked
		First Issue	Mohamad Nor Hassan
		Second Issue	Dr Rabiei Mamat Dr Faizah Aplop Dr Fouad Ts Dr Rosmayati Mohemad Fakhrul Adli Mohd Zaki
21/02/2019		Addition of Revision History, Table of Contents, Formatting Cover Page	Fakhrul Adli Mohd Zaki
24/02/2019	21/02/2019	Updated Task 1, English translation for the instruction, Addition of Java Servlet task. Fix spelling and grammar.	Fakhrul Adli Mohd Zaki
30/3/2021		Added “Troubleshooting Notes: Fixing phpMyAdmin Error” Added new topic “Managing Apache Tomcat” Added new topic “Netbeans 12.3 IDE Installation” Updated existing topic “Link Netbeans” to “Apache Tomcat and Writing a Simple Java Servlet” Updated page numbers	Fakhrul Adli Mohd Zaki

Table of Contents

Task 1: Apache Tomcat and MySQL Installation Using XAMPP	5
Task 2: Change the Default Root Password of MySQL Database	11
Task 3: Managing Apache Tomcat.....	18
Task 4: Netbeans 12.3 IDE Installation	22
Task 5: Linking Netbeans to Apache Tomcat and Writing a Simple Java Servlet	25
Task 6: Writing a Simple JSP Program	38
Task 7: Use Java Reference Datatype/Class Wrapper in JSP.....	44
Task 8: Using JSP Implicit object in JSP page.....	47
Task 9: Populate Array values into HTML's Table	51

Arahan:

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Pusat Pengajian Informatik dan Matematik Gunaan (PPIMG), Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedarkan manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual. Tandakan (/) setiap langkah yang telah selesai dibuat dan tulis kesimpulan bagi setiap aktiviti yang telah selesai dijalankan.

Instruction:

This laboratory manual is for use by the students of the School of Informatics and Applied Mathematics (PPIMG), Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.

Please follow step by step as described in the manual. Tick (/) each step completed and write the conclusions for each completed activity.

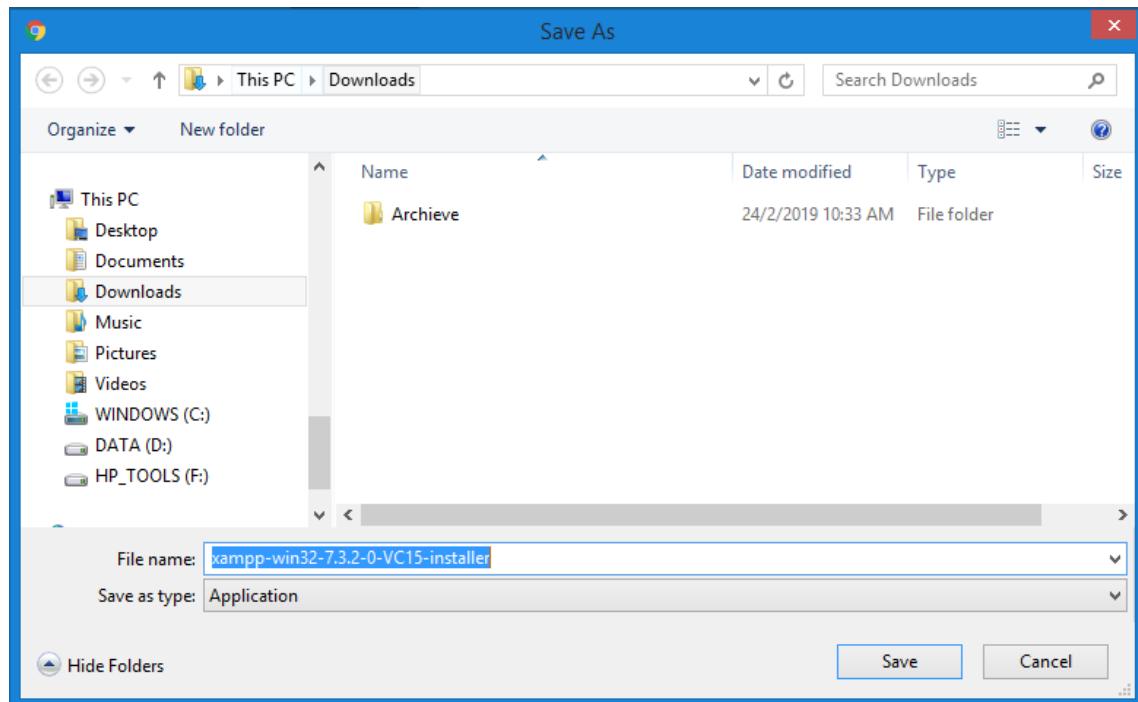
Task 1: Apache Tomcat and MySQL Installation Using XAMPP

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.



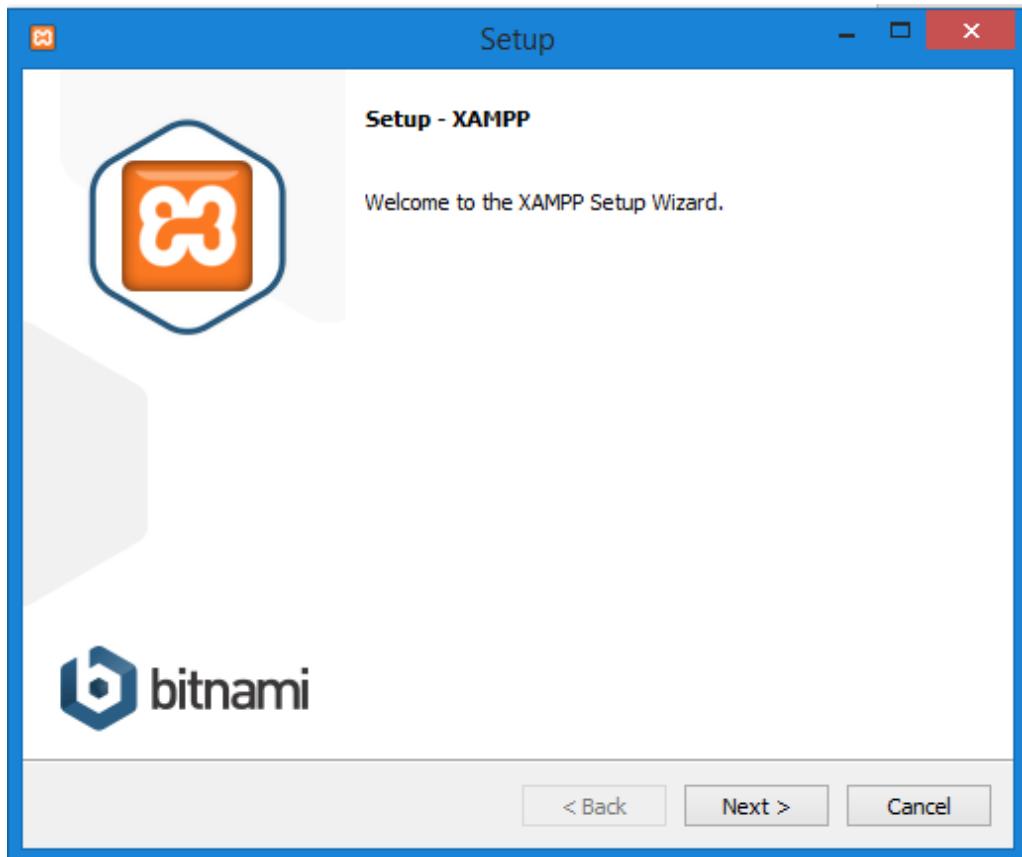
3. Save to a specific directory – for example, the Downloads folder.



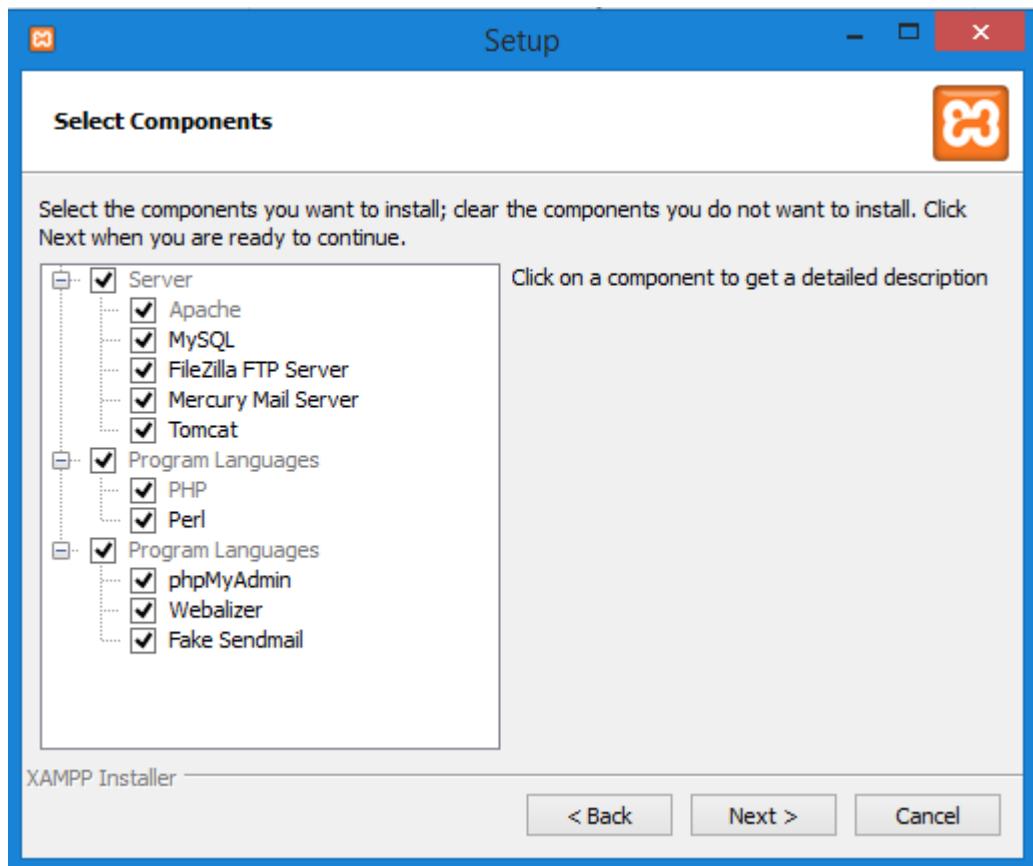
5. Run file xampp-win32-[X.X.X-VCXX]-installer.exe

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.

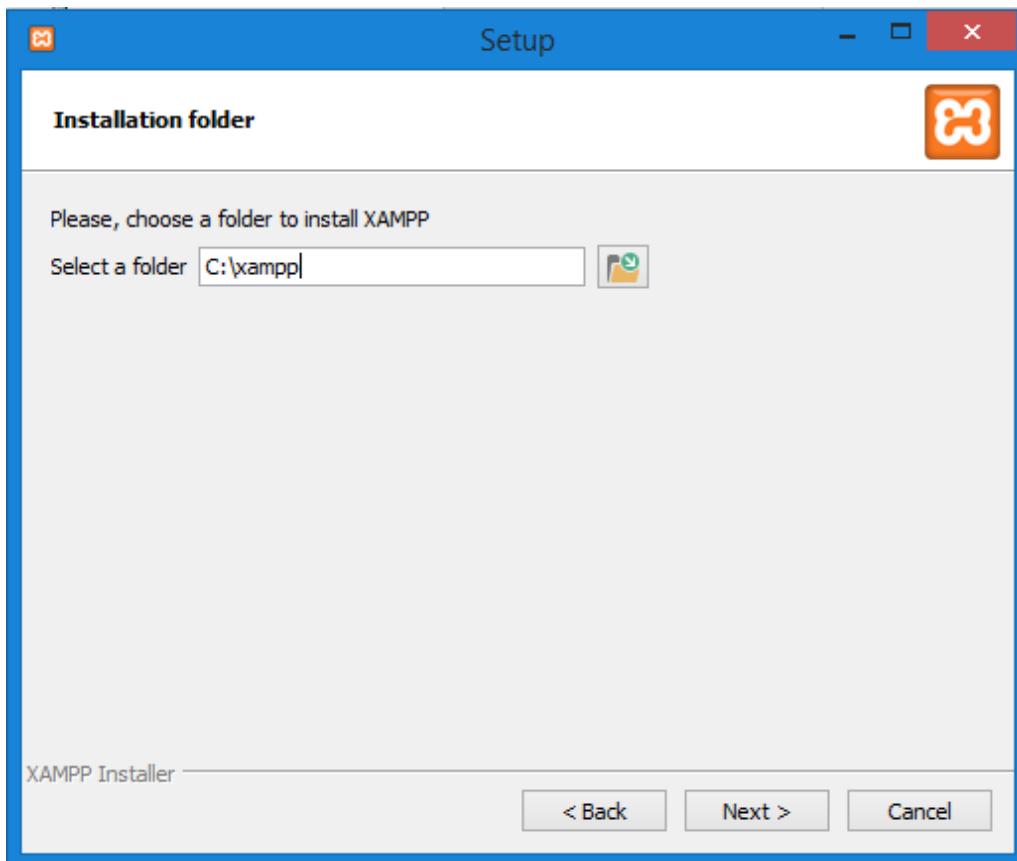
6. Click the Next button.



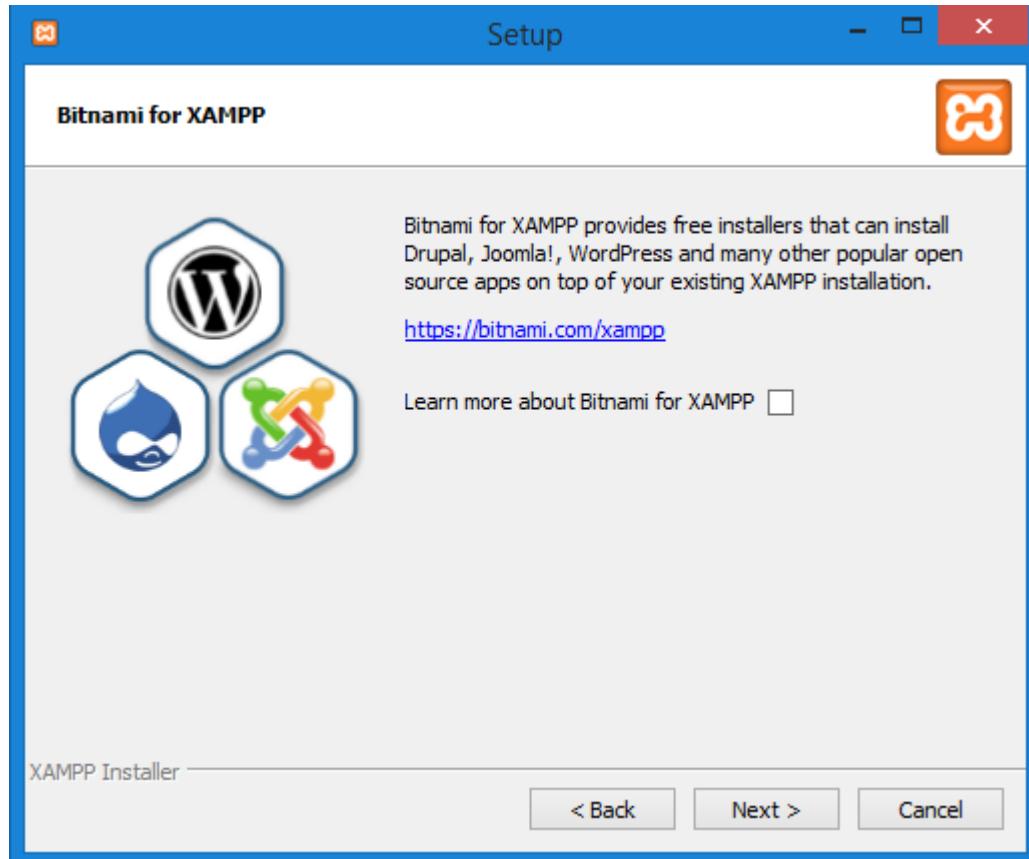
7. Please ensure you choose Tomcat and MySQL.



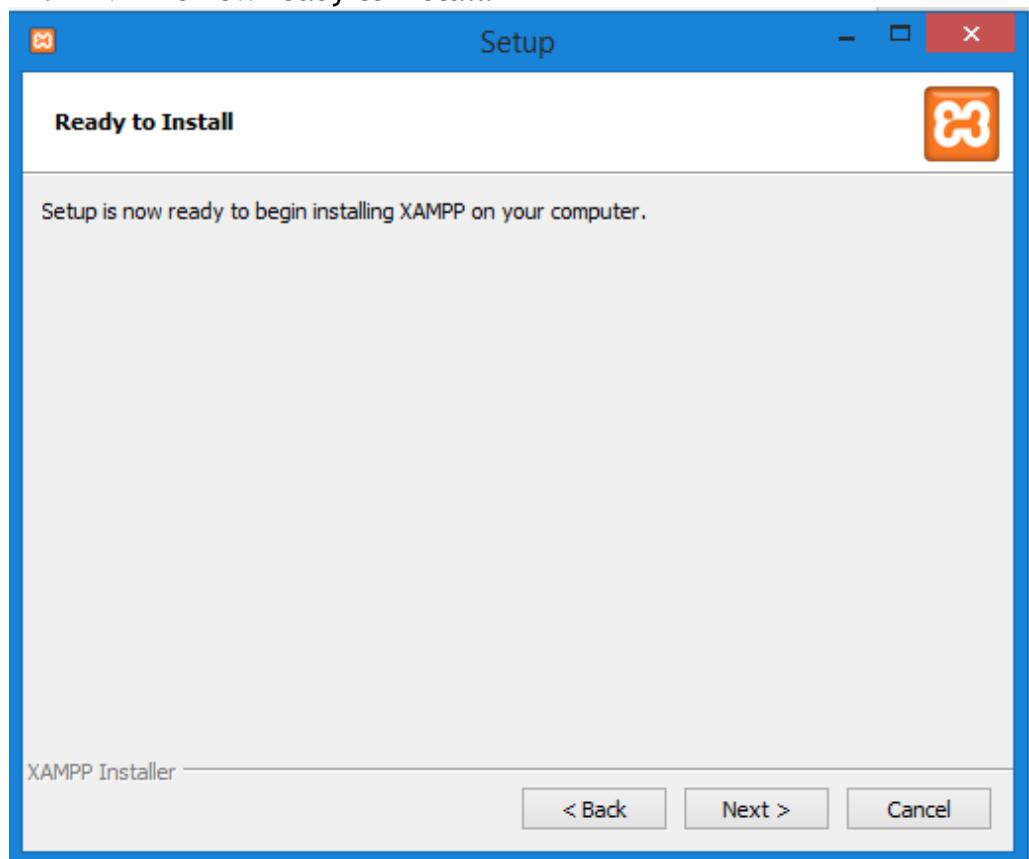
8. Choose a folder as C: \xampp and click the Next > button.



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



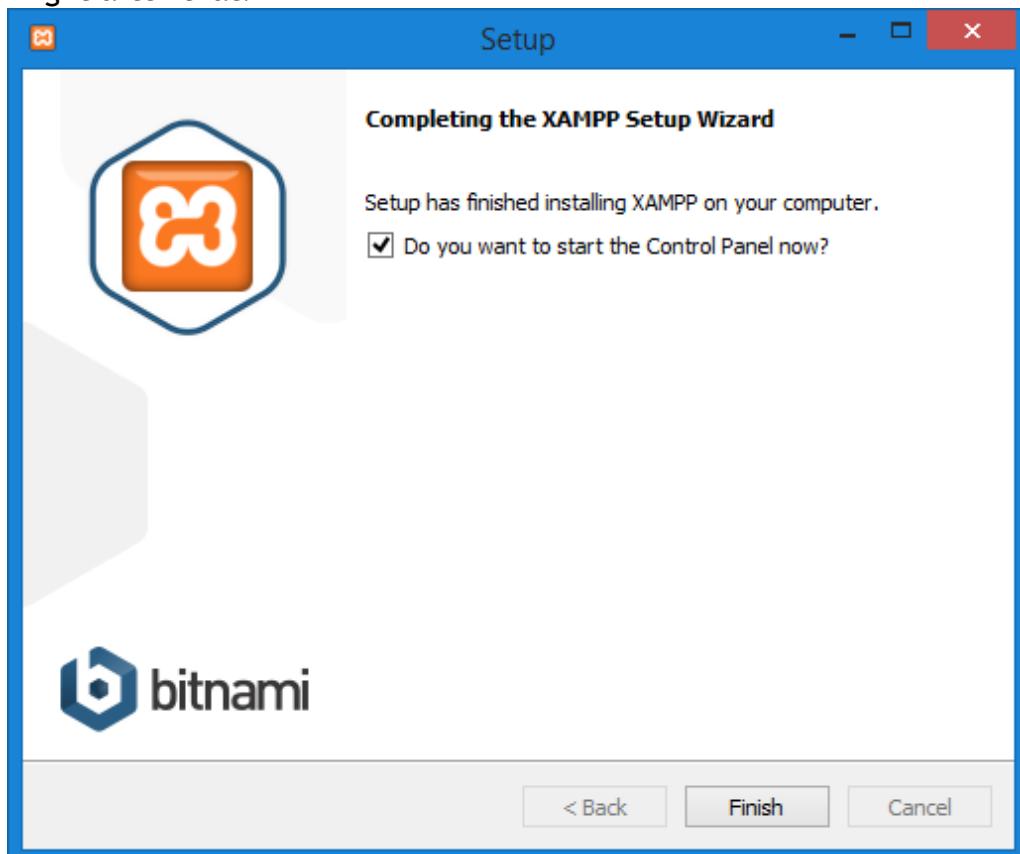
11. XAMPP is now ready to install.



12. Wait for the installation process to complete.



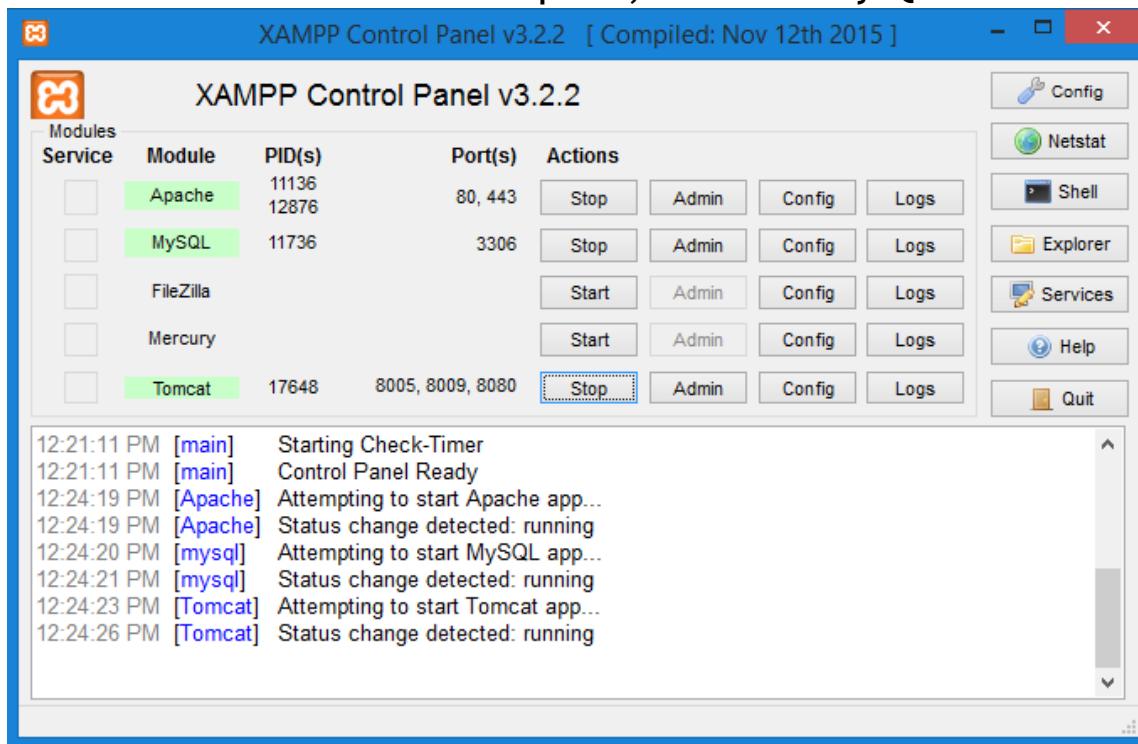
13. Click Finish button once completed and the Control Panel should start right after that.



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?

Task 2: Change the Default Root Password of MySQL Database

Objective:

Change the root password of MySQL Database

Problem

To reset a new root password for MySQL Database

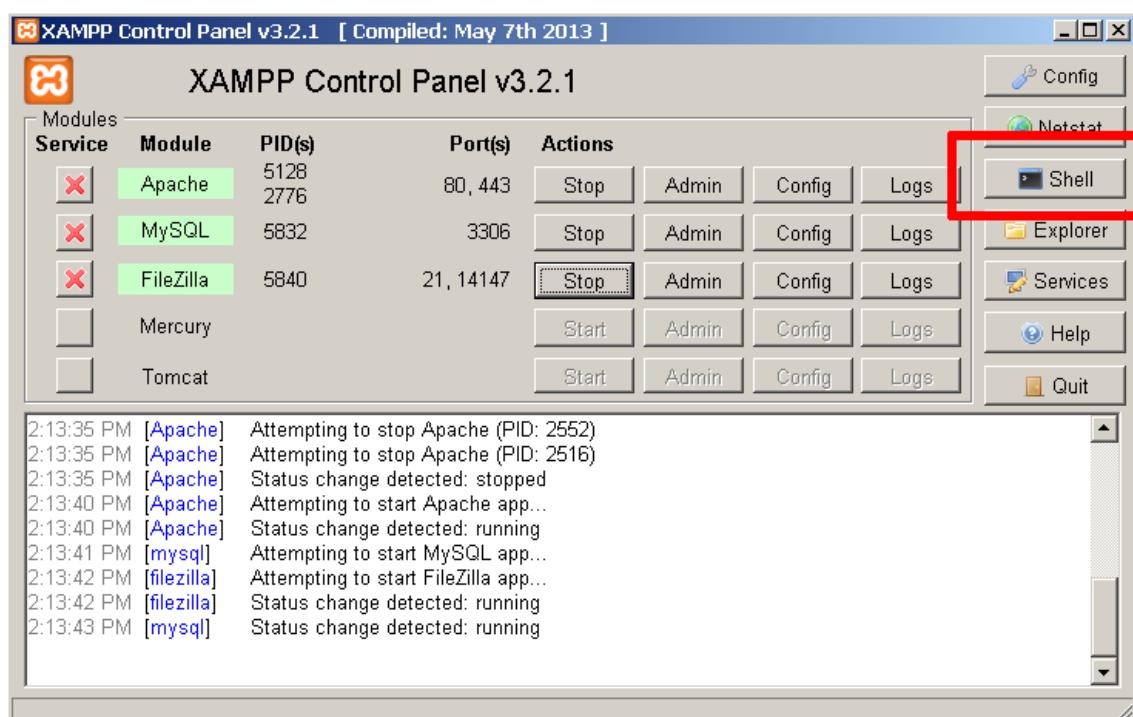
Description:

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"
```

- 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"
```



The screenshot shows a terminal window titled "XAMPP for Windows". The command entered was "mysql --user=root --password=admin -e \"SELECT 1+1\"". The output shows the result of the calculation: 2.

```
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
#
```

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

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

- 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..!

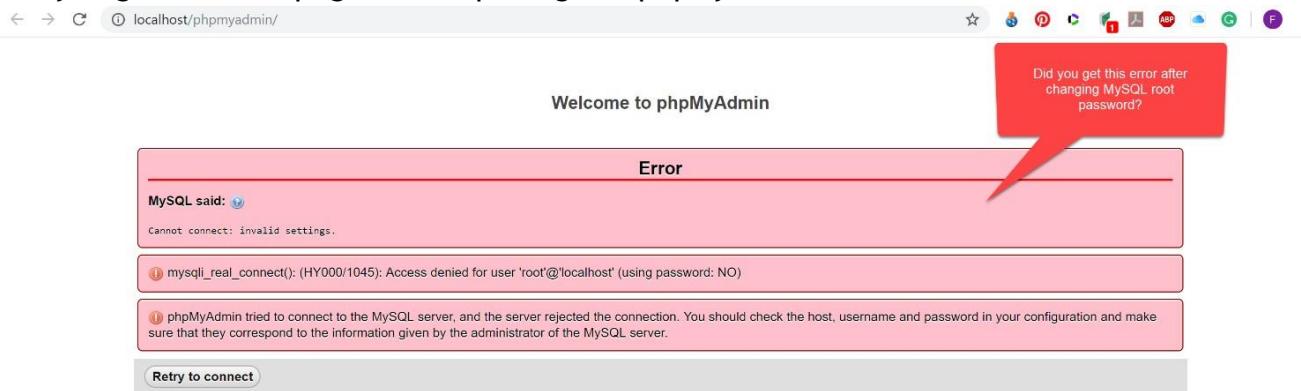
Remark: The above instructions are taken from Apache Friends Documentation. You may retrieve the documentation at <http://localhost/dashboard/docs/reset-mysql-password.html> after done with the XAMPP installation.

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.

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

Modules

Service	Module	PID(s)	Port(s)	Actions
	Apache	6276 12608	80, 443	Stop Admin Config Logs
	MySQL	17596	3306	Stop Admin Config Logs
	FileZilla			Start Admin Config Logs
	Mercury			Start Admin Config Logs
	Tomcat			Start Admin Config Logs

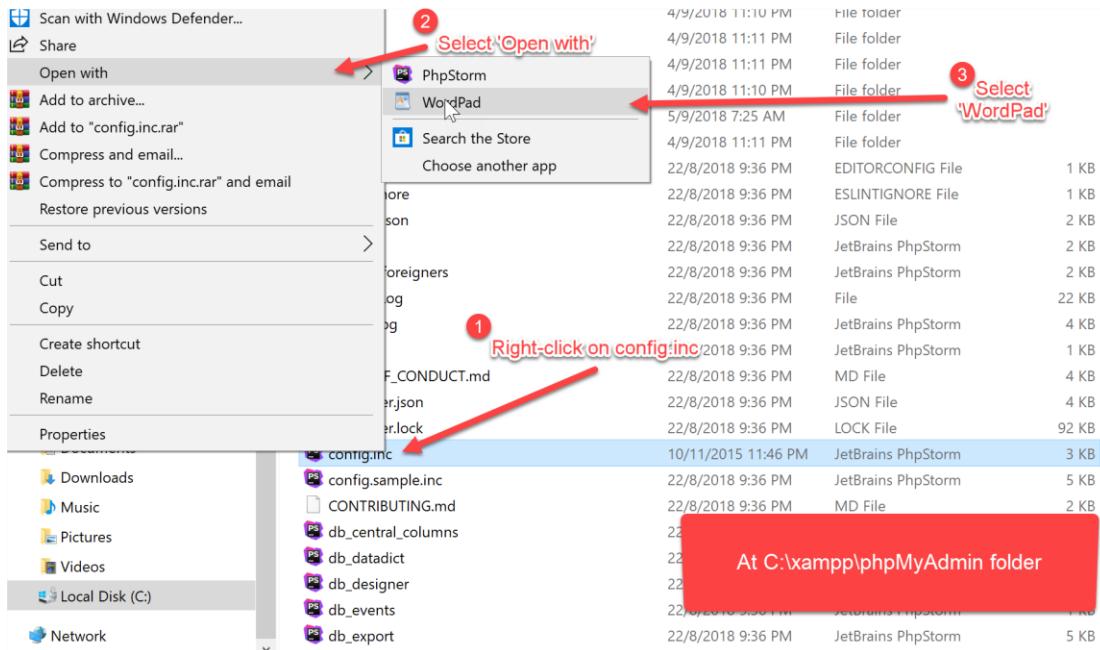
Stop the MySQL Server

1

2

12:24:45 AM [Apache] Attempting to stop Apache (PID: 9896)
12:24:45 AM [Apache] Status change detected: stopped
12:24:50 AM [mysql] Attempting to stop MySQL app...
12:24:50 AM [mysql] Status change detected: stopped
12:24:54 AM [Apache] Attempting to start Apache app...
12:24:55 AM [Apache] Status change detected: running
12:24:56 AM [mysql] Attempting to start MySQL app...
12:24:57 AM [mysql] Status change detected: running

Then, go to C:\xampp\phpMyAdmin
Note: it depend on where you install your XAMPP



```

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'; ① Put your root password here. In this example, 'admin' is the root password.
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = false; ② Change the value from 'true' to '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';

```

③ Save the file

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

XAMPP Control Panel v3.2.2

Re-start the MySQL server

Service	Module	PID(s)	Port(s)	Actions
	Apache	15016 19540	80, 443	Stop Admin Config Logs Start
	MySQL			Admin Config Logs
	FileZilla			Admin Config Logs
	Mercury			Admin Config Logs
	Tomcat			Start Admin Config Logs

```

12:45:11 AM [Apache] Attempting to stop Apache (PID: 12608)
12:45:11 AM [Apache] Status change detected: stopped
12:45:26 AM [Apache] Attempting to start Apache app...
12:45:26 AM [Apache] Status change detected: running
12:45:32 AM [mysql] Attempting to start MySQL app...
12:45:32 AM [mysql] Status change detected: running
12:45:34 AM [mysql] Attempting to stop MySQL app...
12:45:35 AM [mysql] Status change detected: stopped

```

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

XAMPP Control Panel v3.2.2

Click 'Admin' to verify the changes

Service	Module	PID(s)	Port(s)	Actions
	Apache	15016 19540	80, 443	Stop Admin Config Logs
	MySQL	12188	3306	Stop Admin Config Logs
	FileZilla			Start Admin Config Logs
	Mercury			Start Admin Config Logs
	Tomcat			Start Admin Config Logs

```

12:45:26 AM [Apache] Attempting to start Apache app...
12:45:26 AM [Apache] Status change detected: running
12:45:32 AM [mysql] Attempting to start MySQL app...
12:45:32 AM [mysql] Status change detected: running
12:45:34 AM [mysql] Attempting to stop MySQL app...
12:45:35 AM [mysql] Status change detected: stopped
12:46:52 AM [mysql] Attempting to start MySQL app...
12:46:53 AM [mysql] Status change detected: running

```

MySQL Server successfully started

General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

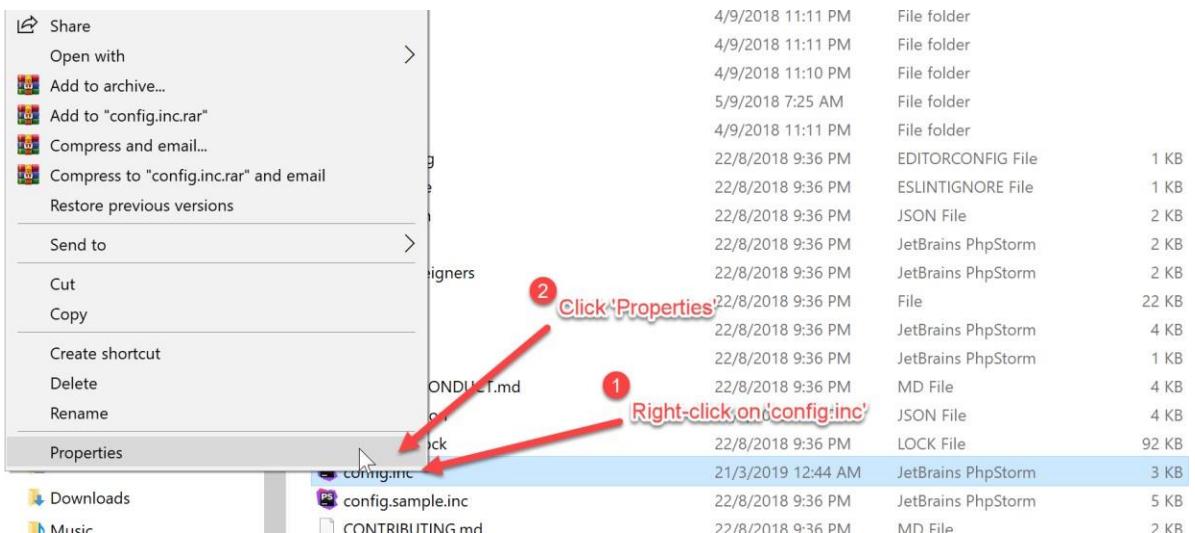
Theme: pmahomme

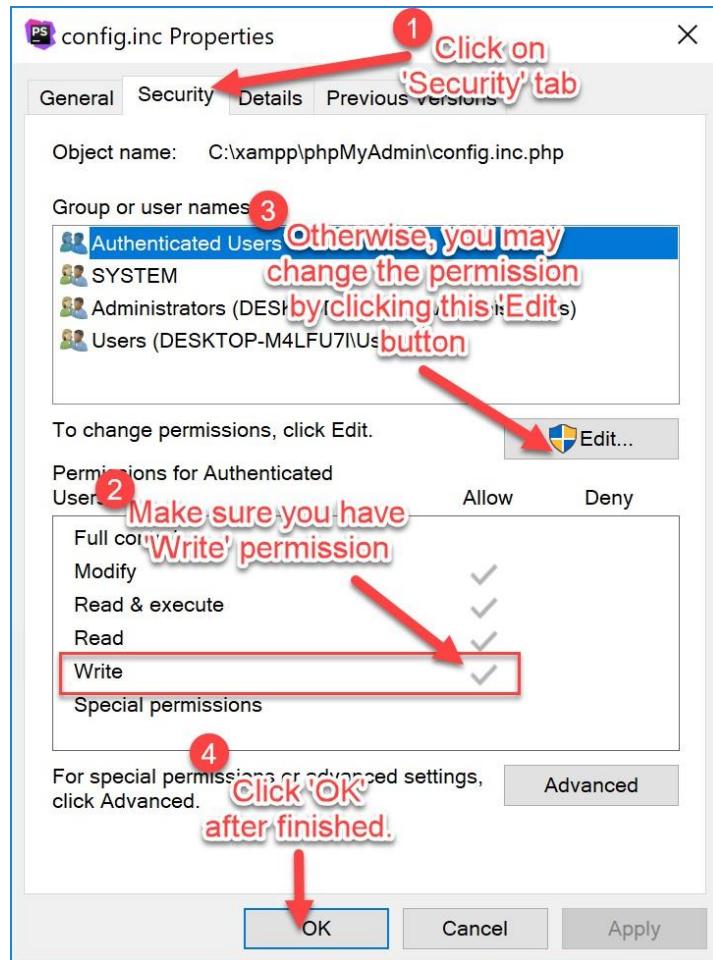
Font size: 82%

More settings

If you done it correctly, then you should see this page. Now, you can start to create a database and the tables. Good luck!

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:**

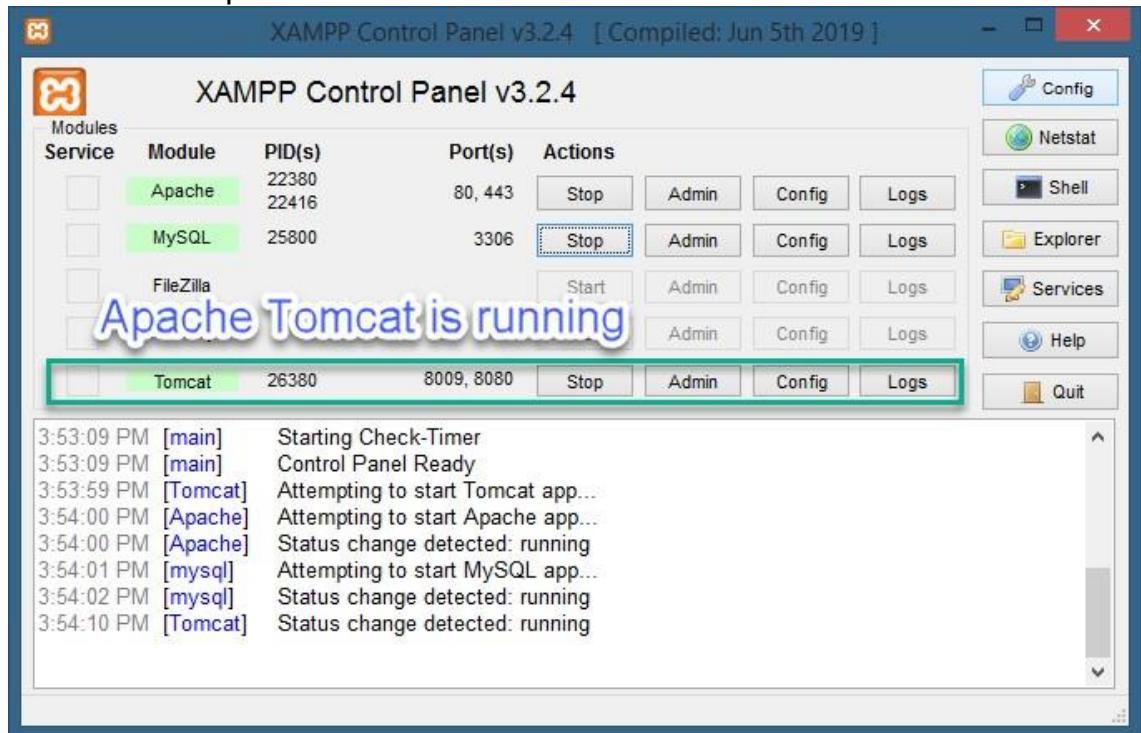
To configure user access for Apache Tomcat in XAMPP and test the access at localhost.

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:

- 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”.

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

You are not authorized to view this page. If you have not changed any configuration files, please examine the file `[conf/tomcat-users.xml]` in your installation. That file must contain the credentials to let you use this webapp.
For example, to add the `manager-gui` role to a user named `tomcat` with a password of `s3cret`, add the following to the config file listed above.

```
<role rolename="manager-gui"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
```

Note that for Tomcat 7 onwards, the roles required to use the manager application were changed from the single `manager` role to the following four roles. You will need to assign the role(s) required for the functionality you wish to access.

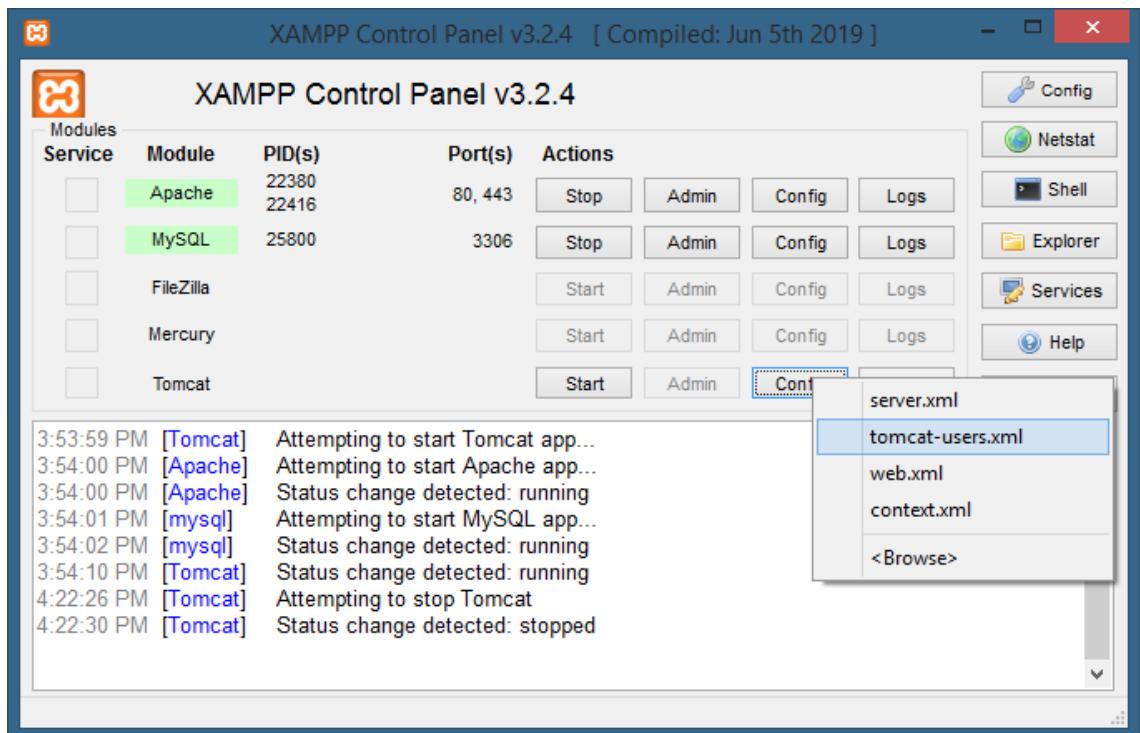
- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

The HTML interface is protected against CSRF but the text and JMX interfaces are not. To maintain the CSRF protection:

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for tools not humans) then the browser must be closed afterwards to terminate the session.

For more information - please see the [Manager App How-To](#).

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



6. The tomcat-users.xml will be opened using the default editor in your system. Scroll down the file until you see the </tomcat-users> tag. Add the tags below at the top of </tomcat-users> tag.

```
<role rolename="manager-script"/>
<role rolename="manager-gui"/>
<user username="admin" password="admin"
roles="manager-script,manager-gui"/>
```

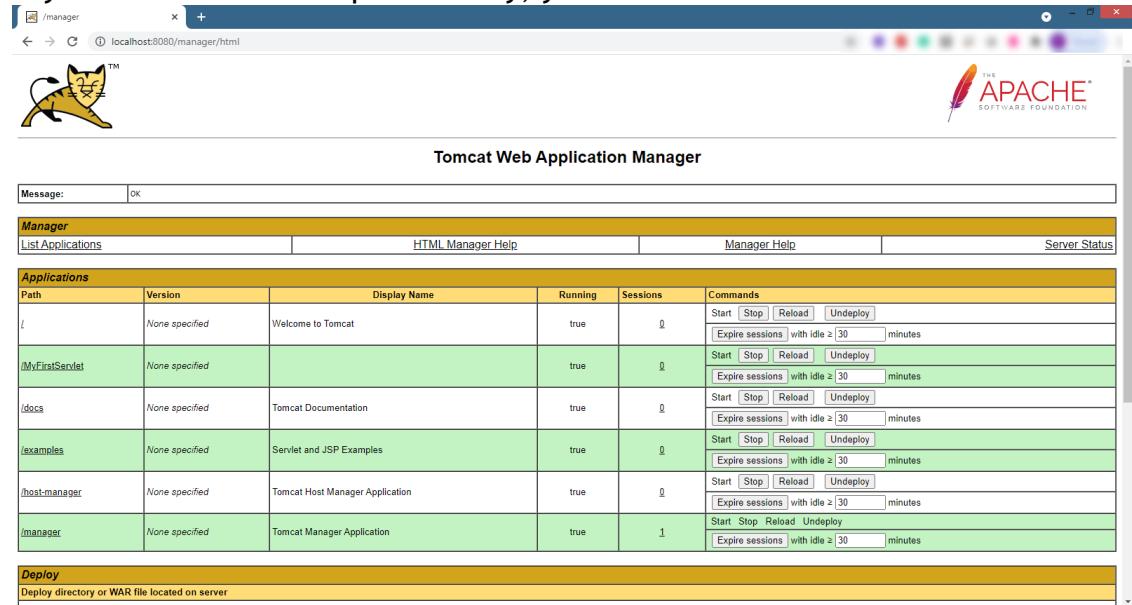
7. Save the file. Finally, tomcat-users.xml will look like this

The screenshot shows a Notepad window titled 'tomcat-users - Notepad'. The file contains XML code for user configuration. A new user entry has been added at the bottom:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
version="1.0">
<!--
NOTE: By default, no user is included in the "manager-gui" role required
to operate the "/manager/html" web application. If you wish to use this app,
you must define such a user - the username and password are arbitrary. It is
strongly recommended that you do NOT use one of the users in the commented out
section below since they are intended for use with the examples web
application.
-->
<!--
NOTE: The sample user and role entries below are intended for use with the
examples web application. They are wrapped in a comment and thus are ignored
when reading this file. If you wish to configure these users for use with the
examples web application, do not forget to remove the <!...> that surrounds
them. You will also need to set the passwords to something appropriate.
-->
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
<user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<role rolename="manager-script"/>
<role rolename="manager-gui"/>
<user username="admin" password="admin" roles="manager-script,manager-gui"/>
</tomcat-users>
```

8. We have added a new user “admin” with a password “admin”. Be sure to change the password when you deploy your application on a production server.

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:



The screenshot shows the Tomcat Web Application Manager interface. At the top, there's a header bar with the Apache logo and the title "Tomcat Web Application Manager". Below the header, there's a message box that says "Message: OK". The main area is titled "Manager" and contains two tabs: "List Applications" and "HTML Manager Help". The "List Applications" tab is selected, displaying a table of deployed applications. The table has columns for Path, Version, Display Name, Running, Sessions, and Commands. The applications listed are: "/>" (Welcome to Tomcat), "/MyFirstServlet" (Servlet and JSP Examples), "/docs" (Tomcat Documentation), "/examples" (Servlet and JSP Examples), "/host-manager" (Tomcat Host Manager Application), and "/manager" (Tomcat Manager Application). Each row in the table includes a set of buttons for managing the application: Start, Stop, Reload, Undeploy, and Expire sessions (with idle ≥ 30 minutes).

Applications					
Path	Version	Display Name	Running	Sessions	Commands
/>	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/MyFirstServlet	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy
Deploy directory or WAR file located on server

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*

The screenshot shows the Apache Software Foundation download page for Netbeans 12.3. The URL in the address bar is <https://apache.org/dyn/closer.cgi/netbeans/netbeans/12.3/Apache-NetBeans-12.3-bin-windows-x64.exe>. The page features the Apache logo and navigation links for News, About, Make a Donation, The Apache Way, Join Us, Downloads, and a search bar. The main content area displays the download link for the Windows x64 installer: <https://downloads.apache.org/netbeans/netbeans/12.3/Apache-NetBeans-12.3-bin-windows-x64.exe>. It also provides instructions for verifying file integrity using PGP signatures or hashes and links to backup sites and becoming a mirror.

COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

HTTP

<https://downloads.apache.org/netbeans/netbeans/12.3/Apache-NetBeans-12.3-bin-windows-x64.exe>

BACKUP SITES

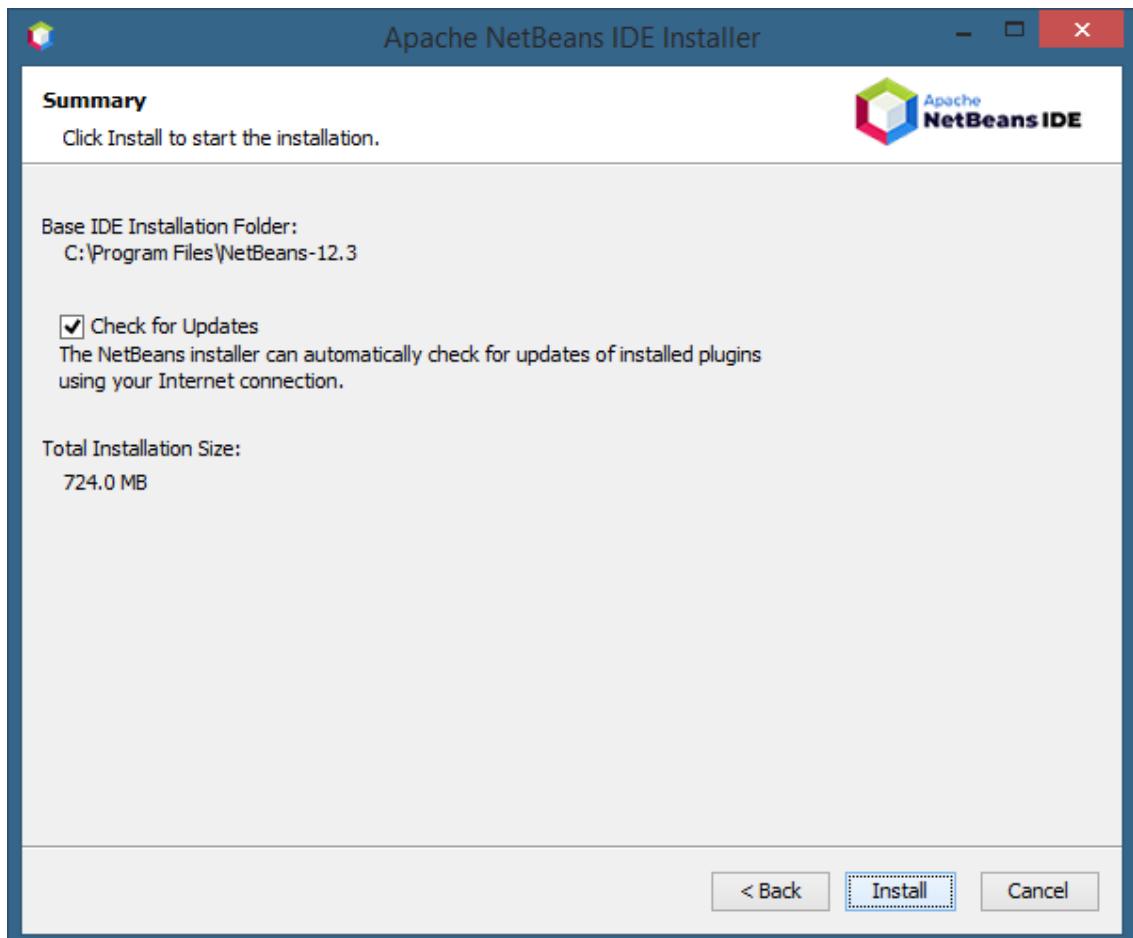
Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA* etc) -- or if no other mirrors are working.

<https://downloads.apache.org/netbeans/netbeans/12.3/Apache-NetBeans-12.3-bin-windows-x64.exe>

BECOMING A MIRROR

The procedure for setting up new mirrors is described in How to become a mirror.

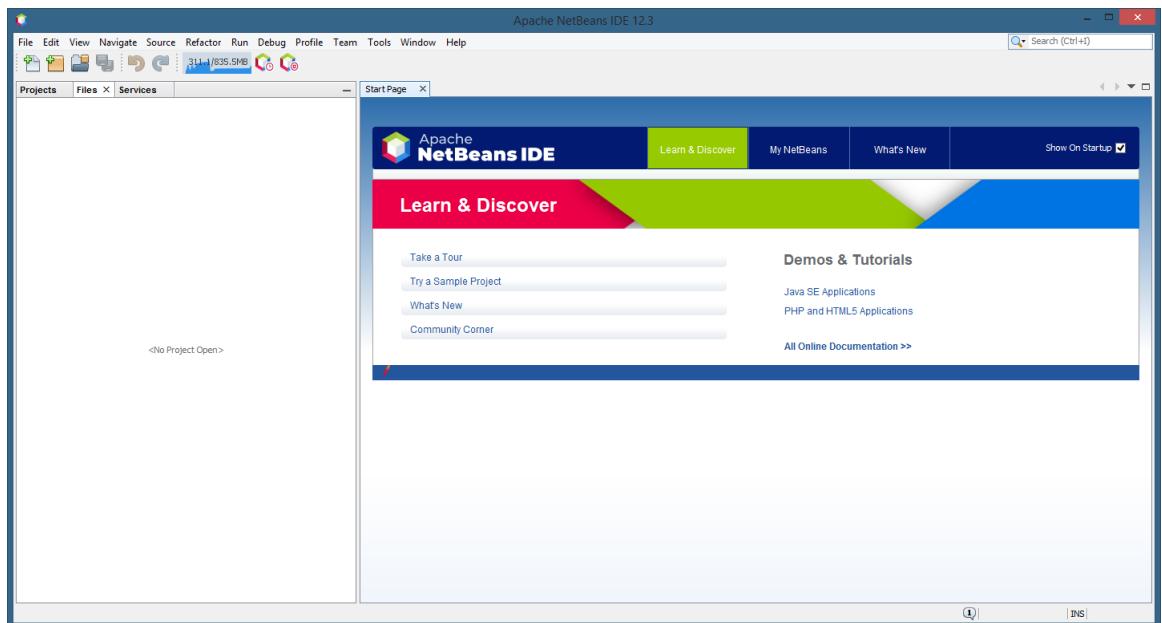
13. After the download finish, double-click on the installer and click “Next” until you reach the following screen:



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

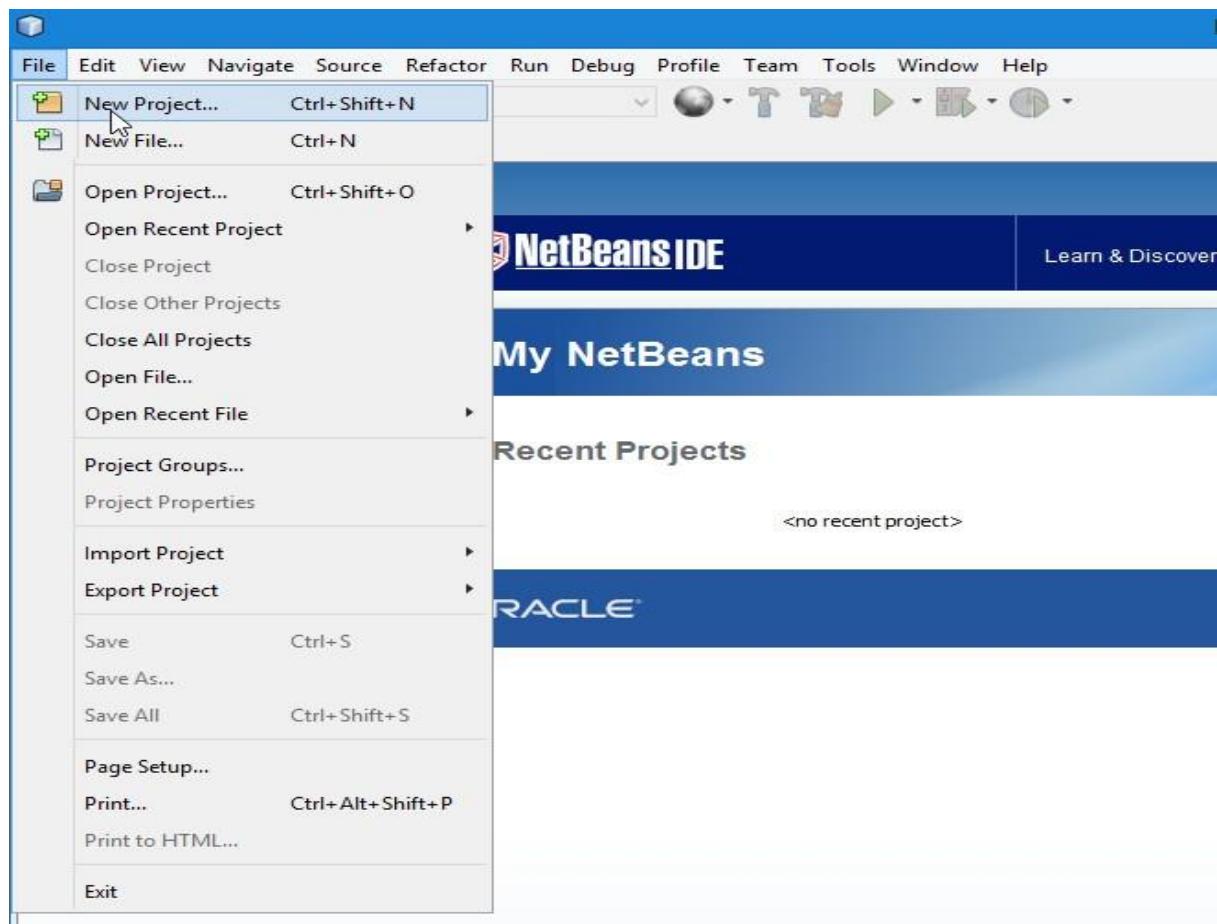
Problem

To write a simple Java Servlet

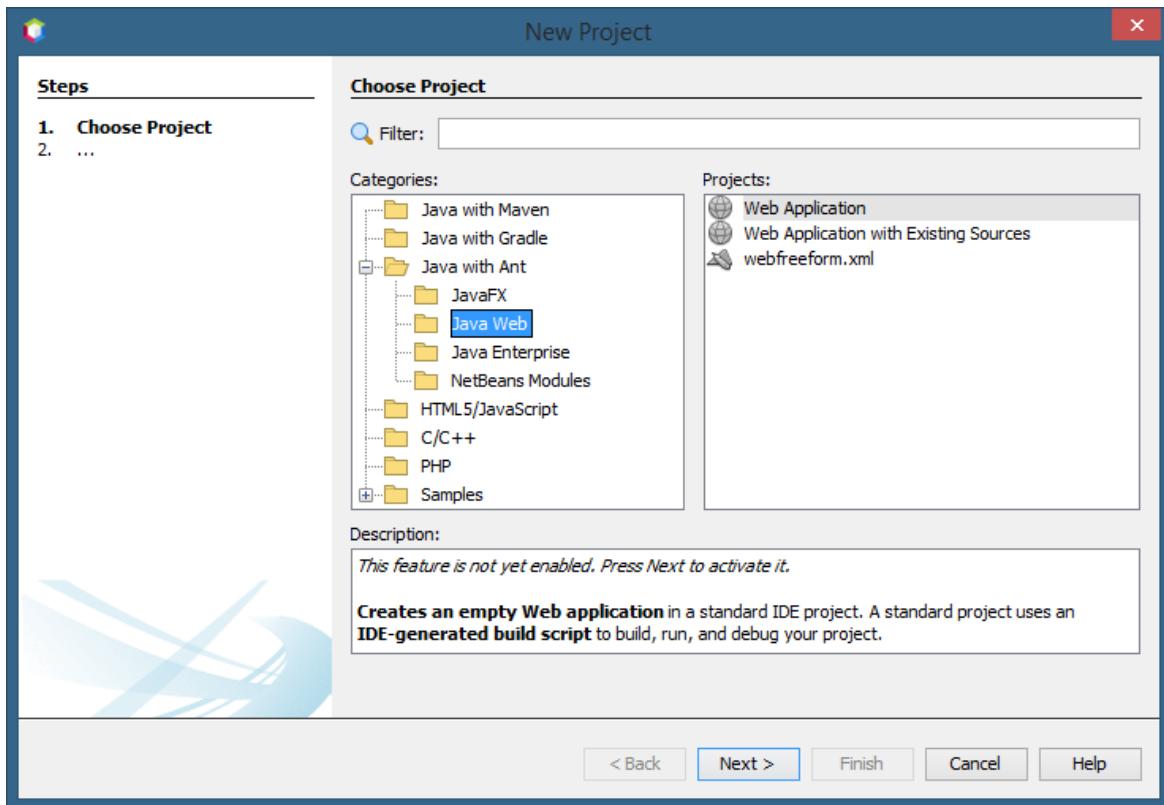
Description:**Estimated time:**

15 minutes

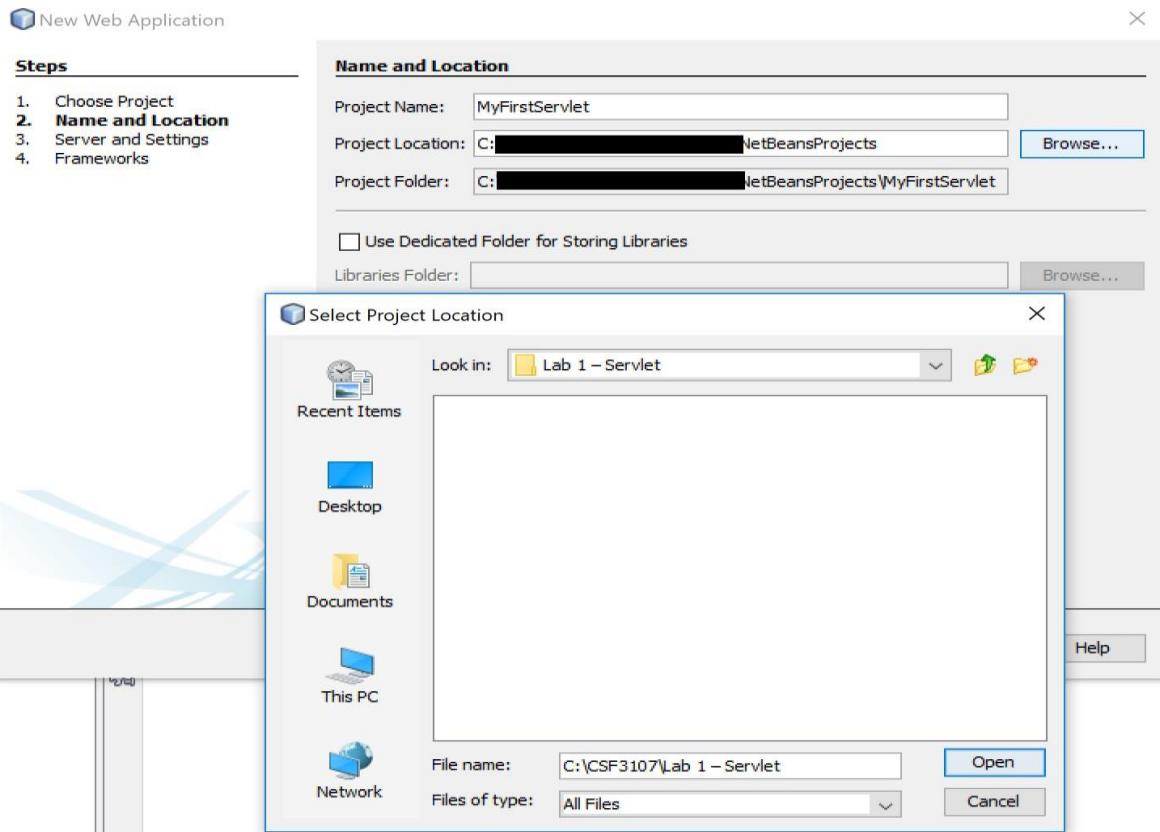
1. Create a directory *C:\[MATRICNUMBER]*.
2. Go to *C:\[MATRICNUMBER]* Lab's directory and create sub-directory as *Lab 1 - Servlet*.
3. Open your NetBeans.
4. Go to File -> New Project



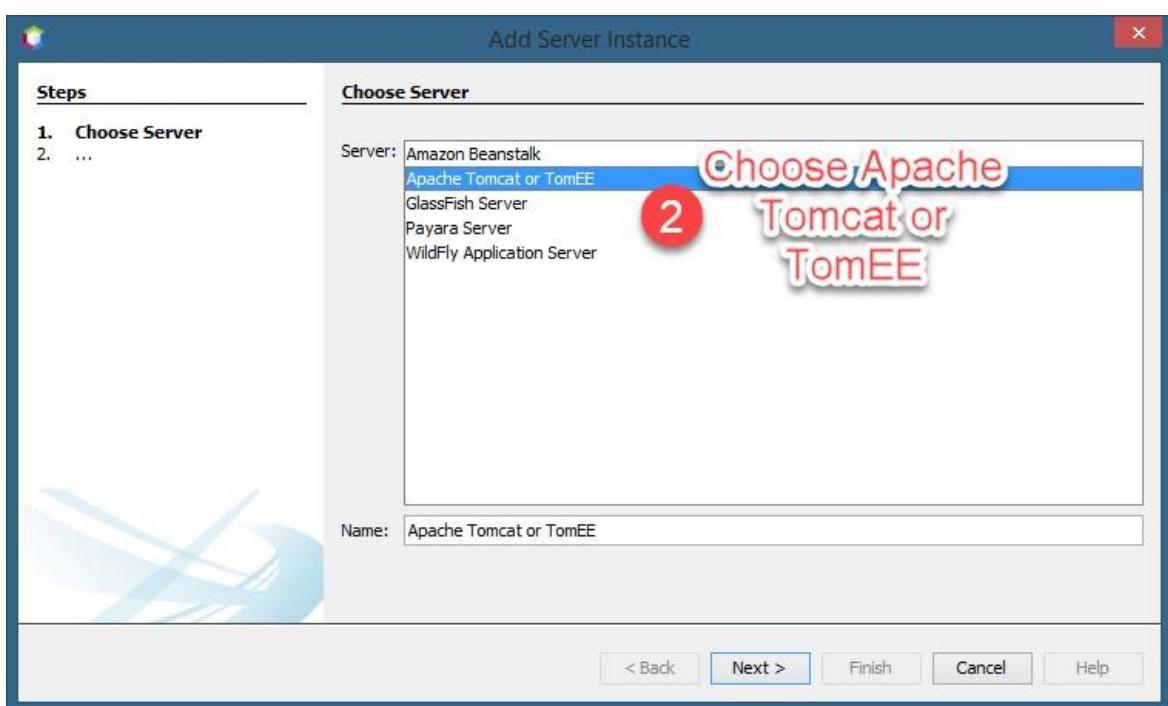
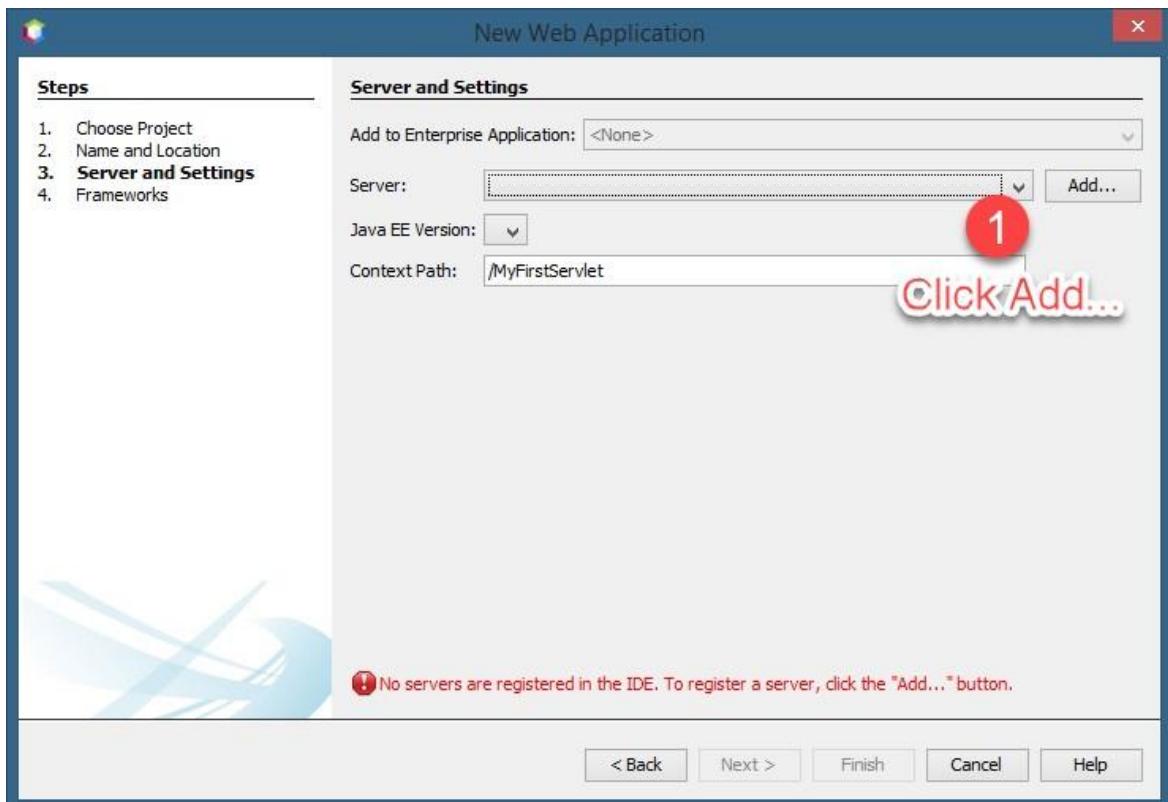
5. Select Java with Ant -> Java Web -> Web Application and click Next.

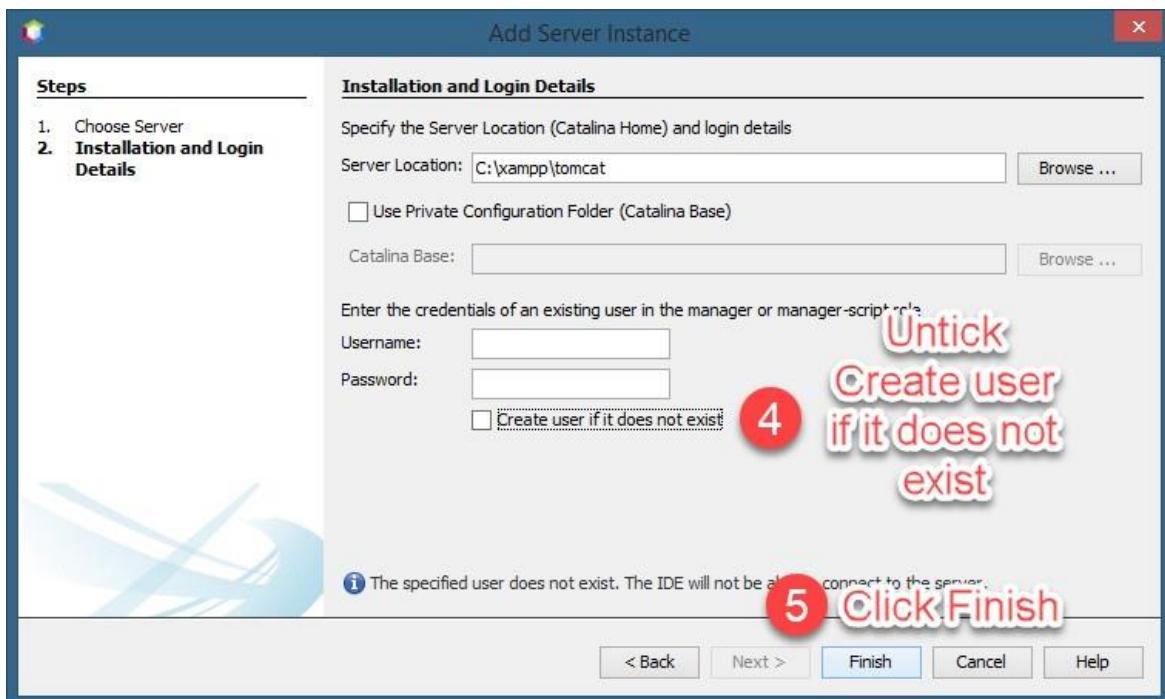
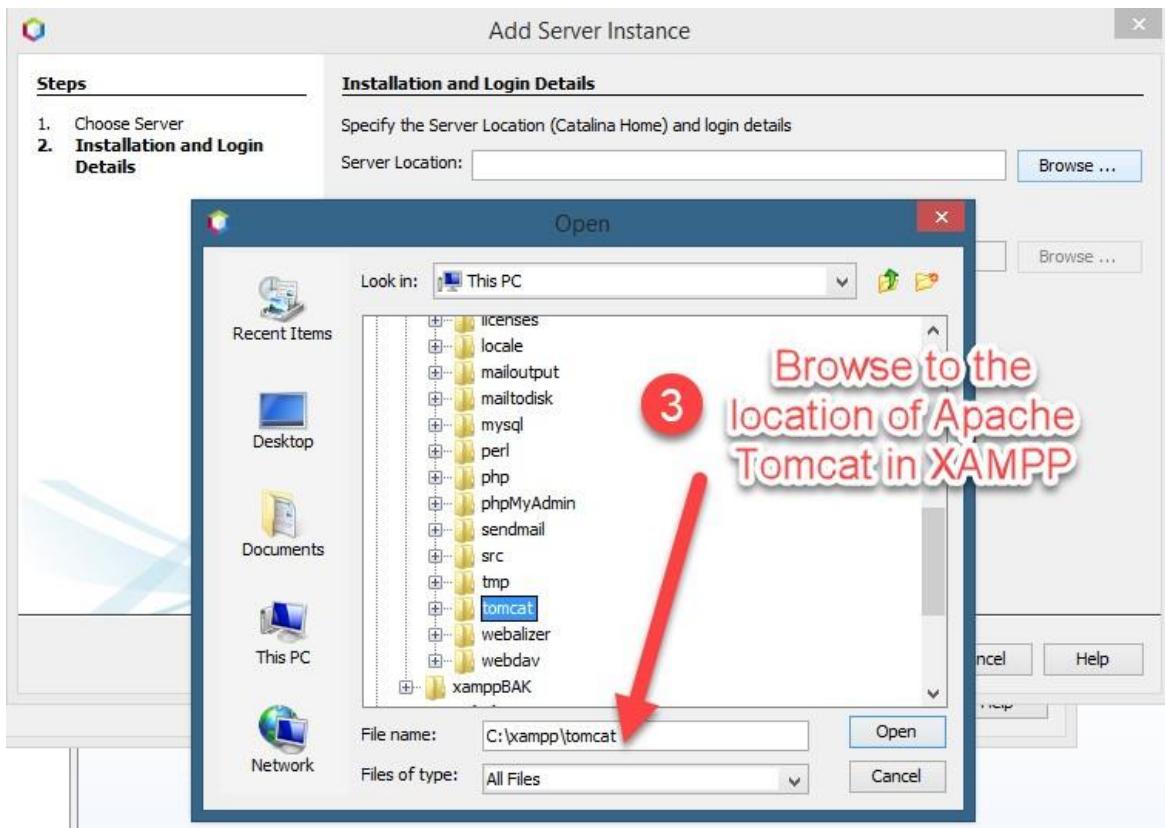


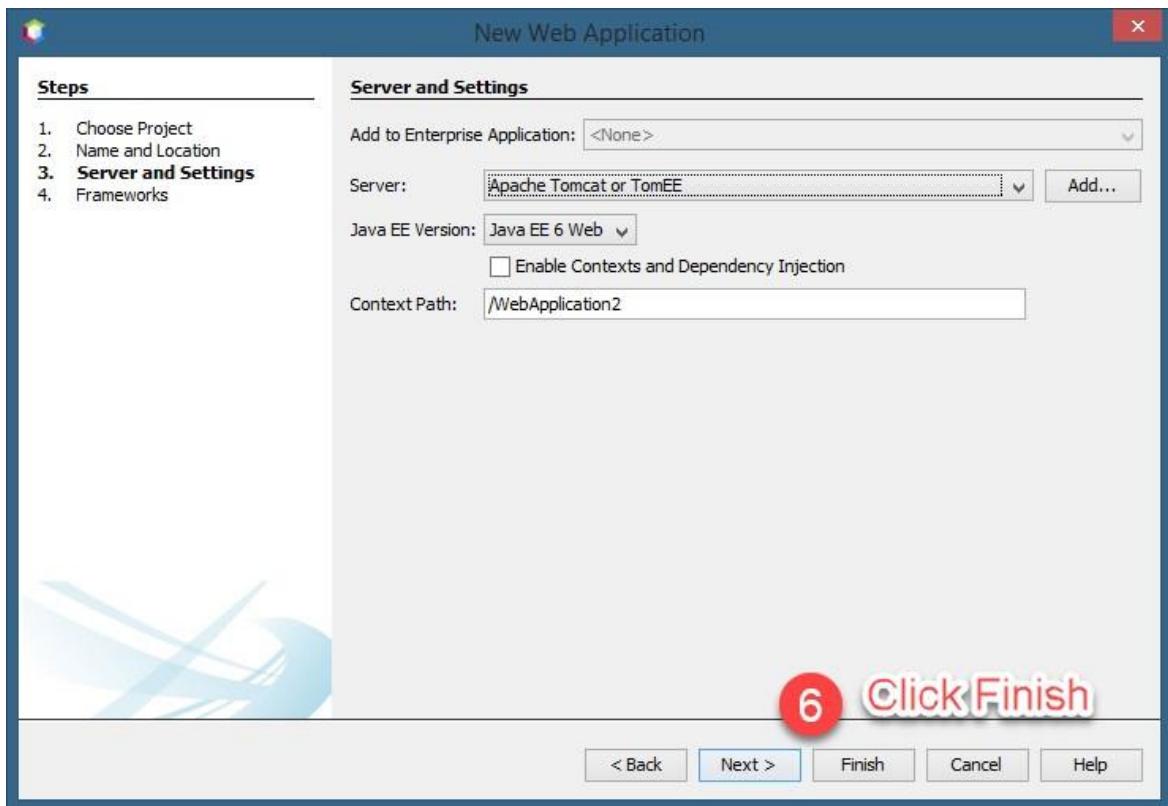
1. Type Project Name: *MyFirstServlet*.
2. Click Browse and choose Project Location: *C:\[MATRICNUMBER]\Lab 1 - Servlet*. Then click the Next button.



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.

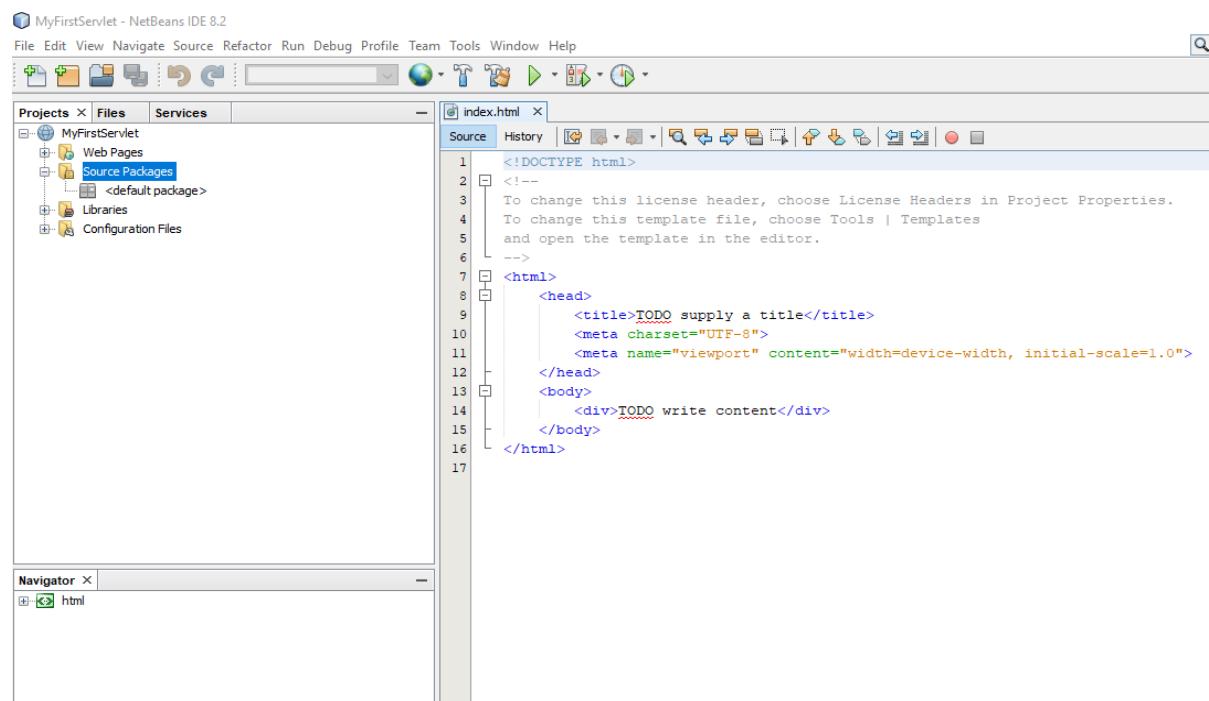




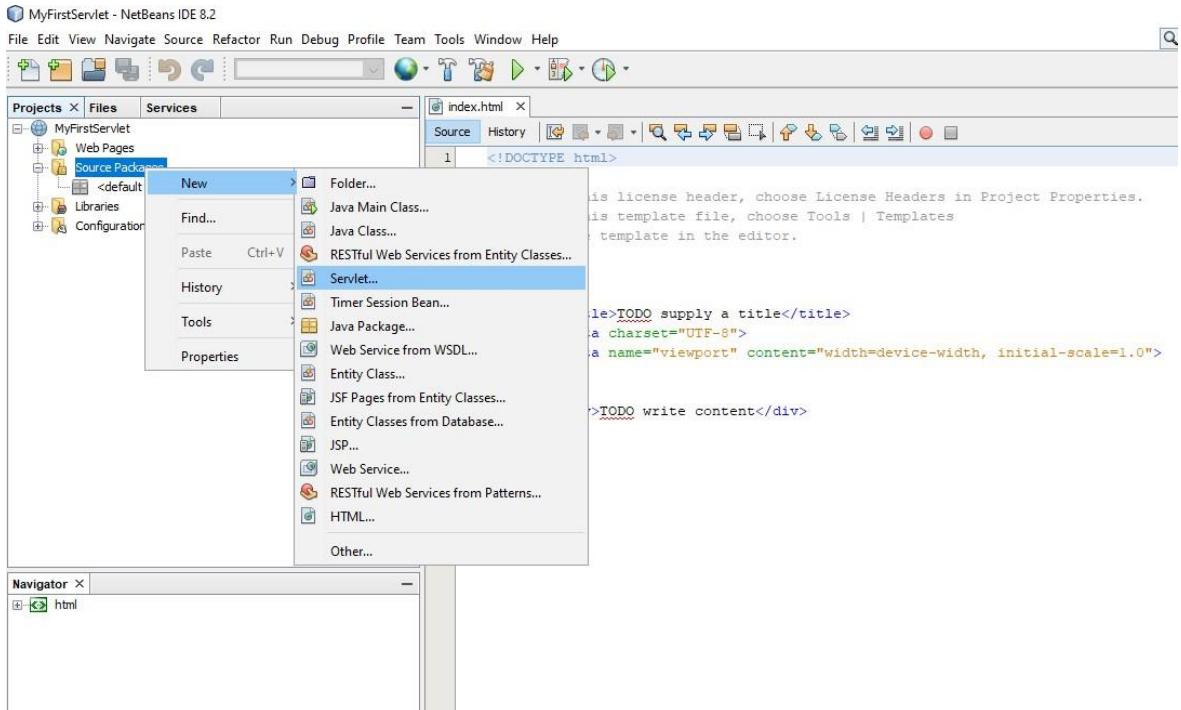


Note: the above steps only need to be ran for the first time you link the Netbeans to Apache Tomcat in XAMPP. After this, you just need to select it from the screen.

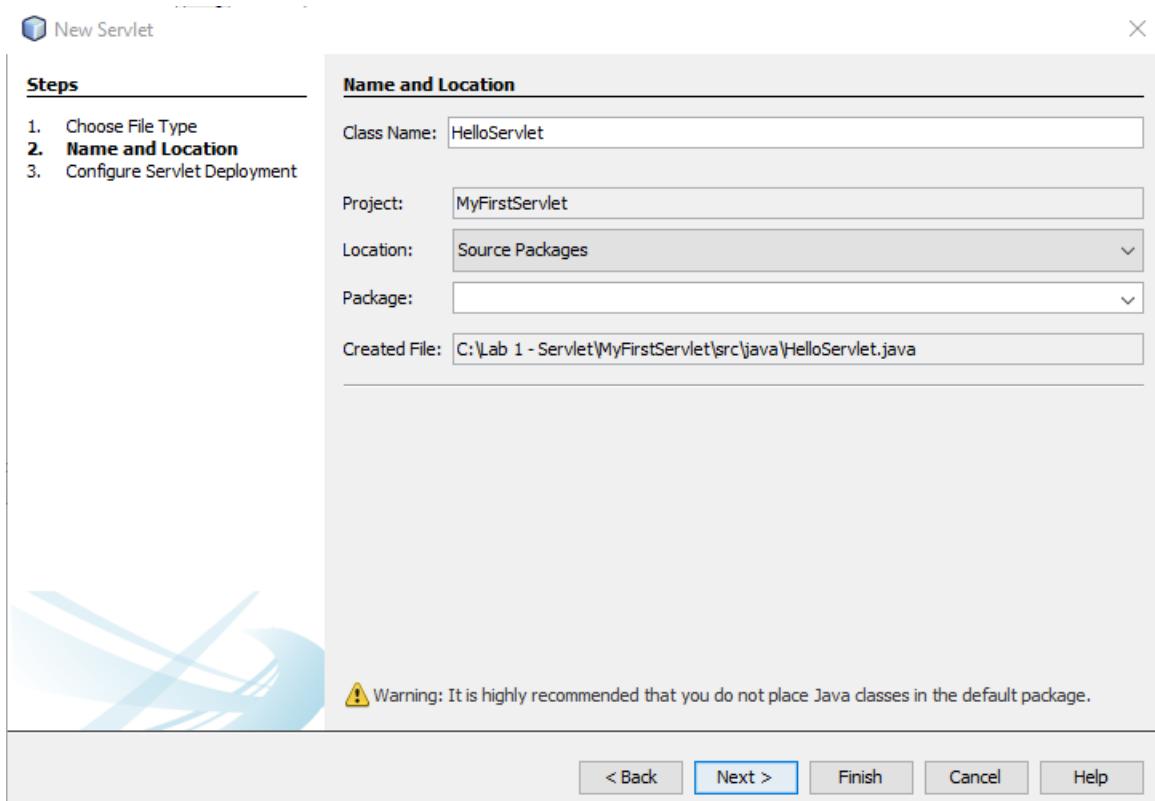
4. If everything okay, you will see the following screen:



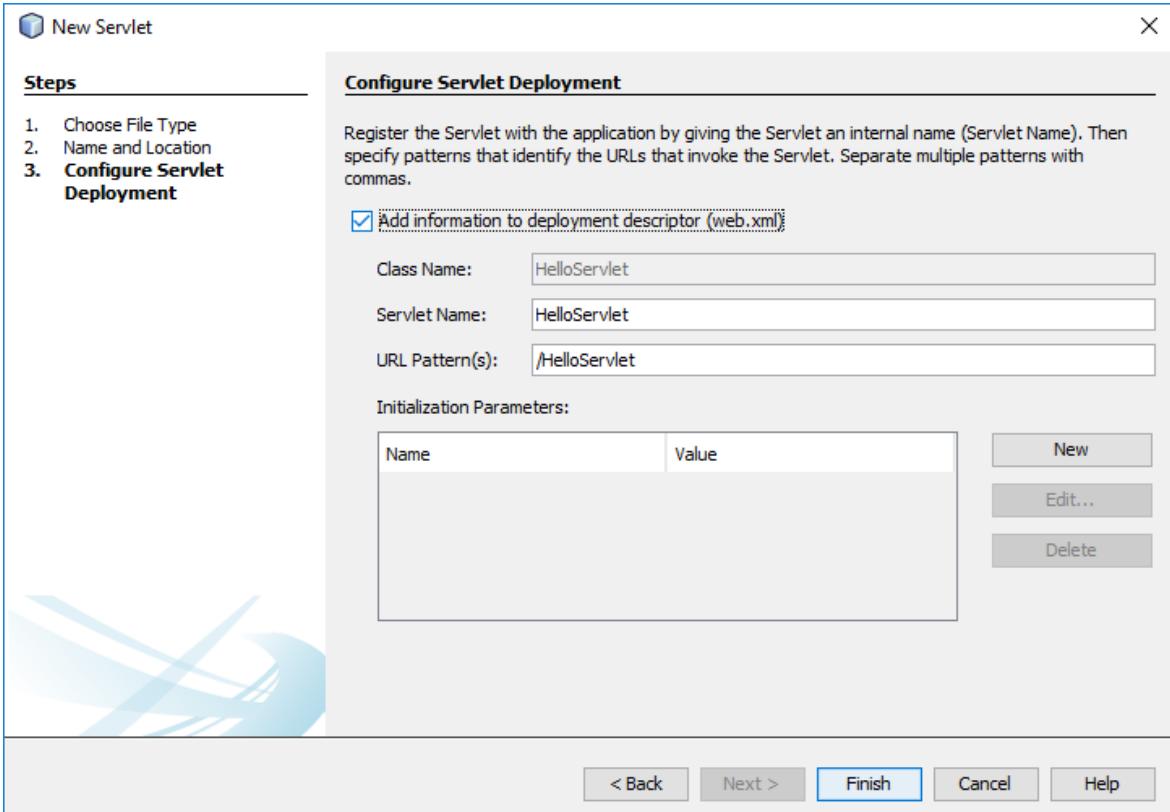
5. Create a new Servlet file.



6. Name your servlet as *HelloServlet*.



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



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.

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.GenericServlet;

/**
 *
 * @author Fakhru Adli
 */
public class HelloServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* ... */
        }
    }

    // doGet(HttpServletRequest request, HttpServletResponse response)
    // doPost(HttpServletRequest request, HttpServletResponse response)
    // getServletInfo(): String
    // processRequest(HttpServletRequest request, HttpServletResponse response)
}

```

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

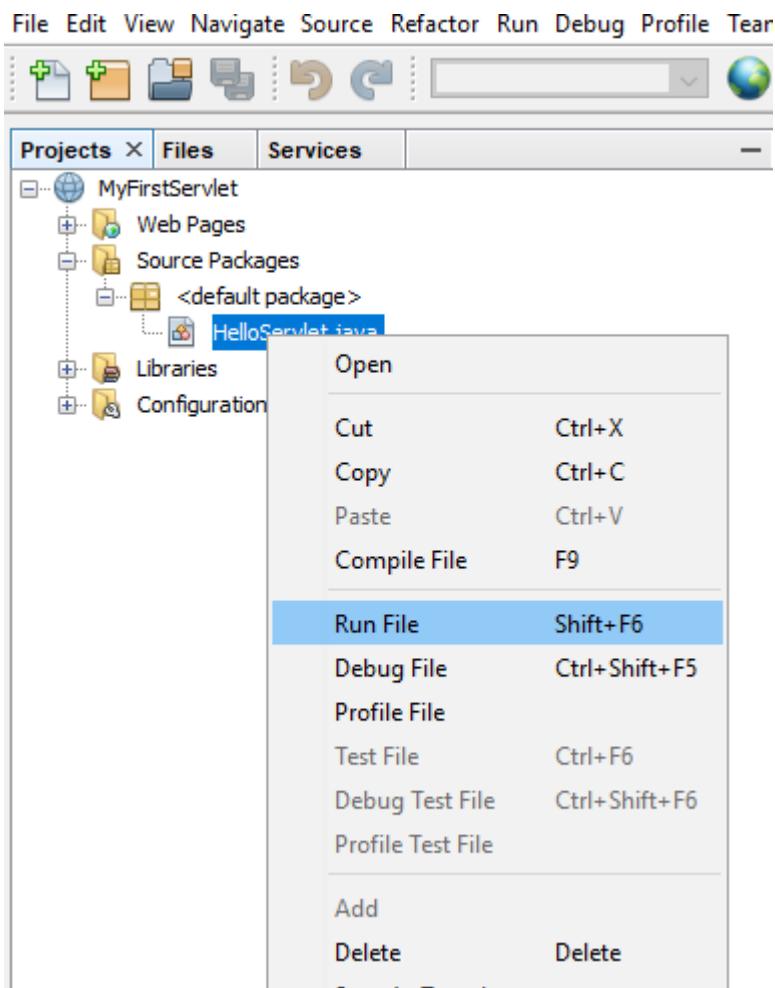
10. We are going to make a simple modification to the existing codes. Modify the them as follows. Refer to line 37, 40 and 41.

The screenshot shows a Java IDE interface with the following details:

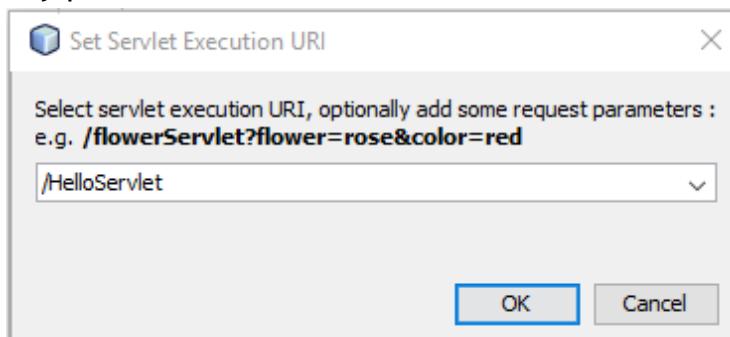
- Title Bar:** Shows "Start Page" and three tabs: "index.html" and "HelloServlet.java".
- Toolbar:** Includes standard icons for file operations, search, and navigation.
- Code Editor:** Displays the Java code for "HelloServlet.java". The code is annotated with Javadoc comments and uses annotations (@param, @throws). The method "processRequest" is highlighted with a yellow background.

```
16 * @author Fahrul Adli
17 */
18 public class HelloServlet extends HttpServlet {
19
20     /**
21      * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
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         throws ServletException, IOException {
32         response.setContentType("text/html;charset=UTF-8");
33         try (PrintWriter out = response.getWriter()) {
34             /* TODO output your page here. You may use following sample code. */
35             out.println("<!DOCTYPE html>");
36             out.println("<html>");
37             out.println("<head>");
38             out.println("<title>Servlet Saya Yang Pertama</title>");
39             out.println("</head>");
40             out.println("<body>");
41             out.println("<h1>Hello, Servlet!</h1>");
42             out.println("<h2>Servlet HelloServlet at " + request.getContextPath() + "</h2>");
43             out.println("</body>");
44             out.println("</html>");
45         }
46     }
47 }
```

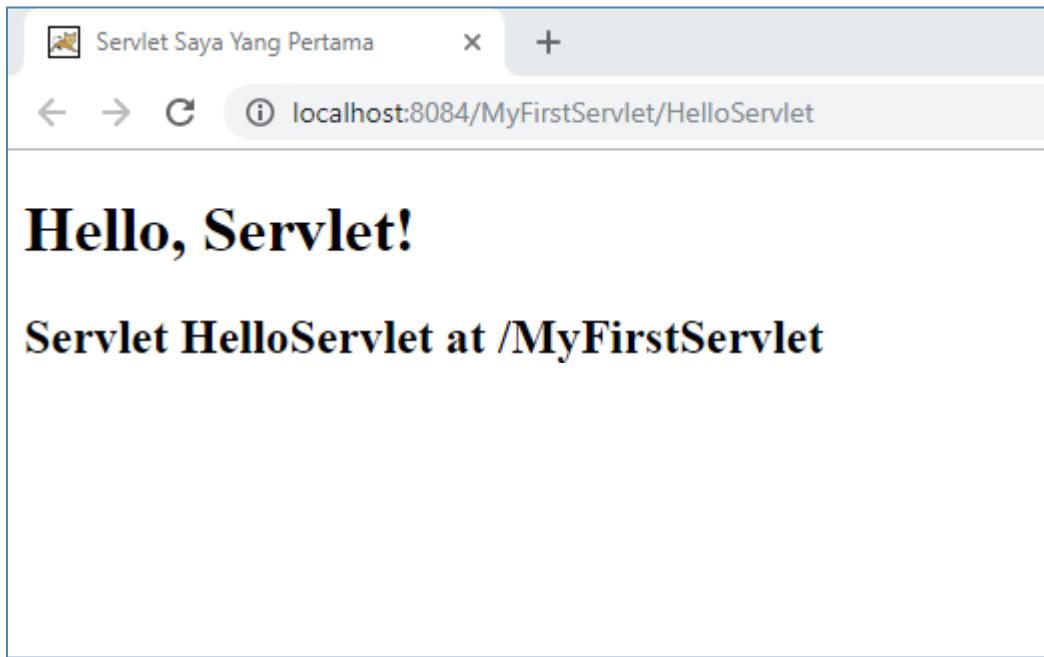
11. After finished your modification on the previous step, run *HelloServlet.java* by right-clicking on it and select *Run File*.



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.

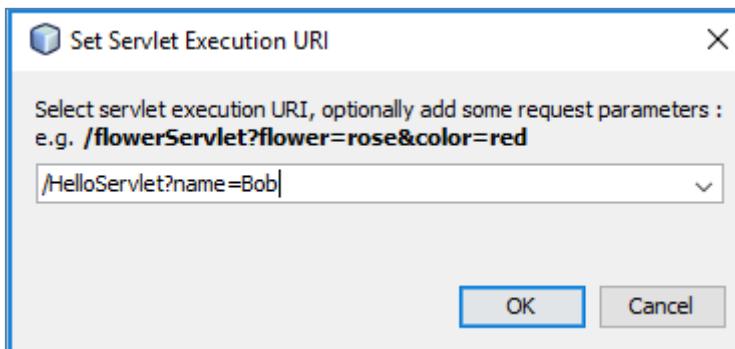


14. Now, we will make further modification to our servlet file. This time, we are going to modify the `doGet()` method. Browse through the codes in `HelloServlet.java` until you find the desired method. Type the codes as provided in the screenshot below. Read the comments carefully and try to understand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

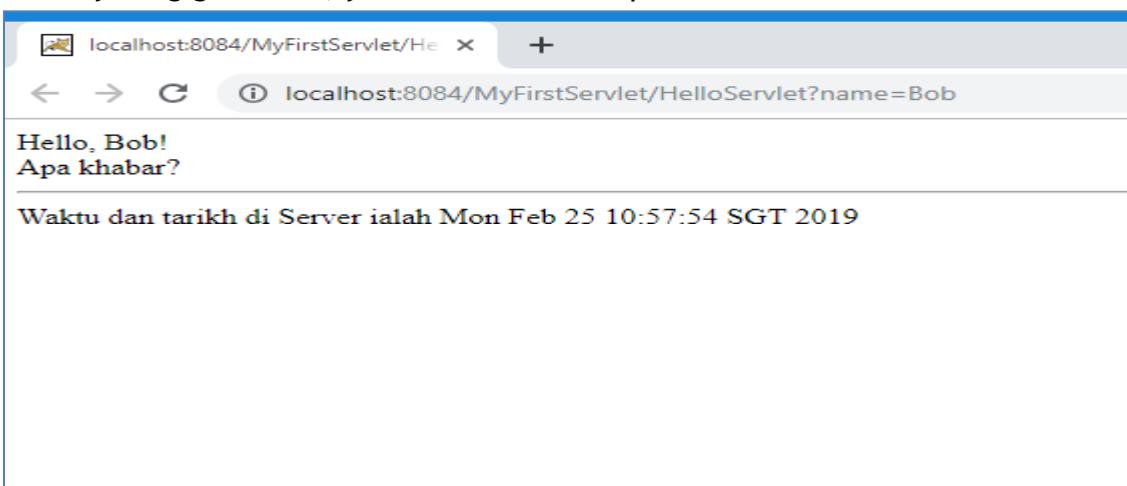
A screenshot of a code editor showing the Java code for `HelloServlet.java`. The code defines a `doGet` method that sets the content type to `text/html`, creates a `PrintWriter` object named `out`, reads a `name` parameter from the request, and prints a personalized response including the name, a question, the current date and time, and the closing HTML tags.

```
Start Page X index.html X HelloServlet.java X
Source History | 
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      /*Step 4: Generate content for our HTML response. Print the name
74      */
75      out.println("<html><body>");
76
77      out.println("Hello, "+name+"!"+<br>);
78      out.println("Apa khabar?"+<hr>);
79      out.println("Waktu dan tarikh di Server ialah "+new java.util.Date());
80      out.println("</html></body>");
81
82  }
```

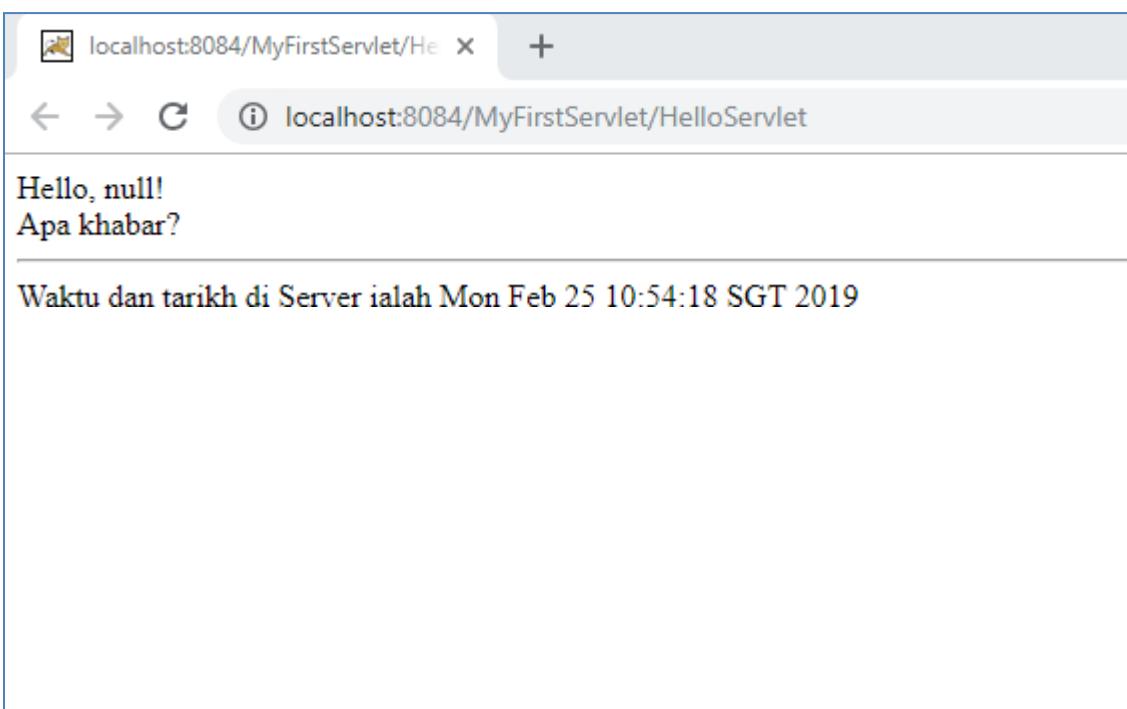
15. After finish, right click on *HelloServlet.java* and click Run. As you have seen previously, a dialogue box shows. This time, we will supply a value *Bob* to the parameter *name*. Then, click the OK button.



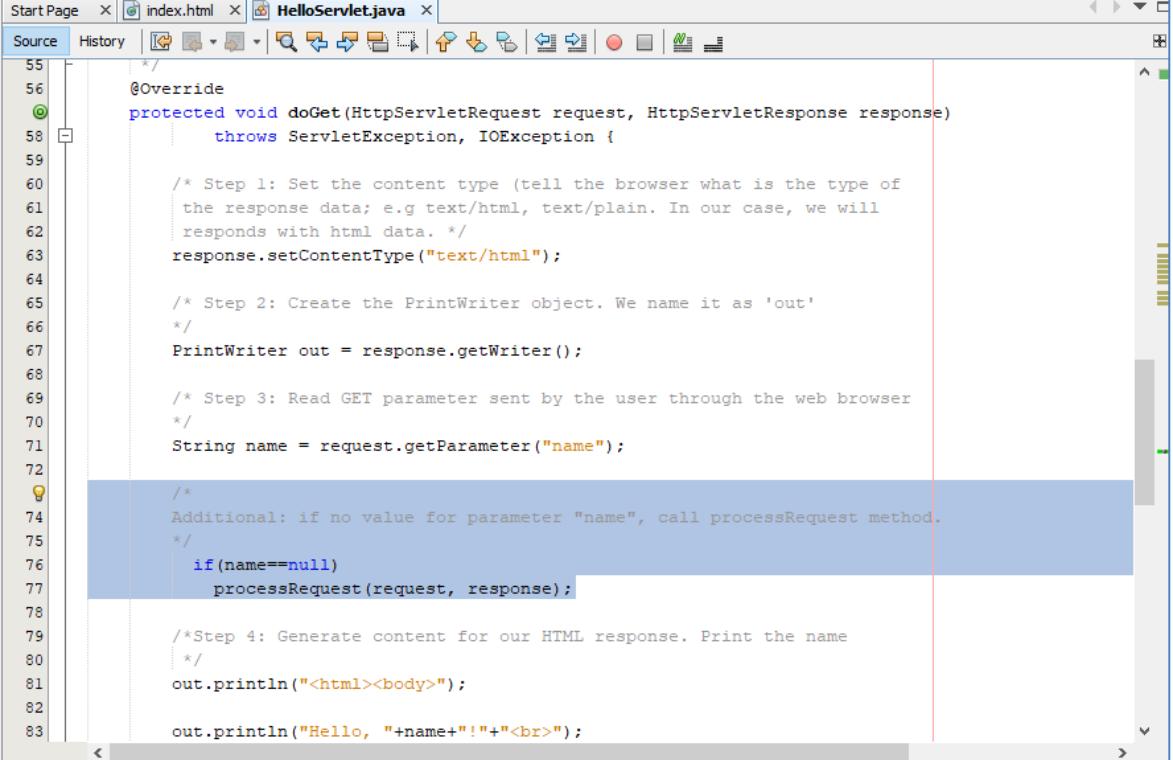
16. If everything goes well, you will see the output as below:



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?



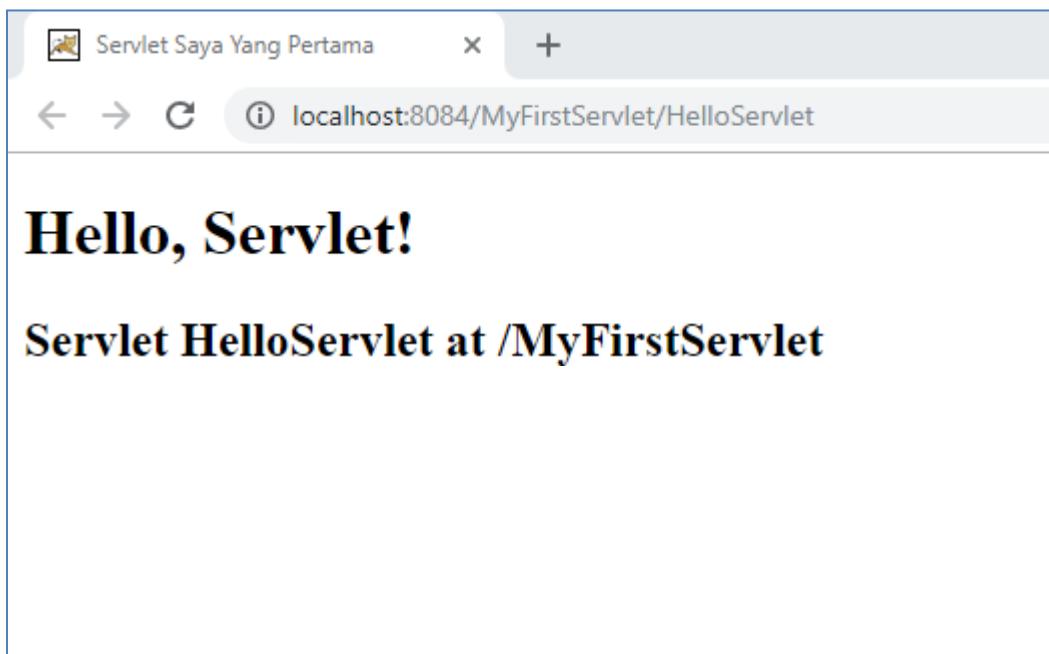
18. You can upgrade your code in HelloServlet.java by putting the following codes into it. By doing this, if no value supplied to the parameter *name*, the request will be passed to *processRequest()* method, and this will avoid from the *null* value appears on the browser.



The screenshot shows a Java code editor with the file "HelloServlet.java" open. The code is a doGet() method with annotations and logic to handle a "name" parameter. A specific section of the code is highlighted in blue:

```
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            respond 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:

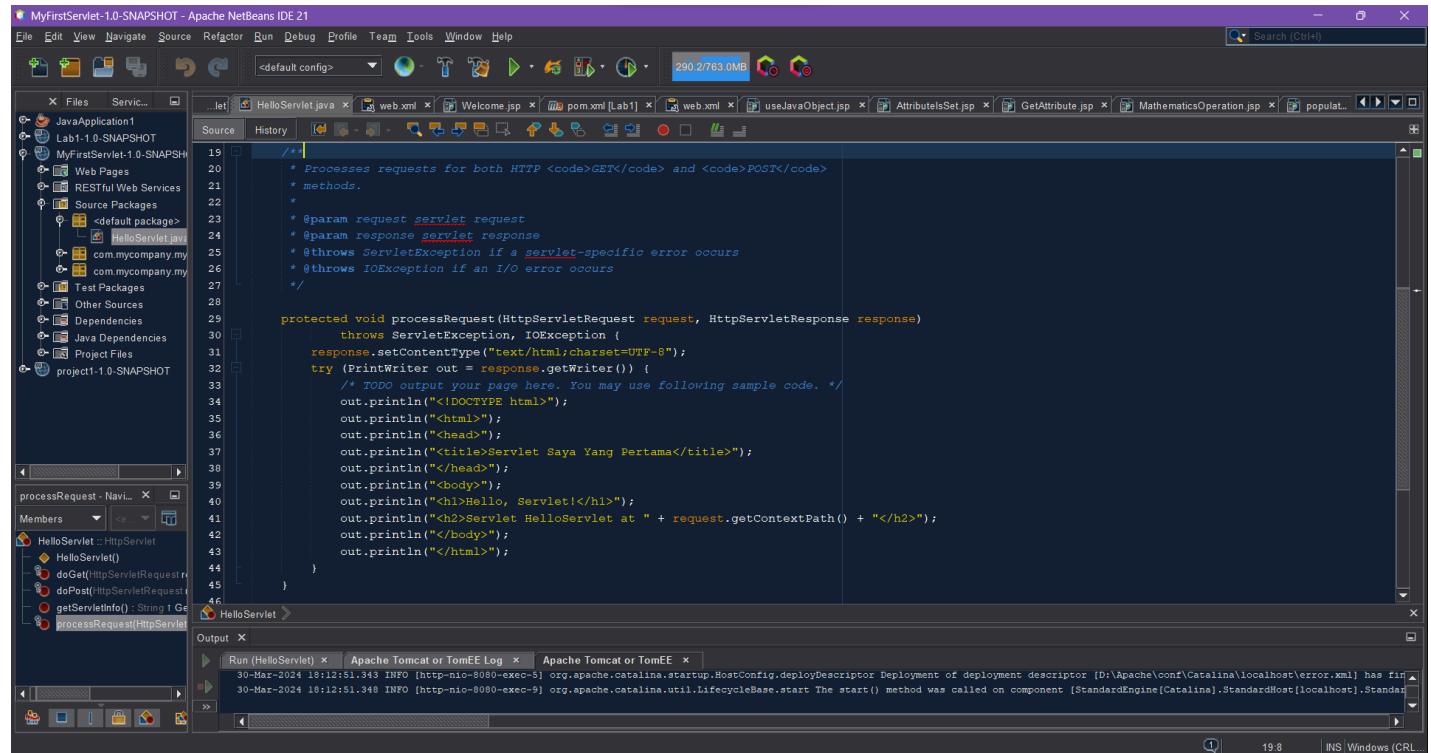


It is the same output as can be seen in Step 18: why?

Yes , cause the *processRequest()* method can avoid from the *null* value appears on the browser

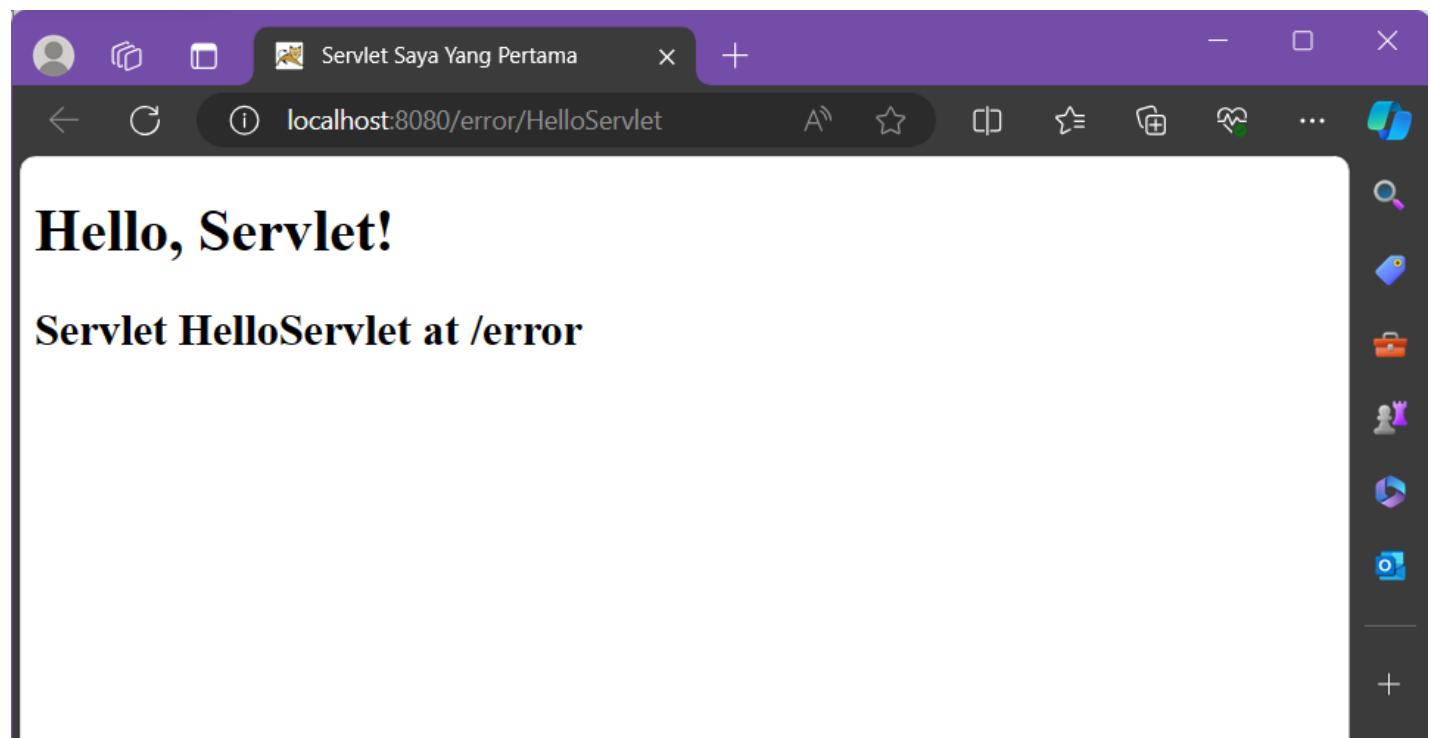
Source code

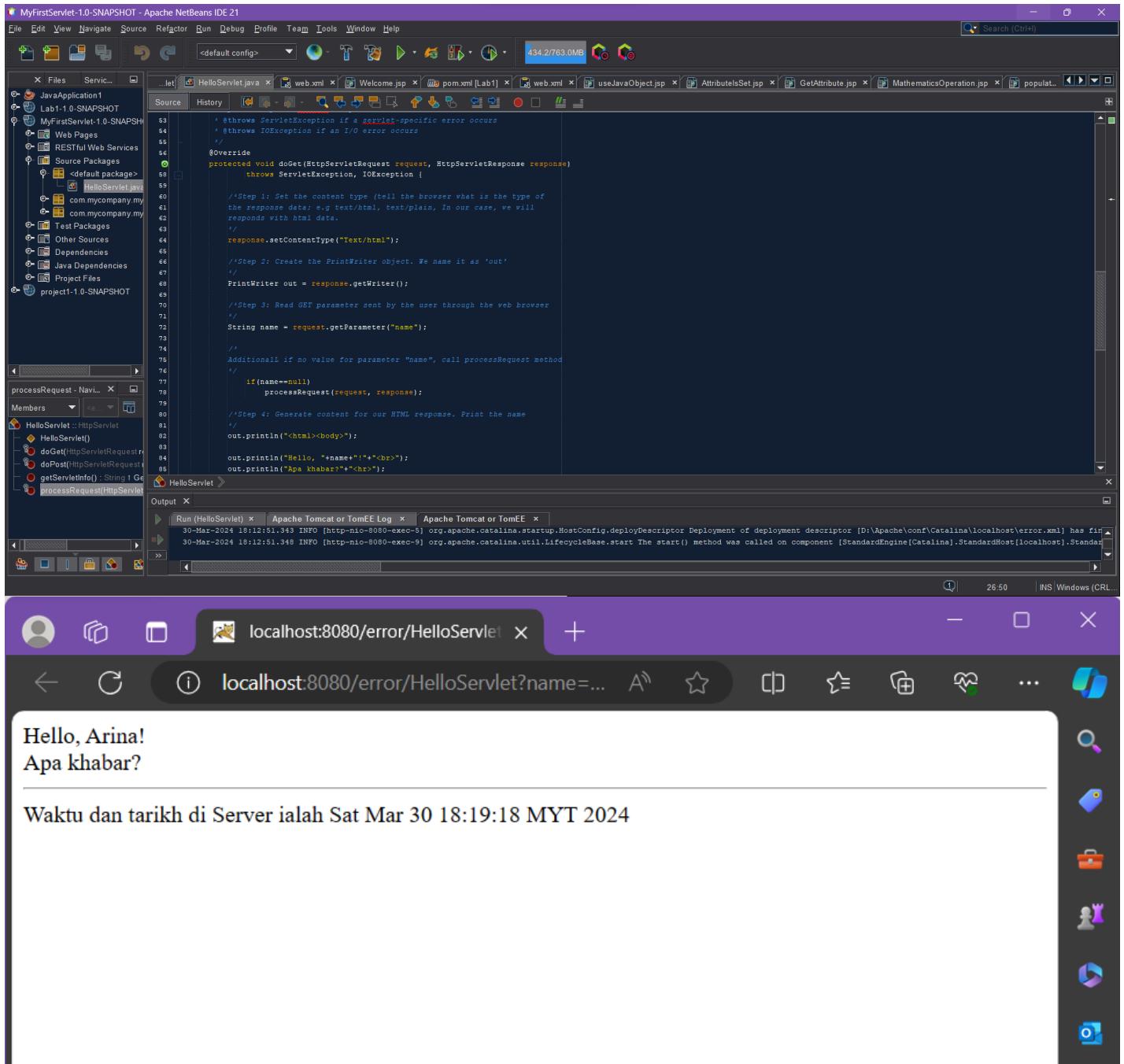
13.



The screenshot shows the Apache NetBeans IDE 21 interface. The title bar reads "MyFirstServlet-1.0-SNAPSHOT - Apache NetBeans IDE 21". The main window displays the "HelloServlet.java" file under the "Source" tab. The code implements a servlet named "HelloServlet" that processes both GET and POST requests. It prints a welcome message to the browser. The code is annotated with Javadoc comments and standard Java syntax. The "Output" panel at the bottom shows deployment logs for Apache Tomcat or TomEE, indicating successful deployment and startup. The left sidebar shows the project structure with packages like "com.mycompany.my" and files like "pom.xml" and "web.xml".

```
/*
 * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
 * methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet Saya Yang Pertama</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Hello, Servlet!</h1>");
        out.println("<h2>Servlet HelloServlet at " + request.getContextPath() + "</h2>");
        out.println("</body>");
        out.println("</html>");
    }
}
```





Task 6: Writing a Simple JSP Program

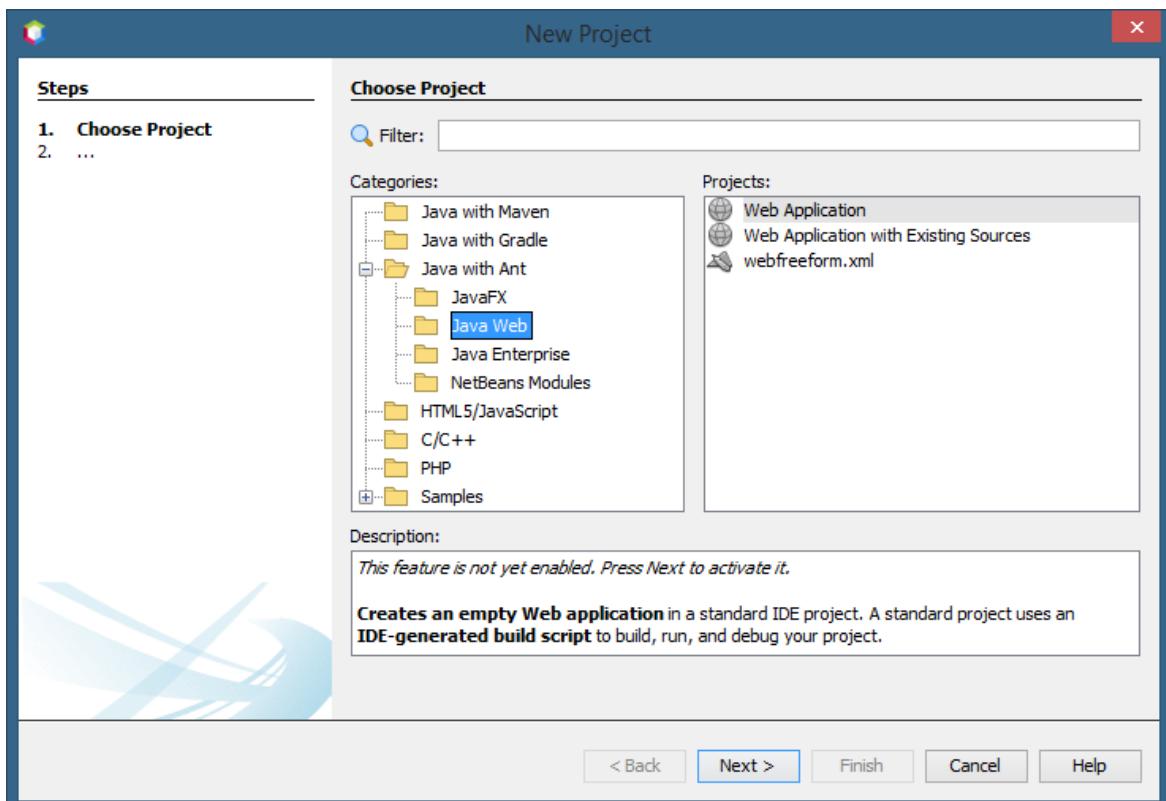
Objective : Writing a simple plain JSP program

Problem

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

Estimated time : 15 minutes

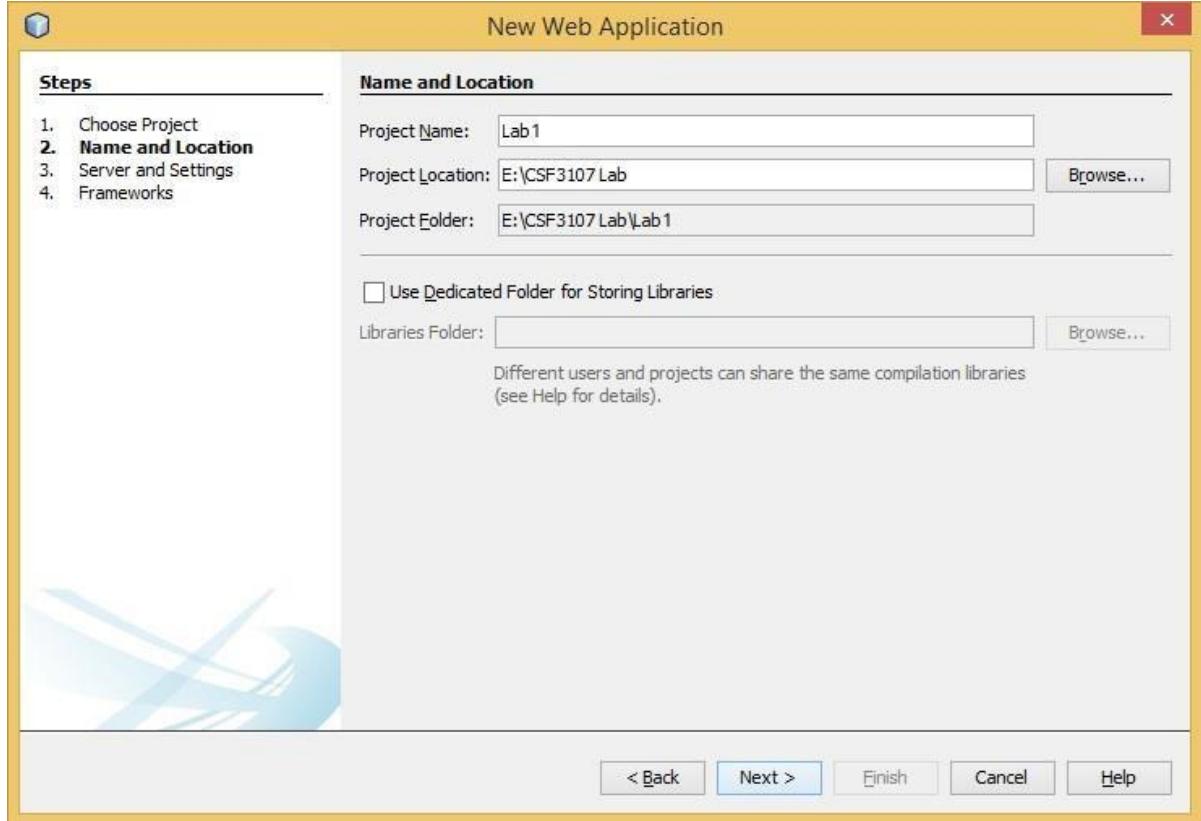
1. Create a directory C:\[MeticNumber] Lab
2. Go to C:\ [MeticNumber] Lab's directory and create sub-directory as Lab 1 - JSP.
3. Open your NetBeans.
4. Go to File -> New Project
5. Select Select Java with Ant -> Java Web -> Web Application and click Next.



6. Click the Next button.

7. Type Project Name: *Lab1*.

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

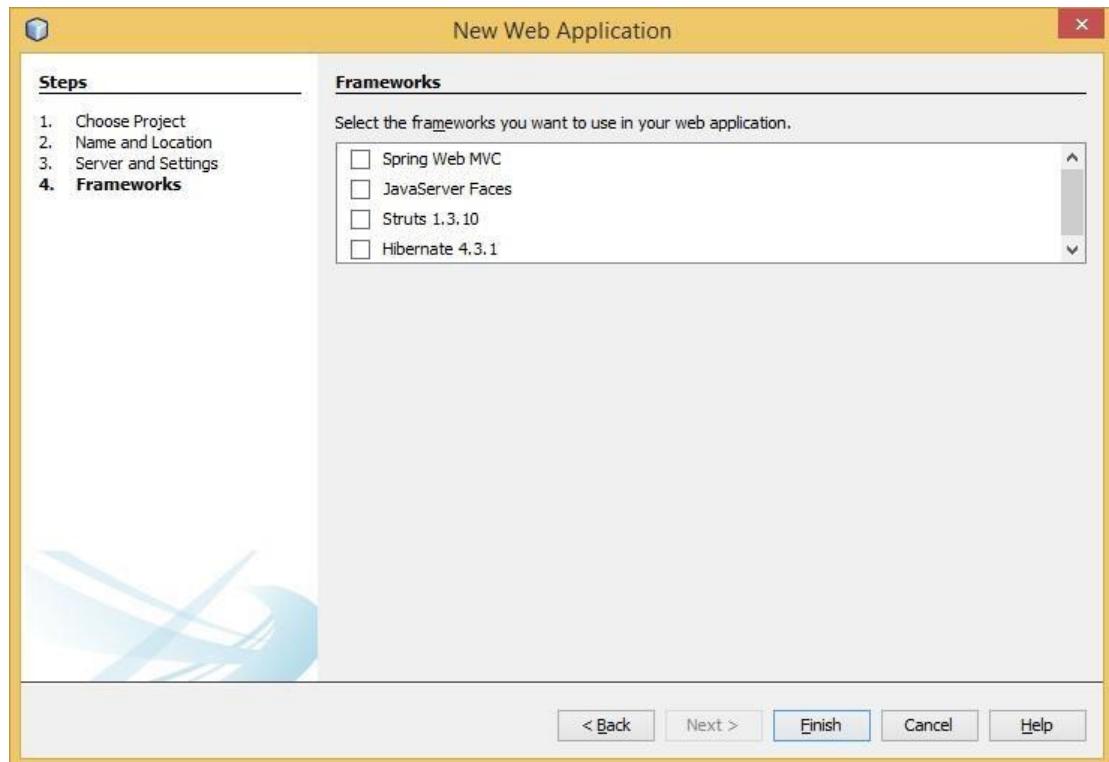


9. Click the *Next* button.

10. Select Server: *Apache Tomcat or TomEE*

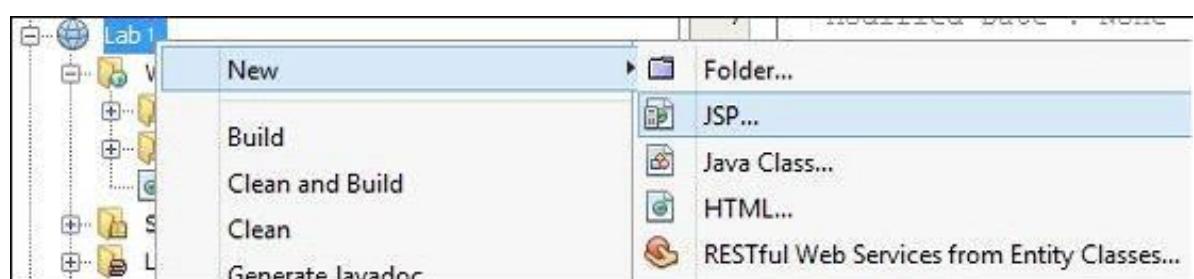
11. Select Java EE Version: *Java EE 6 Web.*

12. Click the Next button.

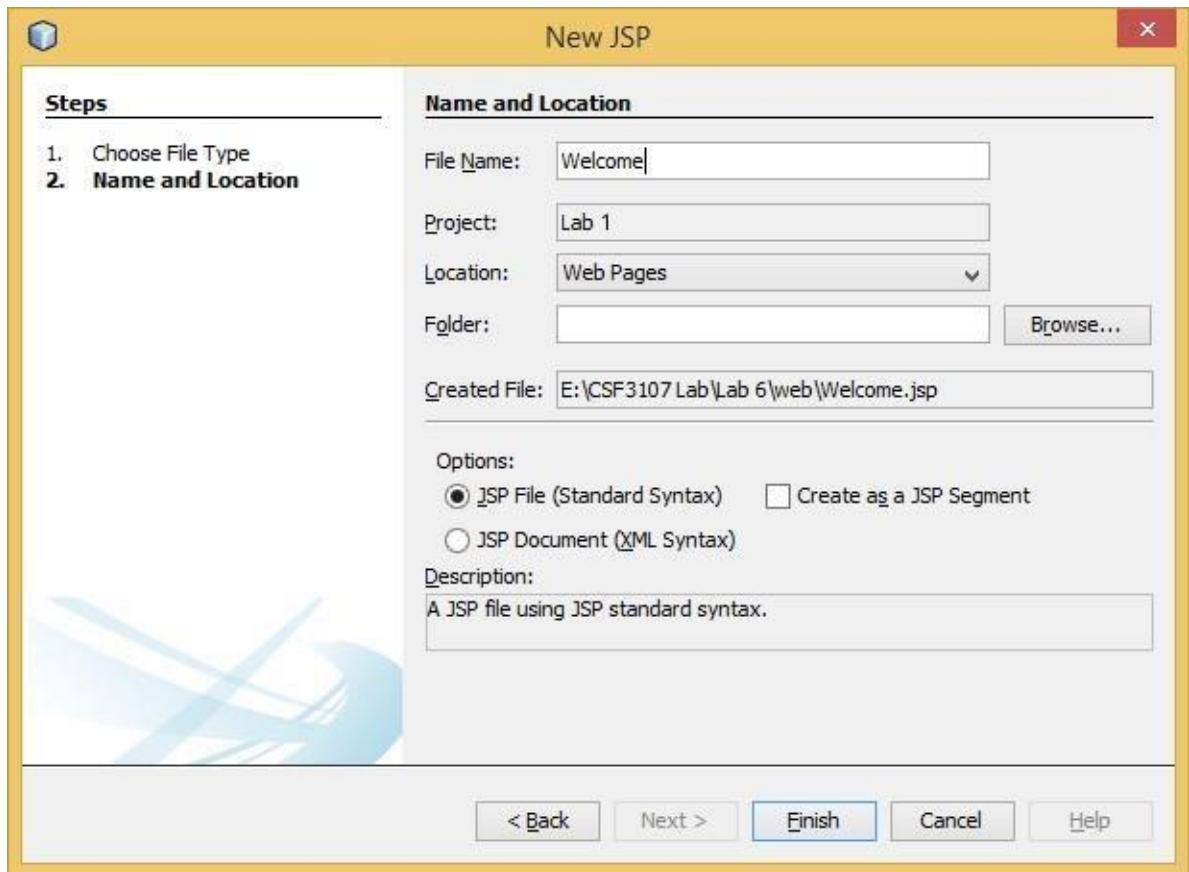


13. Click the Finish button.

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*

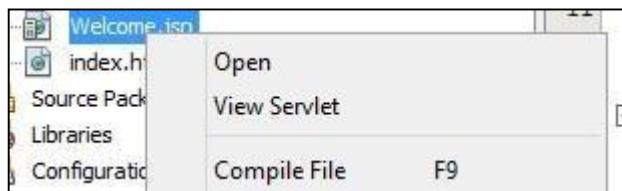
18. Type header1 as *Welcome to [MATRICNUMBER]...!*

```
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>
```

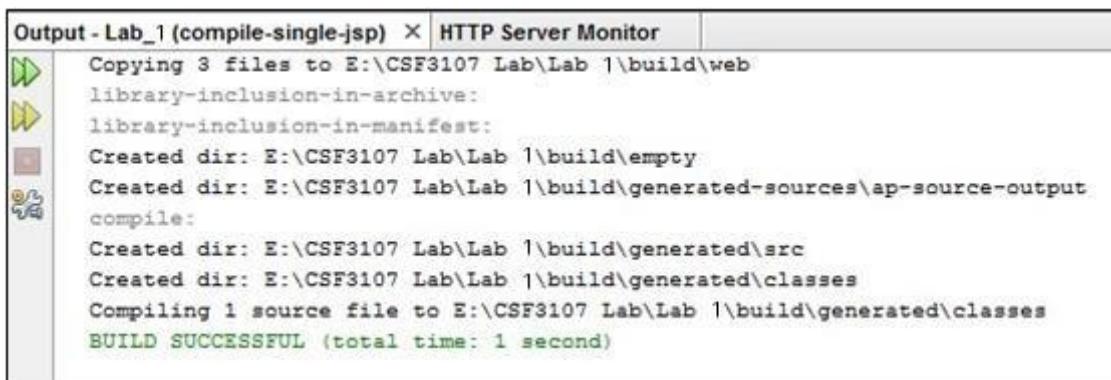
19. Click *SaveAll* icon



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



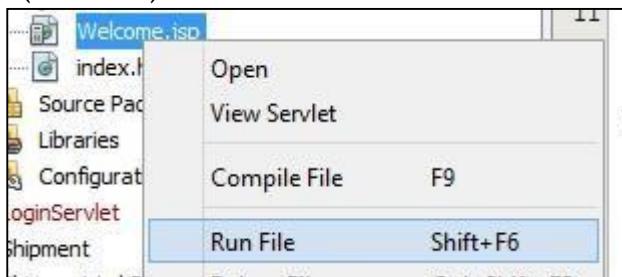
21. You will get notification message the bottom of Netbeans IDE with the green colour.



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.

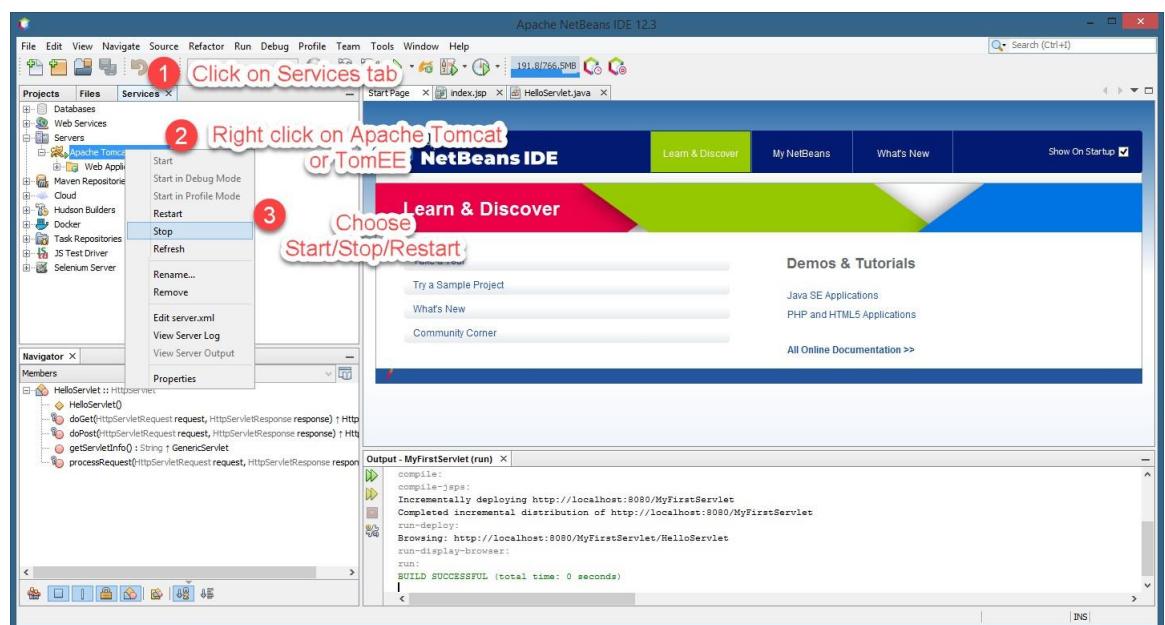
26. Go to the Project's tab. Then right click *Welcome.jsp* file and click *Run File* (Shift+F6).



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?

Writing a simple JSP program to get the output in the browser.

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

Java Server Pages is a server-side technology which is used for creating web applications. It also used to create dynamic web content. It also consist of both HTML tags and JSP tags.

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?

Servlet is a java code while JSP is a HTML-based compilation code.

Source code

The screenshot shows the Apache NetBeans IDE interface with the following details:

- Project Structure:** JavaApplication1 > Lab1-1.0-SNAPSHOT > Web Pages > META-INF > AttributesSet.jsp.
- Code Editor:** The `Welcome.jsp` file is open, displaying JSP code:

```
<%--  
Document : Welcome  
Created on : 28 Mar 2024, 4:19:48 pm  
Author : rynaa  
--%>  
<%@page contentType="text/html" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
    <head>  
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
        <title>S65361 - Web Programming 2</title>  
    </head>  
    <body>  
        <h1>Welcome to CSM3023...!</h1>  
    </body>  
</html>
```
- Navigator:** Shows the structure of the `Welcome.jsp` file.
- Output:** Shows logs from Apache Tomcat or TomEE Log:

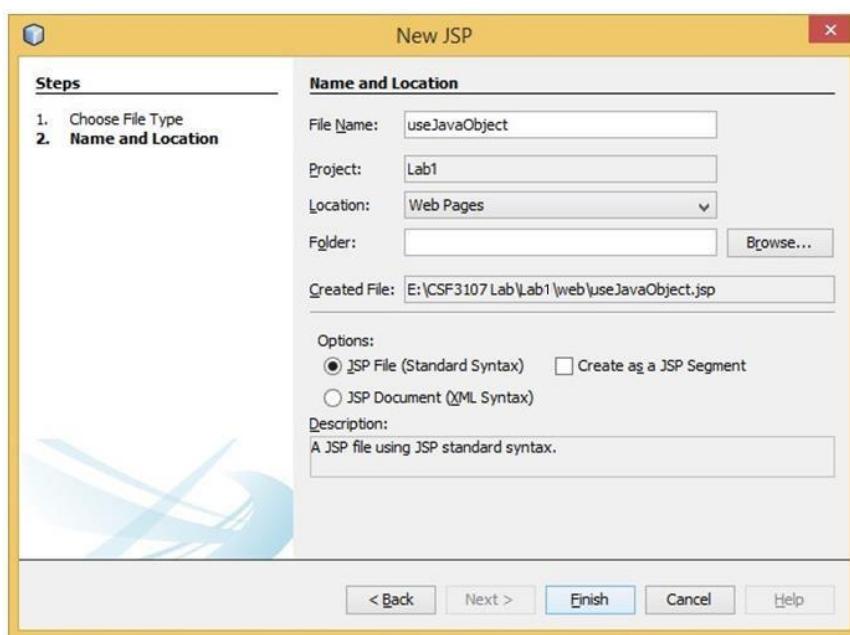
```
startPath=/Lab1  
OK - Started application at context path [/Lab1]
```
- Browser Preview:** A separate window shows the rendered output of the `Welcome.jsp` page: **Welcome to CSM3023...!**

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

Objective	: Using Java's object in JSP page.
Problem	: Display the current date, perform auto refresh header in
Description	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>
```

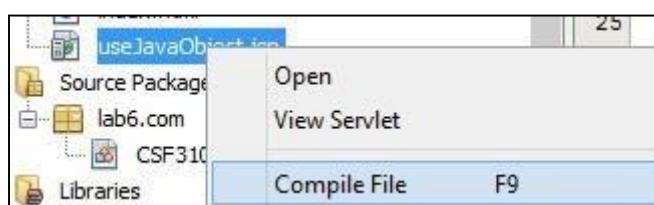
8. Continue writing a code to perform auto refresh header.

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

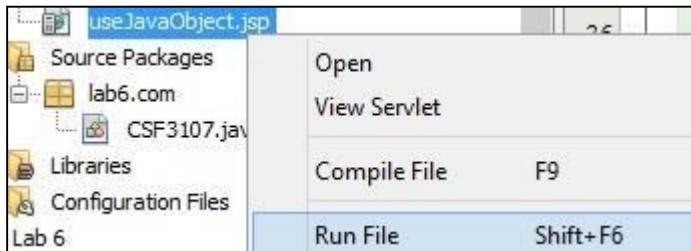
%>
```

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?

Use Java Reference Datatype/Class Wrapper in JSP such as Date and time.

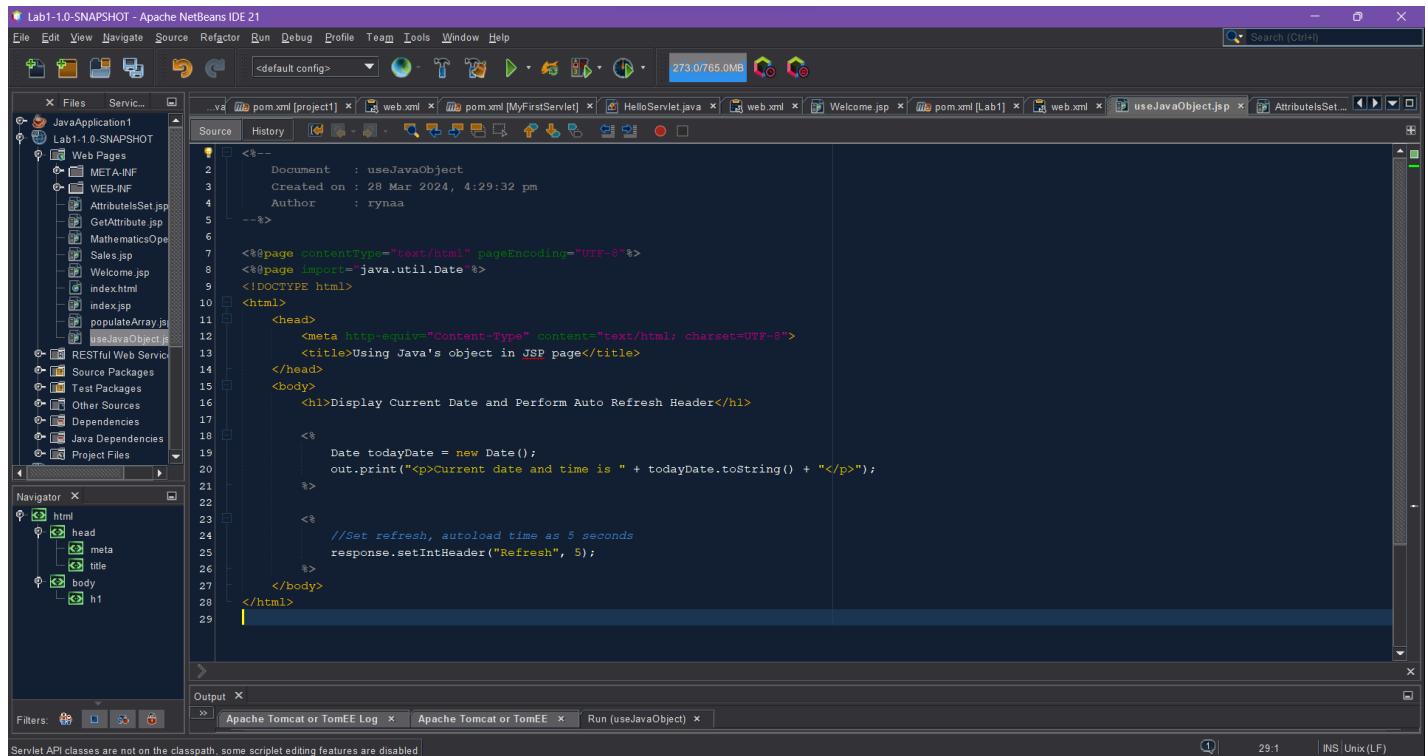
2. What is Java Scriptlet?

Scriptlet is a code used to contain any code fragment that is valid for the scripting language used in a page. The syntax for a scriptlet is <% scripting-language-statements %>.

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

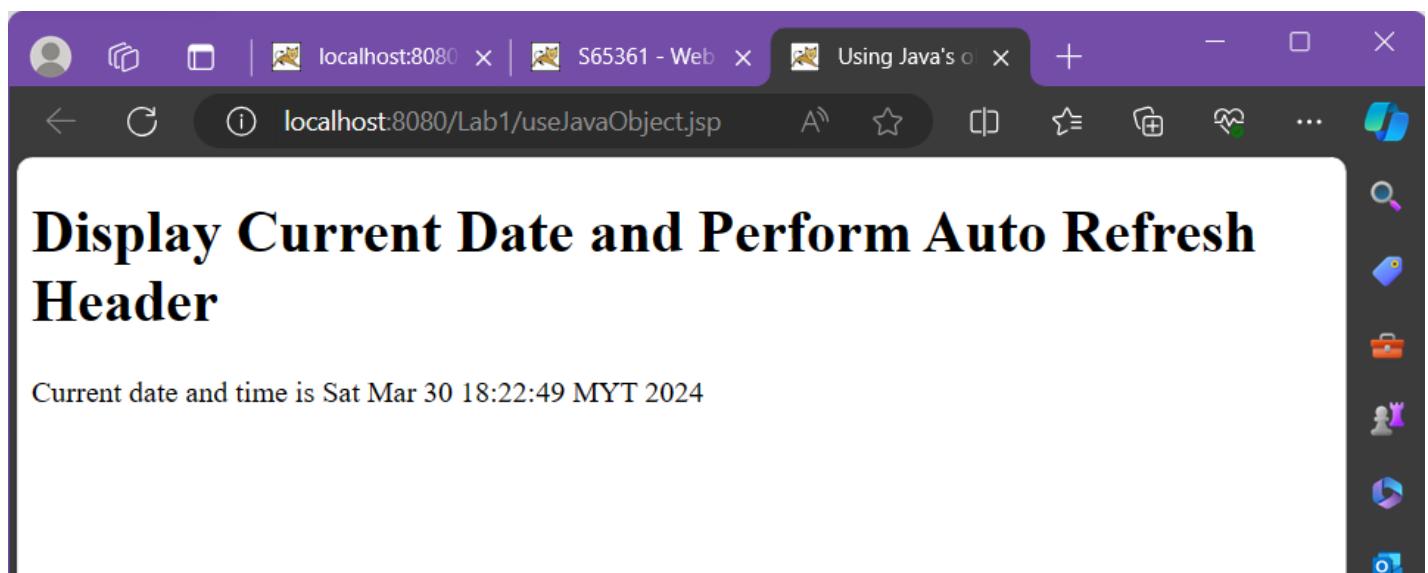
By using JSP expressions, JSP scriptlets and JSP declarations.

Source code



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Title Bar:** Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21
- Toolbar:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Project Explorer:** JavaApplication1, Lab1-1.0-SNAPSHOT, Web Pages, RESTful Web Service, Source Packages, Test Packages, Other Sources, Dependencies, Java Dependencies, Project Files.
- Code Editor:** The main editor window displays the JSP file `useJavaObject.jsp`. The code includes JSP tags, Java code, and HTML output. It prints the current date and time and sets an auto-refresh header.
- Navigator:** Shows the structure of the `html` section, including `head`, `meta`, and `body` sections.
- Output:** Shows logs for Apache Tomcat or TomEE Log, Apache Tomcat or TomEE, and Run (useJavaObject).

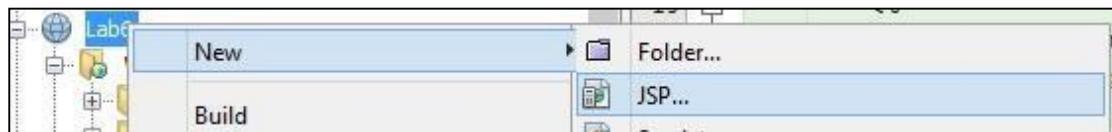


Task 8: Using JSP Implicit object in JSP page

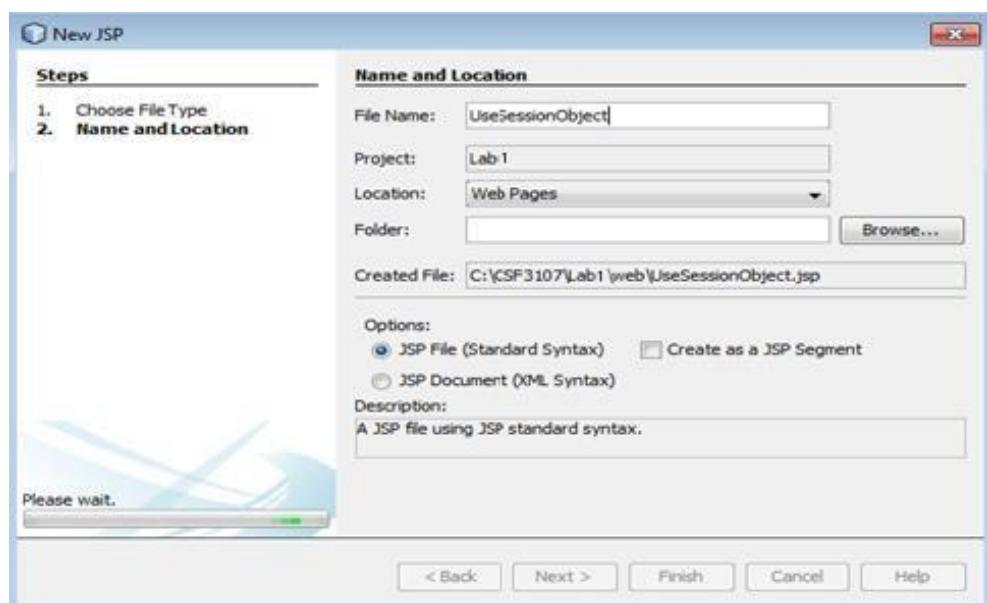
Objective	: Using JSP Implicit object (Session) in JSP page.
Problem	: Using Session object, perform simple Mathematics
Description	operations in JSP's page.

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 *AttributesSet*.



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

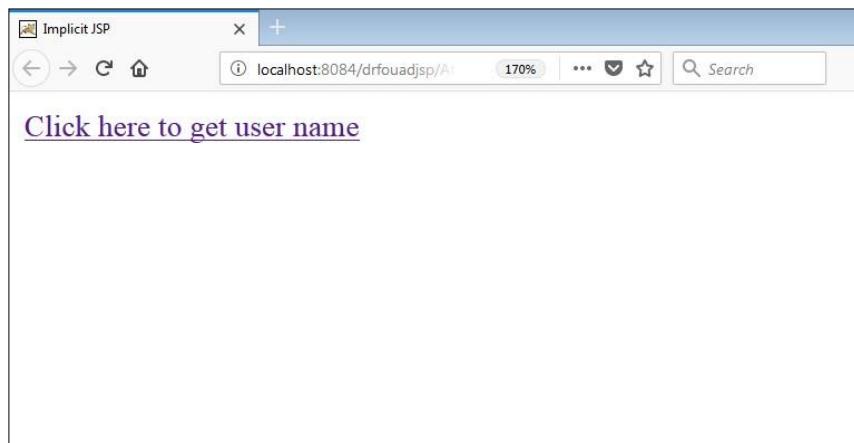
```
<%-->
Document : jsp
Created on : 20-Feb-2018
Author : Dr. Faizah Aplop
--%>

<%@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 *AttributelsSet.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;

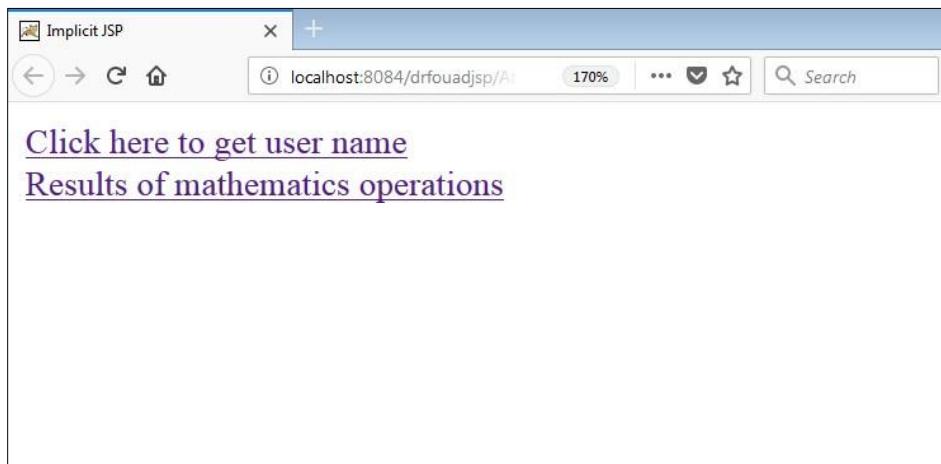
//Find square root for variable num1..
squareroot = (double)(Math.sqrt(num1));

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?

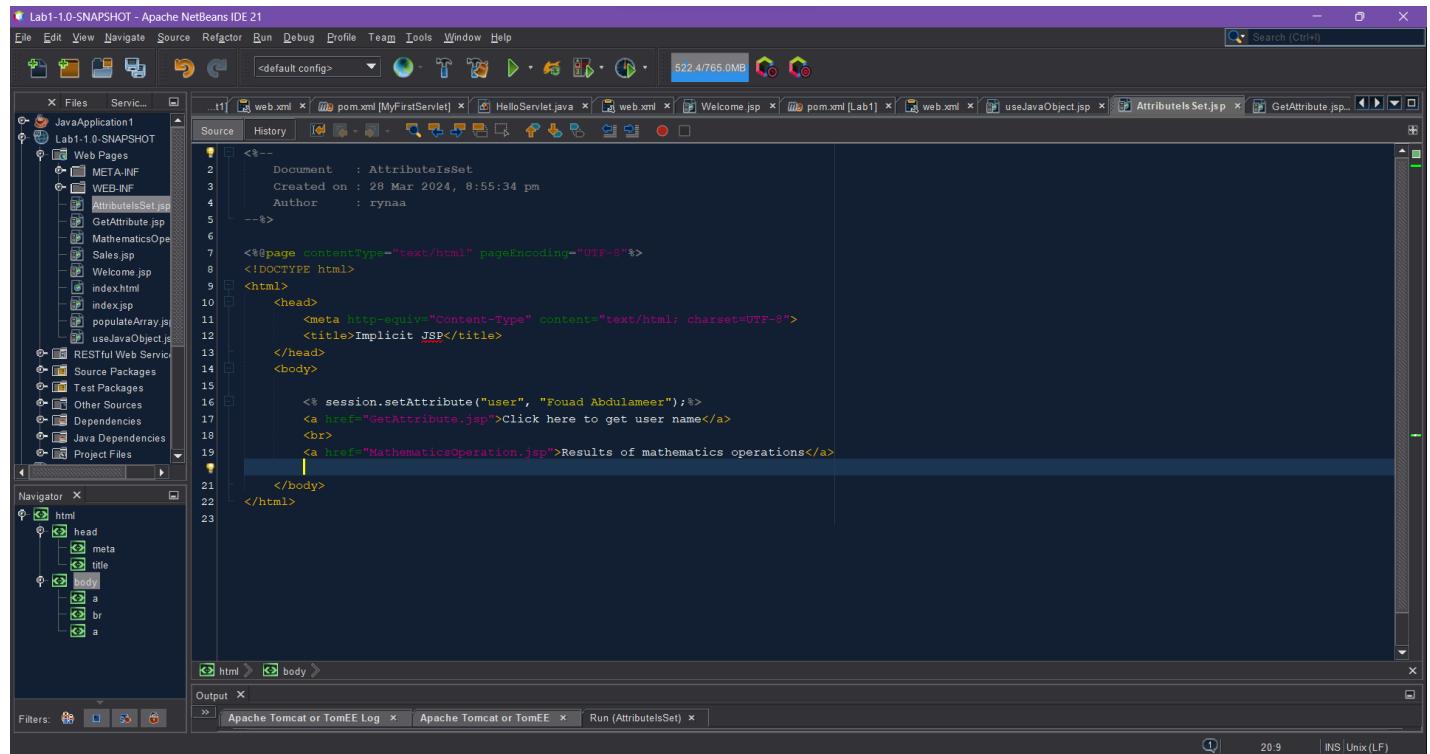
Use session Attributes

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?

It will return 'null'.

Source code

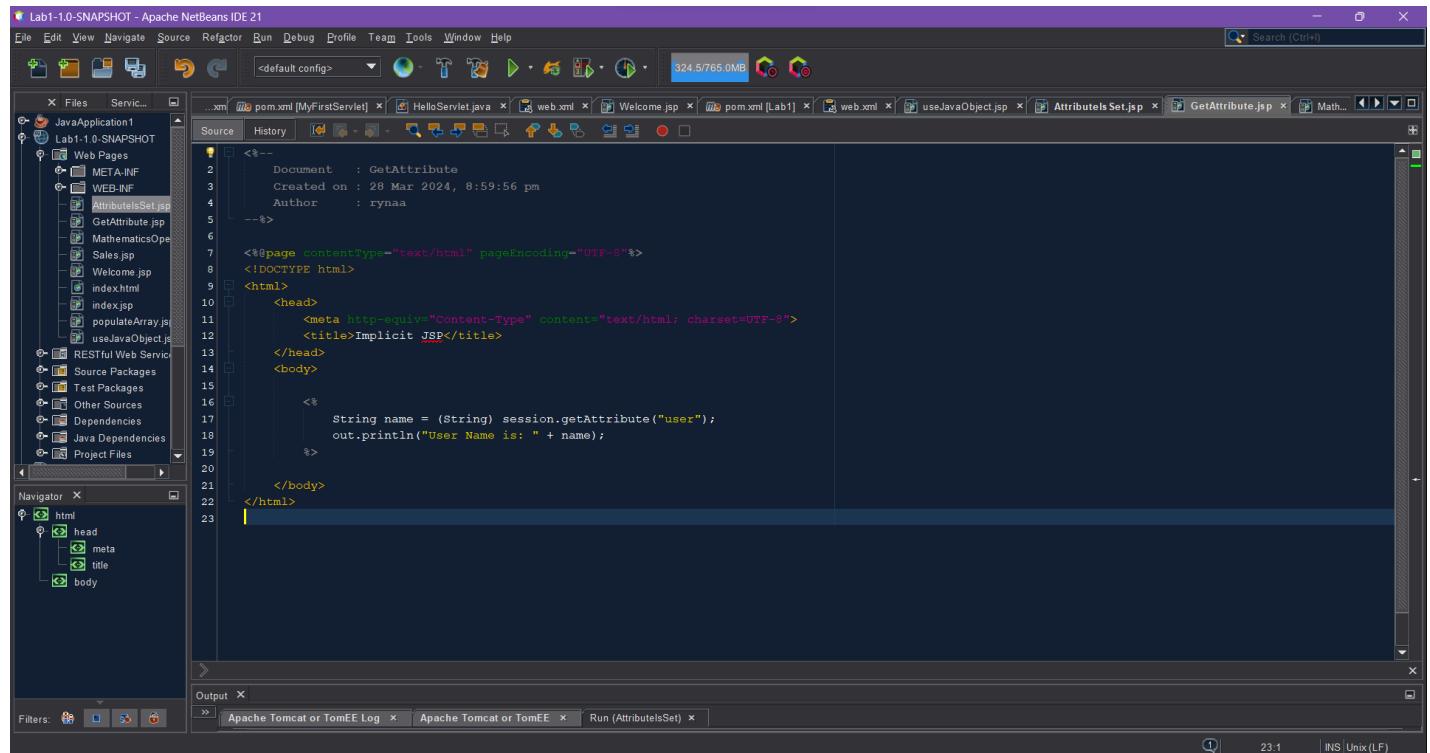
AttributesSet.jsp



The screenshot shows the Apache NetBeans IDE 21 interface with the project "Lab1-1.0-SNAPSHOT" open. The left pane displays the project structure under "JavaApplication1". The central pane shows the source code for "AttributesSet.jsp". The code is a JSP page that includes session attributes and links to other JSP files.

```
<%--  
Document : AttributeIsSet  
Created on : 28 Mar 2024, 8:55:34 pm  
Author : rynaa  
--%>  
<%@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>  
        <br>  
        <a href="MathematicsOperation.jsp">Results of mathematics operations</a>  
    </body>  
</html>
```

GetAttribute.jsp



The screenshot shows the Apache NetBeans IDE 21 interface with the project "Lab1-1.0-SNAPSHOT" open. The left pane displays the project structure under "JavaApplication1". The central pane shows the source code for "GetAttribute.jsp". The code is a JSP page that retrieves the session attribute "user" and prints it to the output.

```
<%--  
Document : GetAttribute  
Created on : 28 Mar 2024, 8:59:56 pm  
Author : rynaa  
--%>  
<%@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>
```

MathematicsOperations.jsp

The screenshot shows the Apache NetBeans IDE interface with the title "Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21". The main window displays the source code for "MathematicsOperations.jsp". The code performs basic arithmetic operations (addition, multiplication) and calculates the square root of a number. It uses Java's Math class and a Formatter for output. The Navigator panel on the left shows the structure of the project, including files like "AttributesSet.jsp", "GetAttribute.jsp", "MathematicsOperations.jsp", "Sales.jsp", "Welcome.jsp", "index.html", "index.jsp", "populateArray.js", and "useJavaObject.js". The Output panel at the bottom shows logs for Apache Tomcat or TomEE.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.math.*"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Mathematics Operations</title>
    </head>
    <body>

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

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

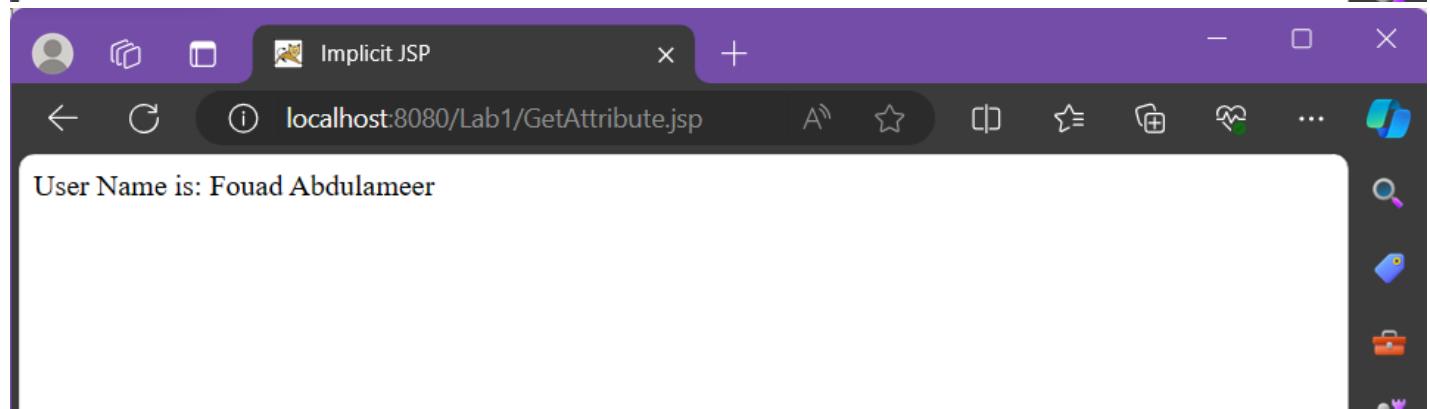
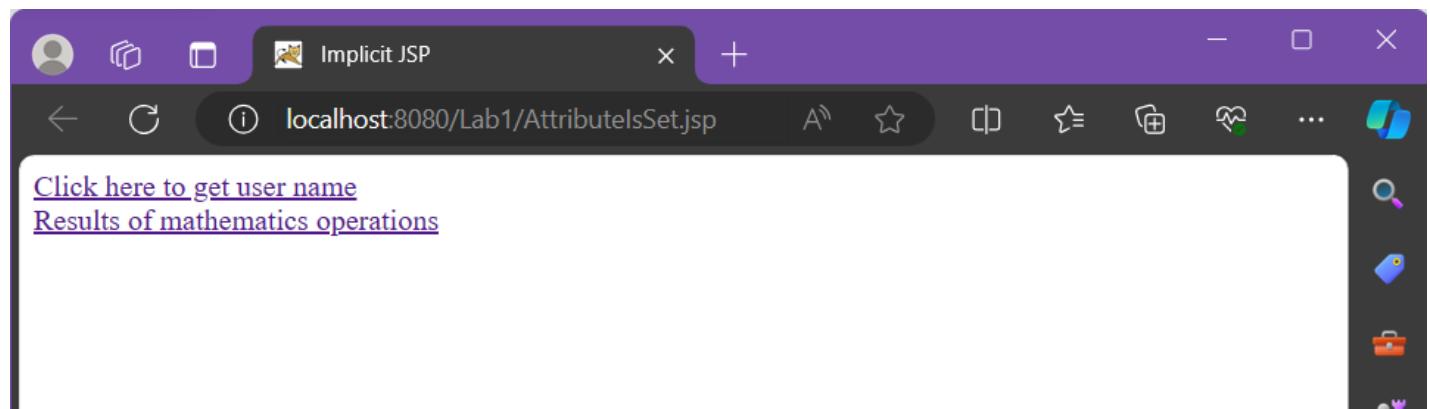
            //perform basic arithmatic operations...
            addition_output = num1 + num2;
            multiply_output = num1 * num2;

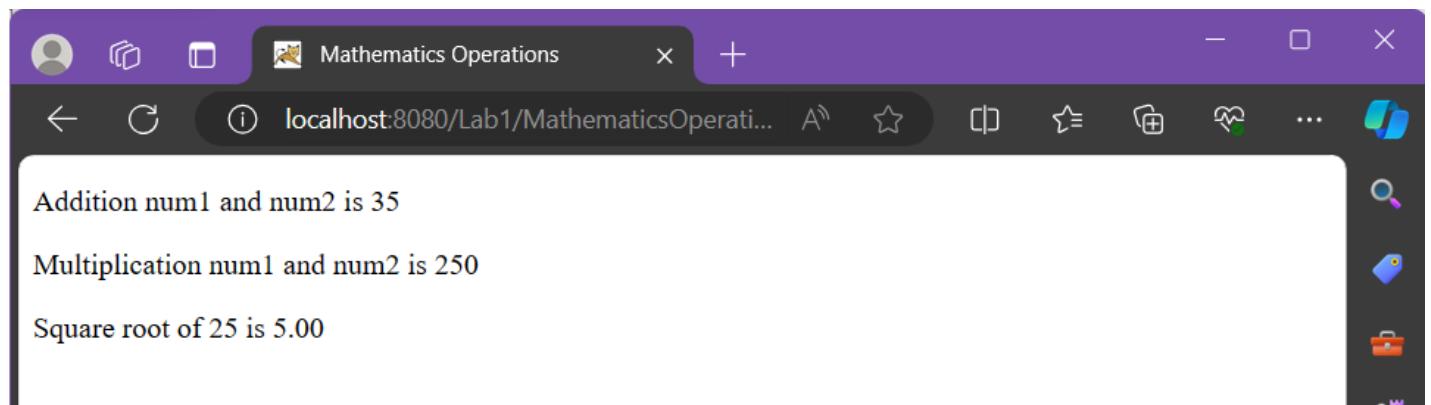
            //Find square root for variable num1...
            squareroot = (double)(Math.sqrt(num1));

            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>");

        %>
    </body>
</html>
```





Task 9: Populate Array values into HTML's Table

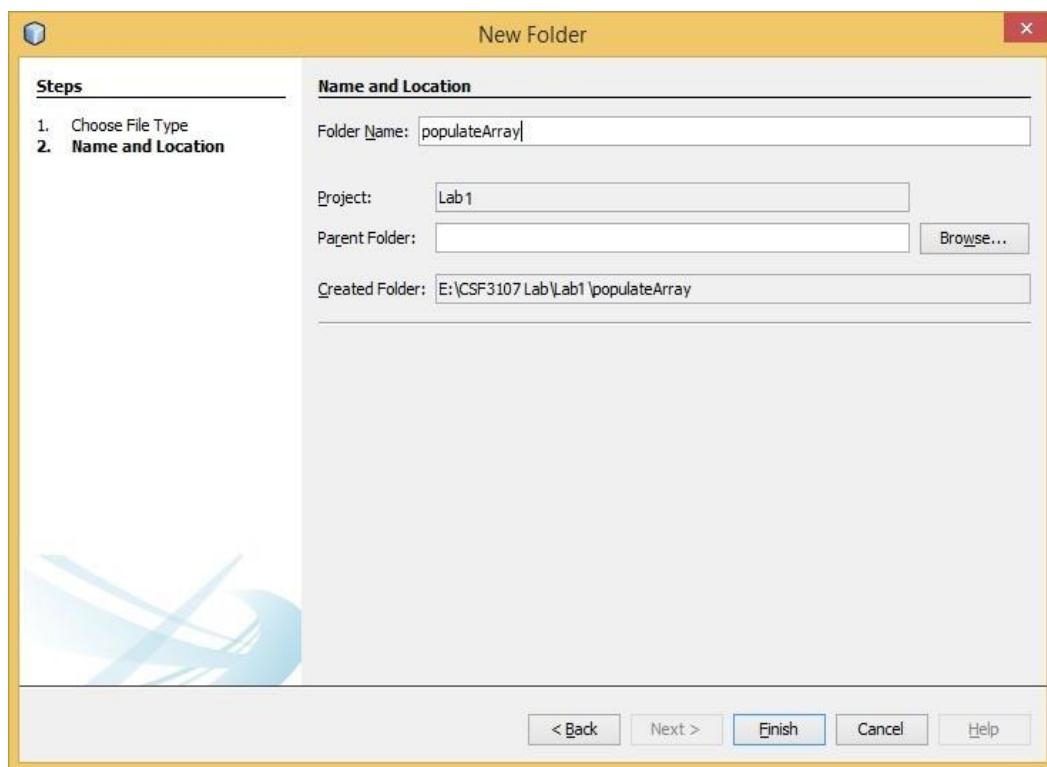
Objective : Read Java array and populate it into HTML's table.

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 Scriptlet 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:

localhost:8084/Lab1/populateArray.jsp

Read Java array and populate it into HTML's table

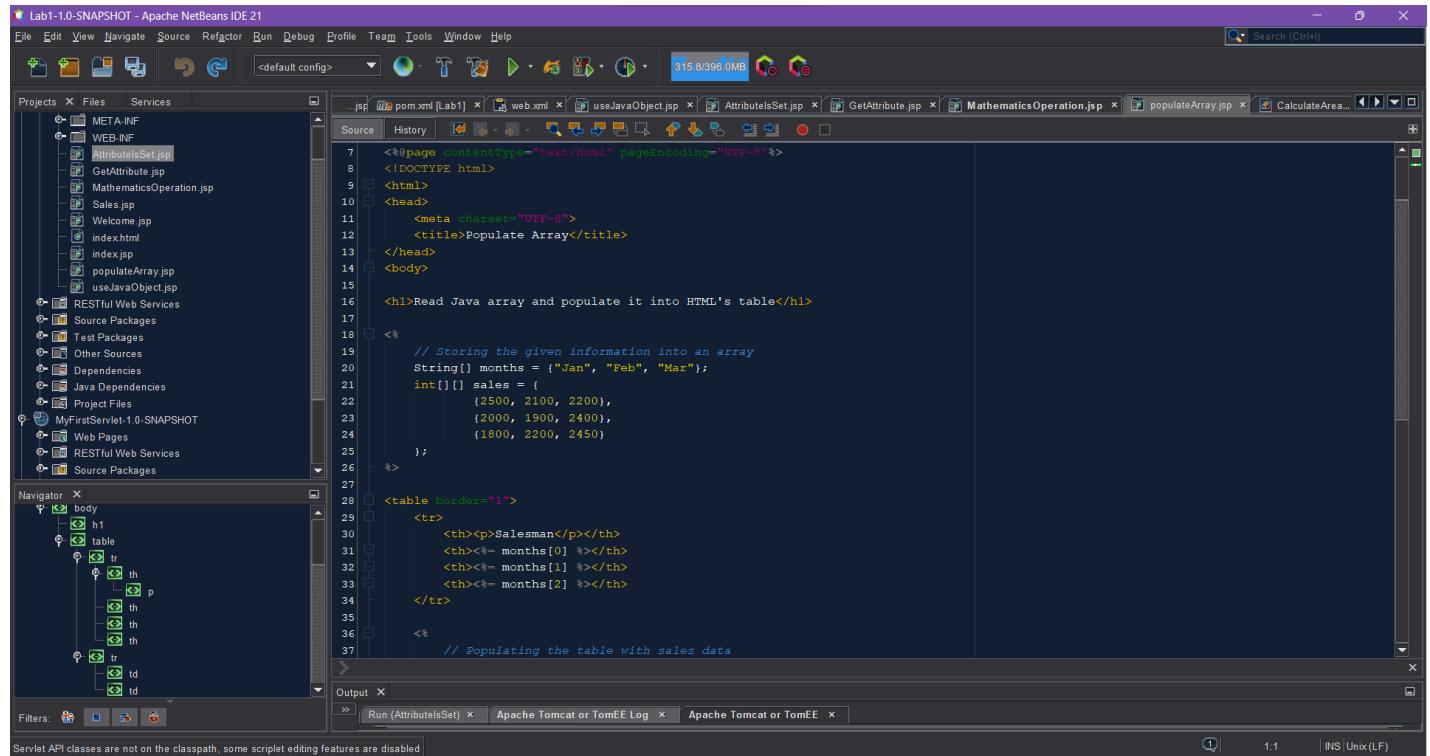
Salesman	Jan	Feb	Mac
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

©2016-Mohamad Nor

Reflection

1. Write a sample syntax to declare 2D Java array.
`String[][] StringArray = new String[2][4];`
2. Define a sequence of steps on how you accomplish Task 7.
3. What is the difference between HTML's page and JSP's page?
 HTML pages are processed on the client-side by web browsers. It was static and do not involve server-side processing while JSP pages are processed on the server-side by a JSP container such as Tomcat.

Source code



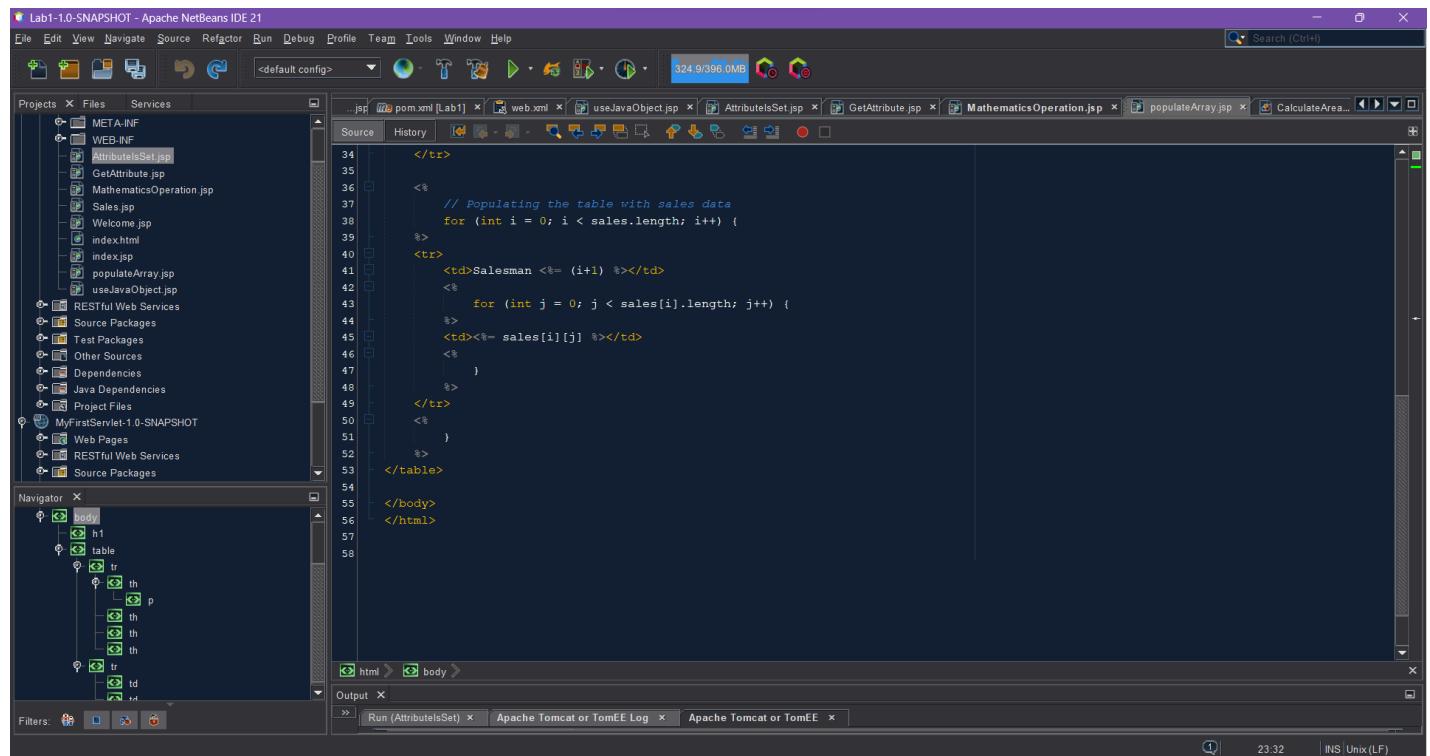
The screenshot shows the Apache NetBeans IDE 21 interface with a Java JSP file open. The code reads a Java array of sales data and populates it into an HTML table.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Populate Array</title>
    </head>
    <body>

        <h1>Read Java array and populate it into HTML's table</h1>

        <%
            // Storing the given information into an array
            String[] months = {"Jan", "Feb", "Mar"};
            int[][] sales = {
                {2500, 2100, 2200},
                {2000, 1900, 2400},
                {1800, 2200, 2450}
            };
        %>

        <table border="1">
            <tr>
                <th><p>Salesman</p></th>
                <th><%= months[0] %></th>
                <th><%= months[1] %></th>
                <th><%= months[2] %></th>
            </tr>
            <%
                // Populating the table with sales data
                for (int i = 0; i < sales.length; i++) {
                    <tr>
                        <td>Salesman <%= (i+1) %></td>
                        <%
                            for (int j = 0; j < sales[i].length; j++) {
                                <td><%= sales[i][j] %></td>
                            }
                        <%
                    </tr>
                }
            </table>
        </body>
    </html>
```



The screenshot shows the continuation of the Java JSP code from the previous screenshot. It completes the table population loop and ends the HTML structure.

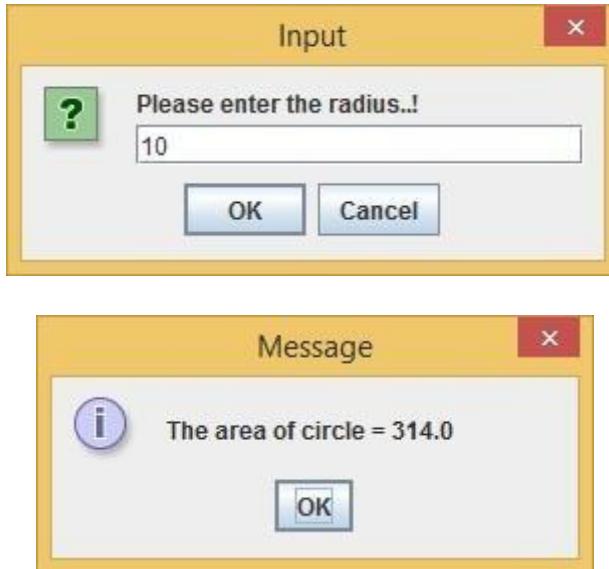
```
</tr>
<%
    // Populating the table with sales data
    for (int i = 0; i < sales.length; i++) {
        <tr>
            <td>Salesman <%= (i+1) %></td>
            <%
                for (int j = 0; j < sales[i].length; j++) {
                    <td><%= sales[i][j] %></td>
                }
            <%
        </tr>
    }
</table>
</body>
</html>
```

The screenshot shows a Microsoft Edge browser window with a purple title bar. The title bar has two tabs: "Mathematics Operations" and "Populate Array". The address bar shows the URL "localhost:8080/Lab1/populateArray.jsp". The main content area displays a heading "Read Java array and populate it into HTML's table" followed by a table with four columns: Salesman, Jan, Feb, and Mar. The table contains four rows of data for Salesman 1, 2, and 3, with Salesman 3 being the last row. A red dot is visible near the bottom center of the page.

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

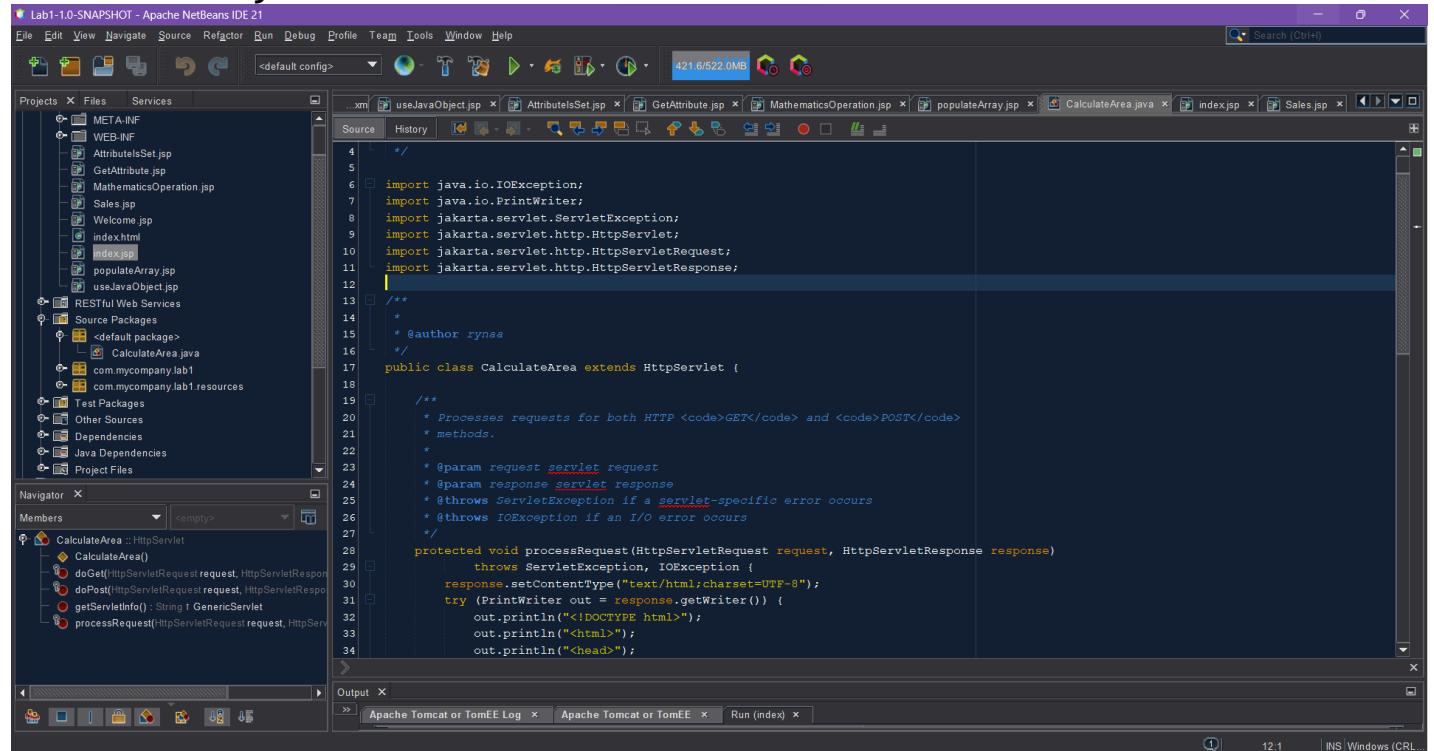
Exercise

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.

Source code calculateArea.java



```
4  /*
5   * To change this license header, choose License Headers in Project Properties.
6   * To change this template file, choose Tools | Templates
7   * and open the template in the editor.
8   */
9  package com.mycompany.lab1;
10
11  import java.io.IOException;
12  import java.io.PrintWriter;
13  import jakarta.servlet.ServletException;
14  import jakarta.servlet.http.HttpServlet;
15  import jakarta.servlet.http.HttpServletRequest;
16  import jakarta.servlet.http.HttpServletResponse;
17
18 /**
19  * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
20  * methods.
21  *
22  * @param request servlet request
23  * @param response servlet response
24  * @throws ServletException if a servlet-specific error occurs
25  * @throws IOException if an I/O error occurs
26  */
27 protected void processRequest(HttpServletRequest request, HttpServletResponse response)
28 throws ServletException, IOException {
29     response.setContentType("text/html;charset=UTF-8");
30     try (PrintWriter out = response.getWriter()) {
31         out.println("<!DOCTYPE html>");
32         out.println("<html>");
33         out.println("<head>");
34     }
35 }
```

Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21

```

protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("    <head>");
        out.println("        <title>Servlet CalculateArea</title>");
        out.println("    </head>");
        out.println("    <body>");
        out.println("        <h1>Servlet CalculateArea at " + request.getContextPath() + "</h1>");
        out.println("    </body>");
        out.println("    </html>");
    }
}

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    // Retrieve the radius parameter from the request
    String radiusStr = request.getParameter("radius");
    // Parse the radius string to a double
}

```

Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21

```

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    // Retrieve the radius parameter from the request
    String radiusStr = request.getParameter("radius");
    // Parse the radius string to a double
    double radius = Double.parseDouble(radiusStr);
    // Calculate the area of the circle
    double area = Math.PI * radius * radius;

    // Output the result
    out.println("<html><body>");
    out.println("    <h2>Area of the circle: " + area + "</h2>");
    out.println("    </body></html>");
}

```

Index.jsp

The screenshot shows the Apache NetBeans IDE interface. The title bar reads "Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The toolbar has icons for file operations like Open, Save, and Find. The status bar at the bottom right shows "384.7/522.0MB" and the time "3:33".

The Projects tab in the left sidebar lists the project structure:

- META-INF
- WEB-INF
 - AttributesSet.jsp
 - GetAttribute.jsp
 - MathematicsOperation.jsp
 - Sales.jsp
 - Welcome.jsp
 - index.html
 - index.jsp
 - useJavaObject.jsp
 - useJavaObject.jsp
- RESTful Web Services
 - Source Packages
 - <default package>
 - CalculateArea.java
 - com.mycompany.lab1
 - com.mycompany.lab1.resources
- Test Packages
- Other Sources
- Dependencies
- Java Dependencies
- Project Files

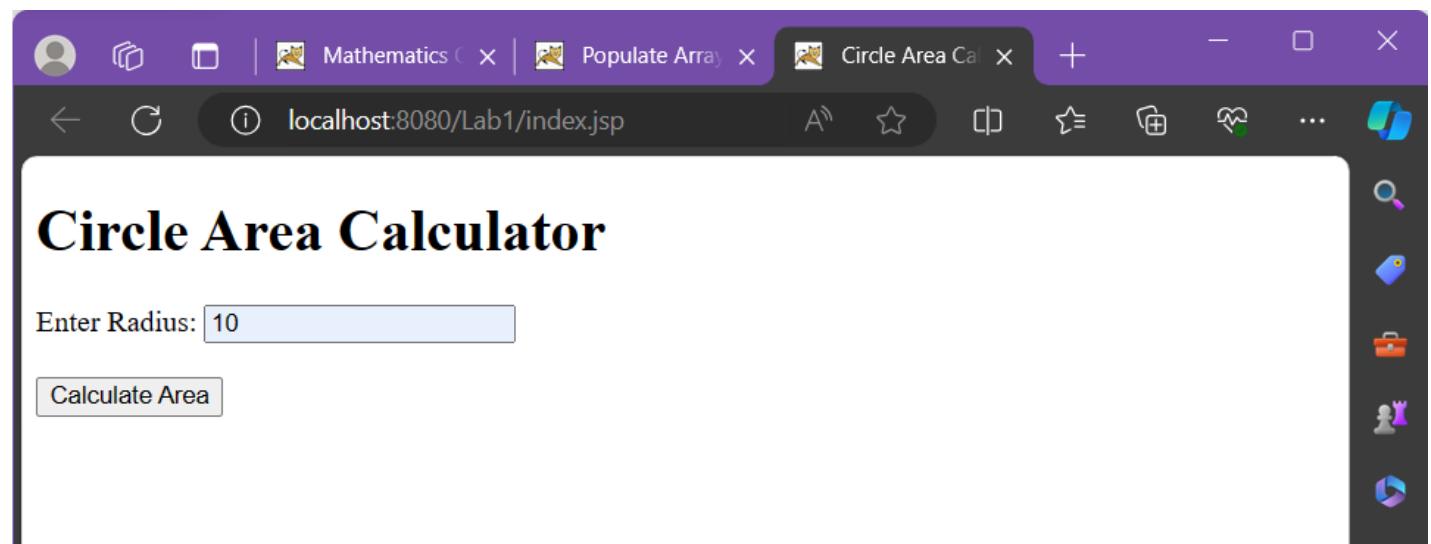
The Files tab is selected, showing the source code for index.jsp:

```
<%--  
 Document : index  
 Created on : 29 Mar 2024, 10:58:07 pm  
 Author : rynaa  
--%>  
<%@page contentType="text/html" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
    <head>  
        <meta charset="UTF-8">  
        <title>Circle Area Calculator</title>  
    </head>  
    <body>  
        <h1>Circle Area Calculator</h1>  
        <form action="CalculateArea" method="post">  
            Enter Radius: <input type="text" name="radius" required pattern="[0-9]+><br><br>  
            <input type="submit" value="Calculate Area">  
        </form>  
    </body>  
</html>
```

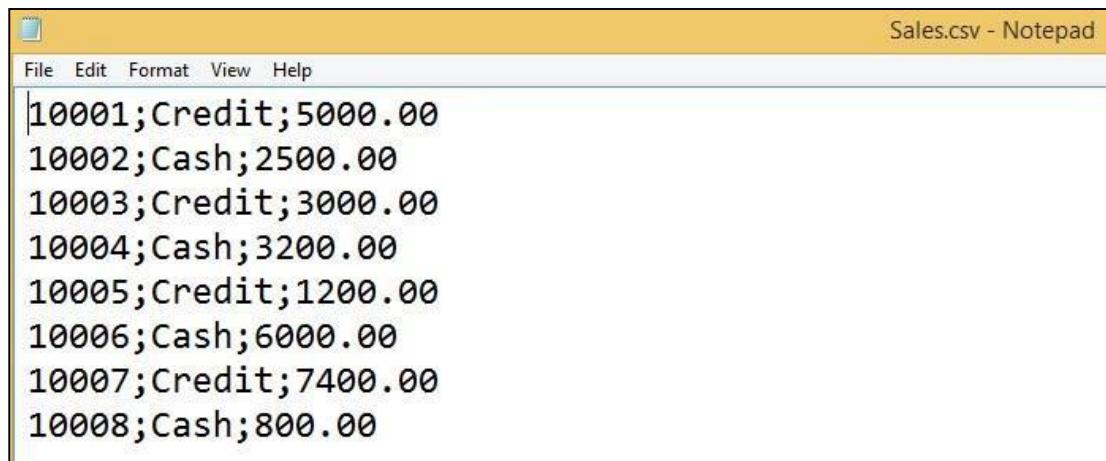
The Navigator tab shows the HTML structure:

- html
 - head
 - meta
 - title
 - body
 - h1
 - form
 - input
 - br
 - br
 - input

The Output tab shows logs for Apache Tomcat or TomEE.



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



The screenshot shows a Windows Notepad window with a yellow title bar containing the text "Sales.csv - Notepad". Below the title bar is a menu bar with options: File, Edit, Format, View, and Help. The main body of the window contains eight lines of CSV data, each consisting of three fields separated by semicolons:

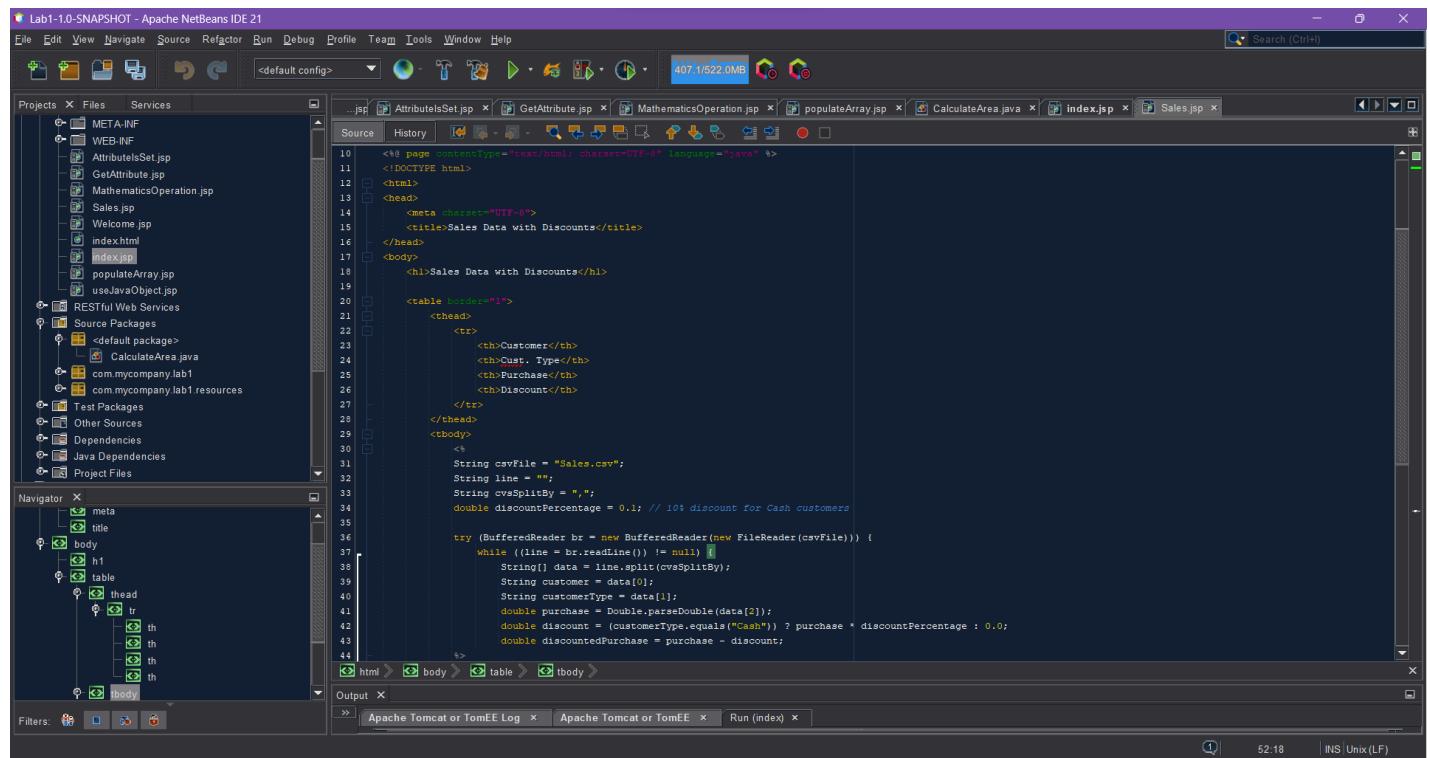
10001	Credit	5000.00
10002	Cash	2500.00
10003	Credit	3000.00
10004	Cash	3200.00
10005	Credit	1200.00
10006	Cash	6000.00
10007	Credit	7400.00
10008	Cash	800.00

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.

Source code



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Project Structure:** The left pane shows a project named "Lab1-1.0-SNAPSHOT - Apache NetBeans IDE 21". It contains several Java files (e.g., AttributesSet.jsp, GetAttribute.jsp, MathematicsOperation.jsp, Sales.jsp, Welcome.jsp, index.html, index.jsp, populateArray.jsp, useJavaObject.jsp) and a RESTful Web Services package.
- Source Editor:** The main editor area displays the content of the `Sales.jsp` file. The code reads a CSV file named "Sales.csv" and processes it to calculate discounts for cash customers.
- Navigator:** The bottom-left pane shows the HTML structure of the page, including meta, title, body, h1, and a table with its header and rows.
- Output:** The bottom-right pane shows the Apache Tomcat or TomEE Log and Run (index) tabs.

```

<%@ page contentType="text/html; charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Sales Data with Discounts</title>
    </head>
    <body>
        <h1>Sales Data with Discounts</h1>
        <table border="1">
            <thead>
                <tr>
                    <th>Customer</th>
                    <th>Cust. Type</th>
                    <th>Purchase</th>
                    <th>Discount</th>
                </tr>
            </thead>
            <tbody>
                <%
                    String csvFile = "Sales.csv";
                    String line = "";
                    String csvSplitBy = ",";
                    double discountPercentage = 0.1; // 10% discount for Cash customers

                    try (BufferedReader br = new BufferedReader(new FileReader(csvFile))) {
                        while ((line = br.readLine()) != null) {
                            String[] data = line.split(csvSplitBy);
                            String customer = data[0];
                            String customerType = data[1];
                            double purchase = Double.parseDouble(data[2]);
                            double discount = (customerType.equals("Cash")) ? purchase * discountPercentage : 0.0;
                            double discountedPurchase = purchase - discount;
                            ...
                    }
                %>
            </tbody>
        </table>
    </body>
</html>

```

