

**UNIVERSITI MALAYSIA TERENGGANU**

**CSM3023 WEB-BASED APPLICATION DEVELOPMENT**

**(K1)**

**BACHELOR OF COMPUTER SCIENCE (MOBILE**

**COMPUTING) WITH HONORS**

**LAB 4**

**SEMESTER II 2023/2024**

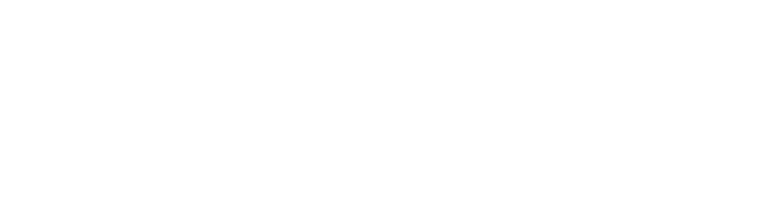
**Prepared for:**

DR. MOHAMAD NOR HASSAN

**Prepared by:**

NUR FADHILAH BINTI MOHD RAHMAT

(S67172)



**Week 3**

JSP: Scrip

t

let,

Expression &

Standard Actions

Web Programmin

g

2

Lecturers

PUSAT PENGAJIAN INFORMATIK DAN MATEMATIK GUNAAN

(

PPIMG), UNIVERSITI MALAYSIA TERENGGANU (UMT

)

Name:

Matric

#

:

Semester:

Lab:

Demonstrator:

## 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 |

## Table of Contents

[**Task 1**: Using JSP Scripting 6](#_Toc32376)

[**Task 2**: Using JSP (Scripting, Declaration and Expression) 7](#_Toc32377)

[**Task 3**: Using JSP Standard Action (Include and Param) 10](#_Toc32378)

[**Task 4**: Using JSP Standard Action (Forward) 14](#_Toc32379)

[**Task 5**: Use Java Scriptlet To Construct Business Logic 18](#_Toc32380)

**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 mengedar 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: Using JSP Scripting**

# Task 1: Using JSP Scripting

**O**

**bj**

**e**

**ct**

**i**

**ve**

**:**

J

SP S

c

r

i

p

t

l

e

t

a

n

d JSP

E

xpr

e

ss

io

n

i

n

a

pp

l

i

c

a

t

i

o

n

.

**P**

**r**

**o**

**b**

**l**

**e**

**m**

**D**

**e**

**s**

**c**

**r**

**i**

**p**

**t**

**i**

**o**

**n**

**:**

Prepare a simple int

erface to perform the following

payment

process;

i. If Customer Type is Normal Customer (assign

value as

“1”)

and Order Quantity > 100,

customer entitle 10% discount.

ii. If Customer Type is Privilege Customer (assign

value as

“2”) and Order Quantity > 100,

customer entitle 25%

discount.

iii. Order Quantity must be in number.

iv. Finally,

display the results.

**Es**

**t**

**ima**

**t**

**e**

**d t**

**i**

**me**

**:**

4

0

m

inute

s

1. Create Project *Lab3*.

2. Create a new HTML’s file.



1. Type file name as *customer.*

1. Prepare the following Graphical User Interface (GUI).

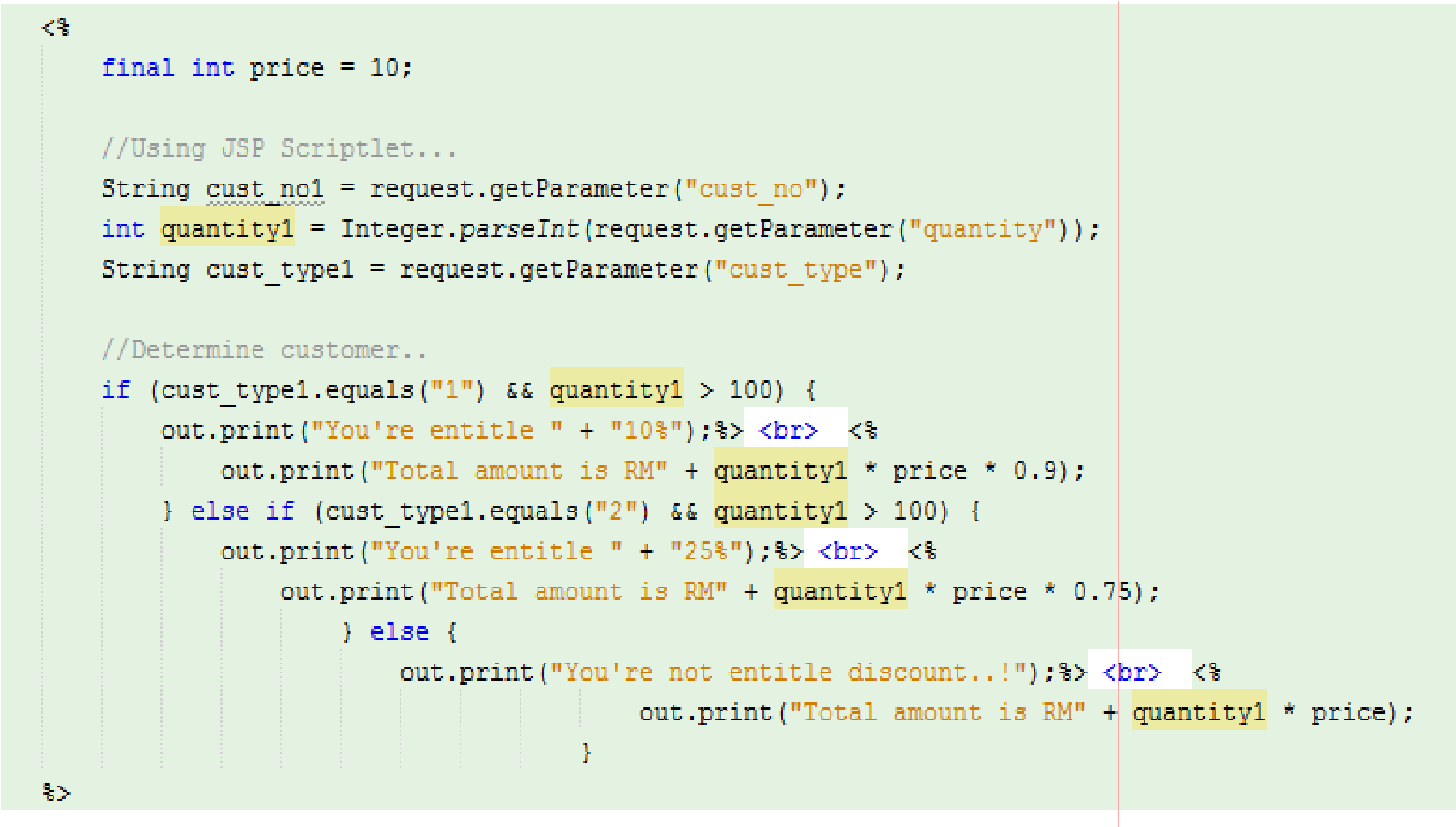


1. You must ensure the amount must be written as number.

### 6. The value for Normal Customer is “1” and Privilege Customer is “2”

1. Create a new file name known as *processCustomer.jsp*.

1. Define related variables and methods as below.

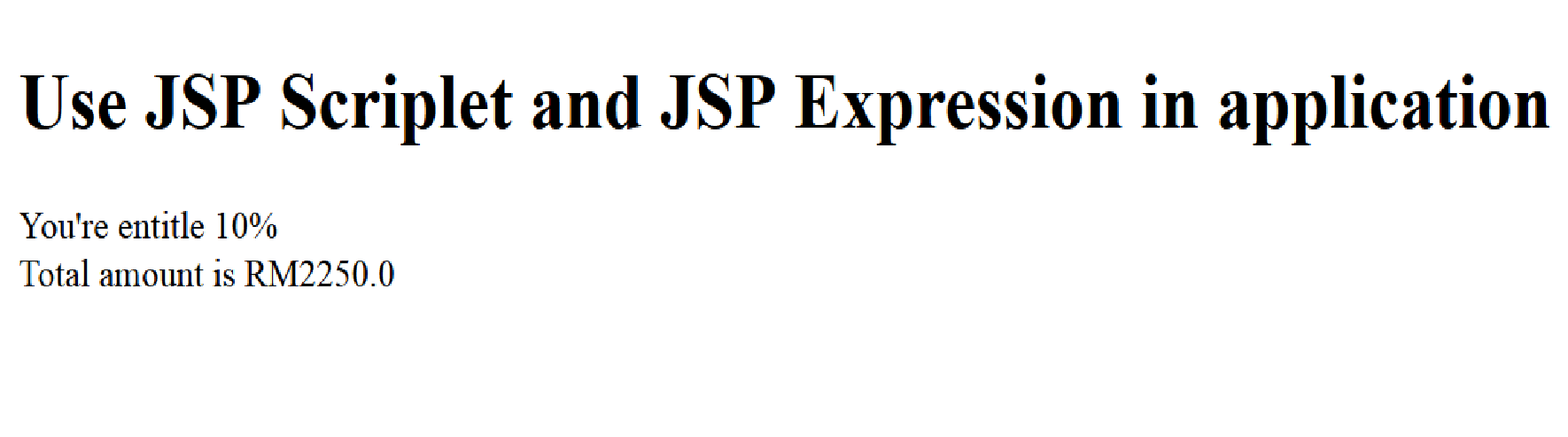


9. Compile *customer.html* and *processCustomer.jsp* file.

10. Run *customer.html.*

1. Enter information to the interface.

1. Output will appear in web browser.



### Reflection

1. What you have learnt from this exercise?

Form Handling: How to create HTML forms and handle submissions using JSP to process user inputs.

1. Explain three (3) type of JSP scripting?

JSP Scriptlets:

* Syntax: <% code %>
* Description: Scriptlets are blocks of Java code that are embedded in the JSP and run whenever the JSP is executed. The code within a scriptlet can declare variables, write conditional statements, loop through data, and more. Anything that you can do in a regular Java method, you can do within a scriptlet.

JSP Declarations:

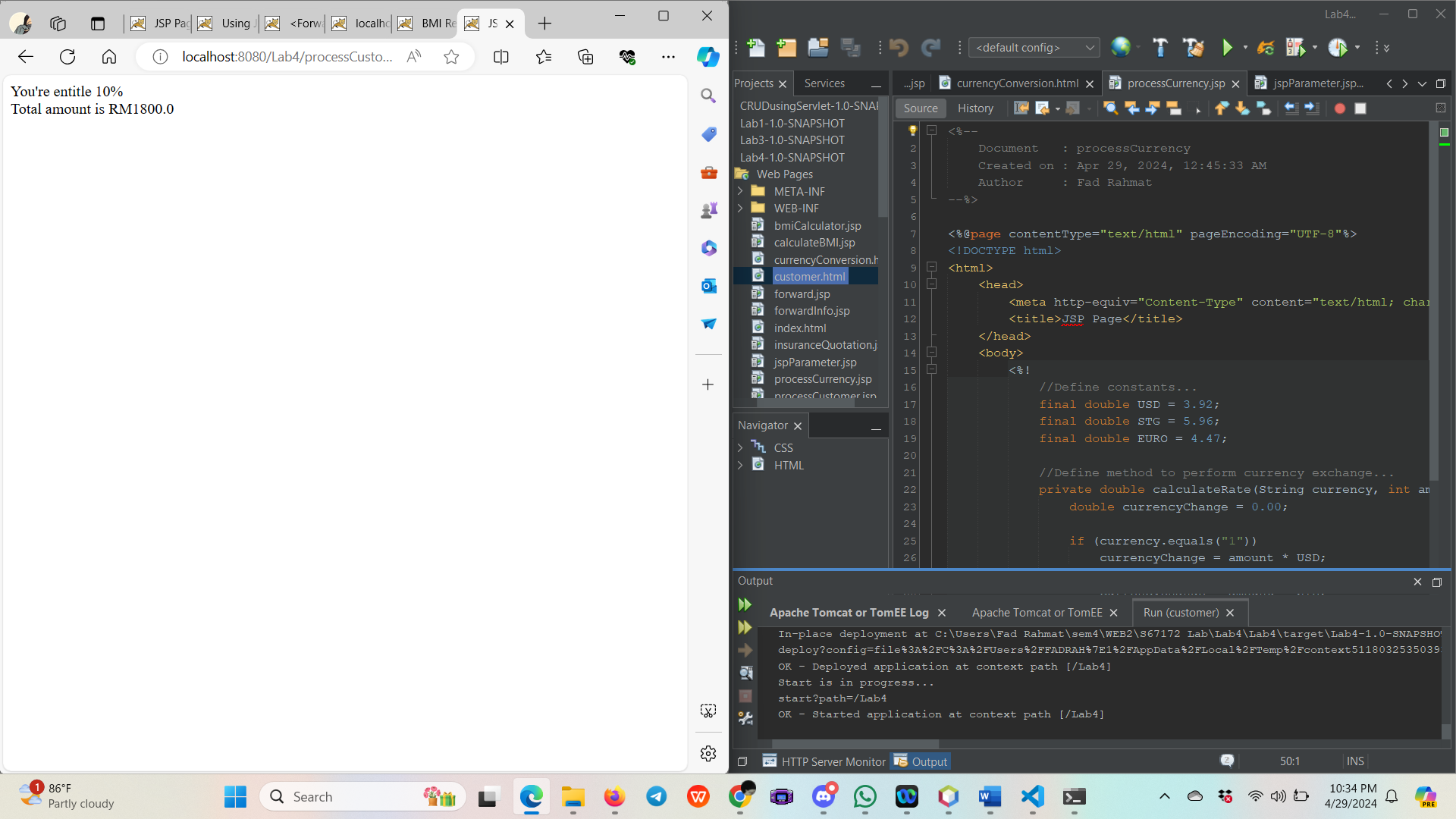
* Syntax: <%! declaration %>
* Description: Declarations are used to declare methods, variables, or classes within the JSP page. Unlike scriptlets, declarations do not produce output directly; they define variables and methods that the other types of scripting elements can use.

JSP Expressions:

* Syntax: <%= expression %>
* Description: Expressions automatically output their results to the client. They are typically used to insert values directly into the output stream of the JSP's response. The expression is evaluated, converted to a string, and inserted in the place where the JSP expression appears in the HTML.

Output:





# Task 2: Using JSP (Scripting, Declaration and Expression)

### Estimated time : 40 minutes

**O**

**bj**

**e**

**ct**

**i**

**ve**

**:**

Use

J

SP

D

ec

l

a

r

a

t

i

o

n

t

a

g,

J

SP

S

c

r

i

p

t

l

et

a

n

d JSP

E

xpr

e

ss

io

n

i

n

a

pp

l

i

c

a

t

i

o

n

.

**P**

**r**

**o**

**b**

**l**

**e**

**m**

**D**

**e**

**s**

**c**

**r**

**i**

**p**

**t**

**i**

**o**

**n**

**:**

Create currency conversion page to Malaysia Ringgit

into

US Dollar, Euro or Pound

Sterling.

1

USD = RM

3.92

1

Pound Sterling = RM

5.96

1

Euro = RM

4.47

;

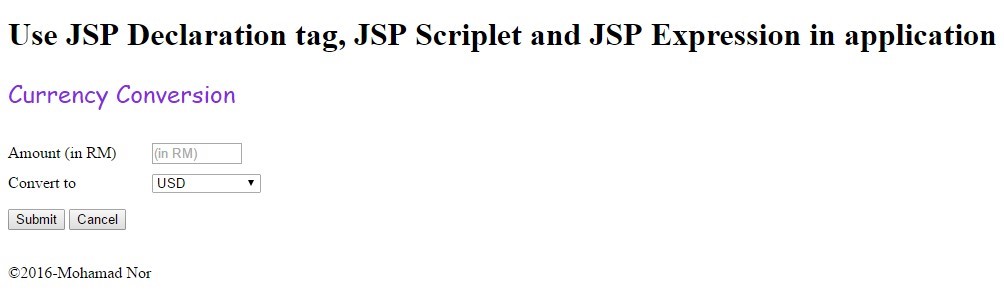
1. Choose Project *Lab3*.

2. Create a new HTML’s file.



1. Type file name as *currencyConversion.*

1. Prepare the following Graphical User Interface (GUI).

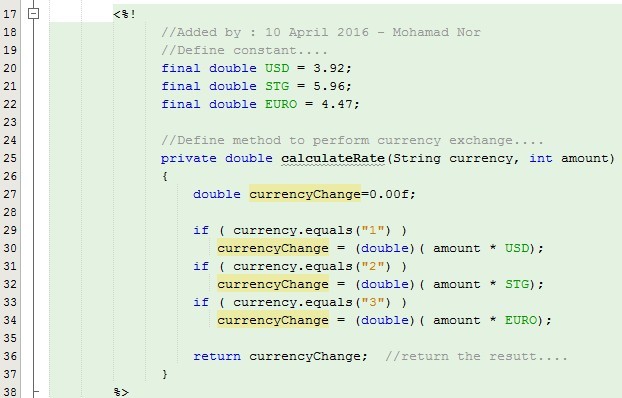


1. You must ensure the amount must be written as number.

1. The value for USD is “1”, Pound Sterling is “2” and Euro is “3”

1. Create a new file name known as *processCurrency.jsp*.

1. Define related variables, currency rate as a constant and method *calculateRate(String code, int amount)* in JSP declaration tag as below.



1. In your JSP scriptlet, retrieve the value for *Amount* and *Convert to* and assign to respective variables*.*

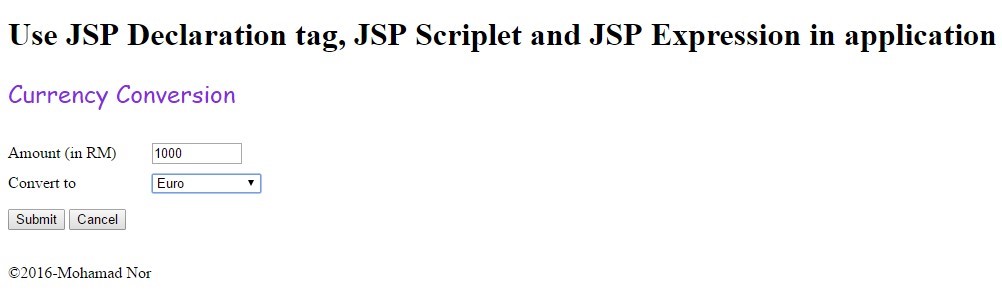
1. Call method *calculateRate(String code, int amount)* to perform currency conversion.

1. Finally, display the result using JSP Expression tag.

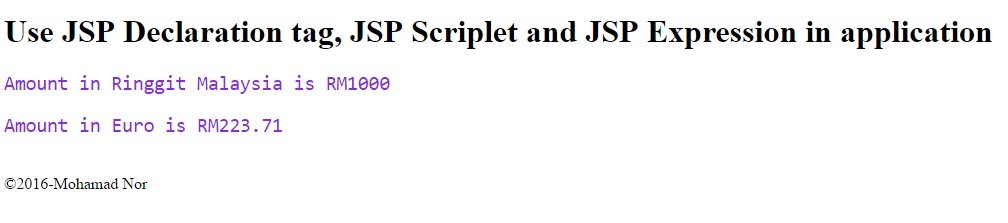
12. Compile *currencyConversion.html* and *processCurrency.jsp* file.

1. Run *currencyConversion.html.*

1. Enter the following information



1. Output will appear in web browser (*Note: Amount must be in 2 decimal places*).



**Reflection**

1. What have you learn from this exercise?

**Form Design and Processing**: The example highlighted how to design HTML forms and capture user input in JSP. It involves setting up forms in HTML with different input types and then using JSP to process the submitted form data.

# output:

# ask 3:

# Using JSP Standard Action (Include and Param)

**O**

**bj**

**e**

**ct**

**i**

**ve**

**:**

Usi

n

g

*<*

*jsp*

*:*

*i*

*n*

*cl*

*u*

*d*

*e>*

a

n

d

*<*

*jsp*

*:*

*p*

*a*

*r*

*a*

*m*

*>*

t

o d

i

s

p

l

a

y

in

f

o

r

m

a

t

io

n

o

n

JSP

p

a

ge

**P**

**r**

**o**

**b**

**l**

**e**

**m**

**D**

**e**

**s**

**c**

**r**

**i**

**p**

**t**

**i**

**o**

**n**

**:**

Di

s

p

l

a

y

t

h

e

c

o

u

r

s

e

i

n

f

o

rm

at

i

o

n

.

**Es**

**t**

**ima**

**t**

**e**

**d t**

**i**

**me**

**:**

2

0

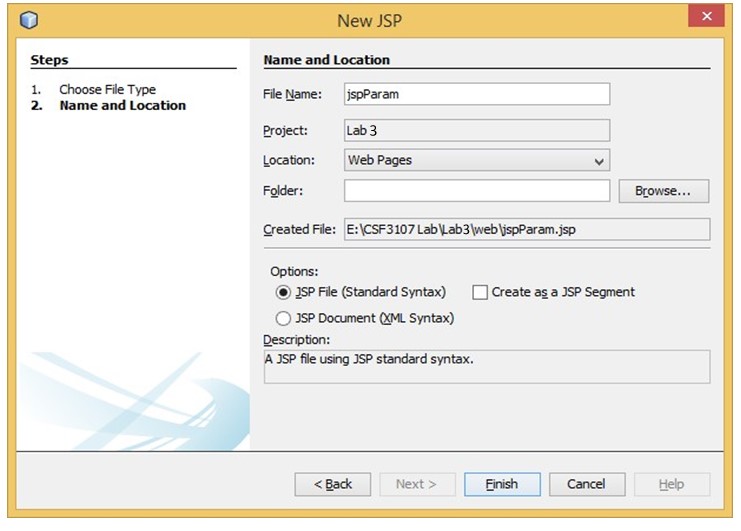
m

inute

s

1. Go to Project *Lab3*.

1. Create a new JSP’s file known as *jspParameter*.



1. Prepare the following HTML’s syntax.



4

.

Add J

a

v

a

s

c

r

i

p

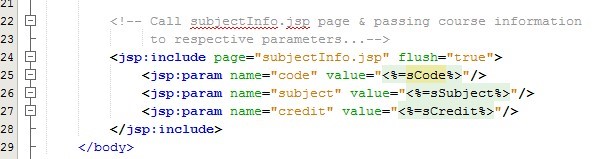
t

l

et

.

1. Add JSP Standard Action *<jsp:include>* to call *subjectInfo.jsp’s* page and *<jsp:parameter>* to store the subject’s information .



1. Save *jspParameter.jsp*’s file.
2. Create another JSP’s file known as *subjectInfo*.





8

.

Wr

it

e

t

h

e

fol

l

o

w

in

g

H

TM

L

’s

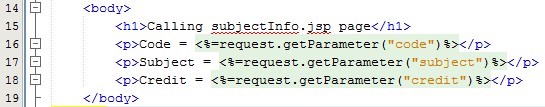
sy

nta

x

.

9. Add three (3) paragraphs and use JSP expression to retrieve and assign value to these paragraphs.



1. Save all files.

1. Compile and run *jspParameter.jsp*’s file.

1. Output will appear in web browser.

### Reflection

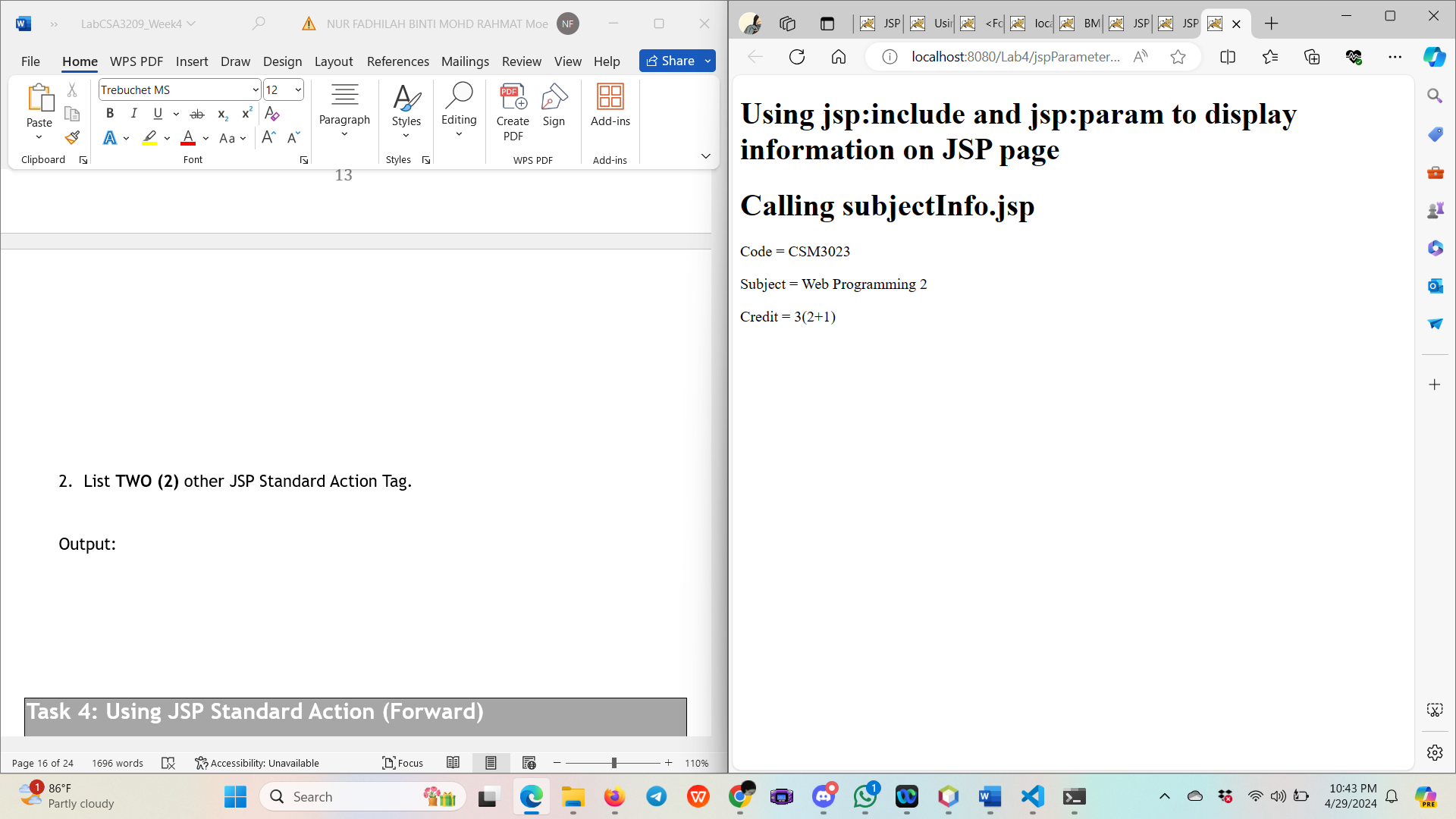
1. What you have learnt from this exercise?

**Modularization**: The use of jsp:include and jsp:param allows for modularization of code, enabling the reuse of common components across multiple pages. This promotes code maintainability and reduces redundancy.

1. List **TWO (2)** other JSP Standard Action Tag.

* <jsp:forward>
* jsp:useBean>

Output:



# Task 4: Using JSP Standard Action (Forward)

**O**

**bj**

**e**

**ct**

**i**

**ve**

**:**

Usi

n

g

*<*

*jsp*

*:*

*forward*

*>*

t

o d

i

s

p

l

a

y

user

in

f

o

r

m

a

t

io

n

and

object

o

n

JSP

p

a

ge

**P**

**r**

**o**

**b**

**l**

**e**

**m**

**:**

Di

s

p

l

a

y

user

information

.

**Es**

**t**

**ima**

**t**

**e**

**d t**

**i**

**me**

**:**

2

0

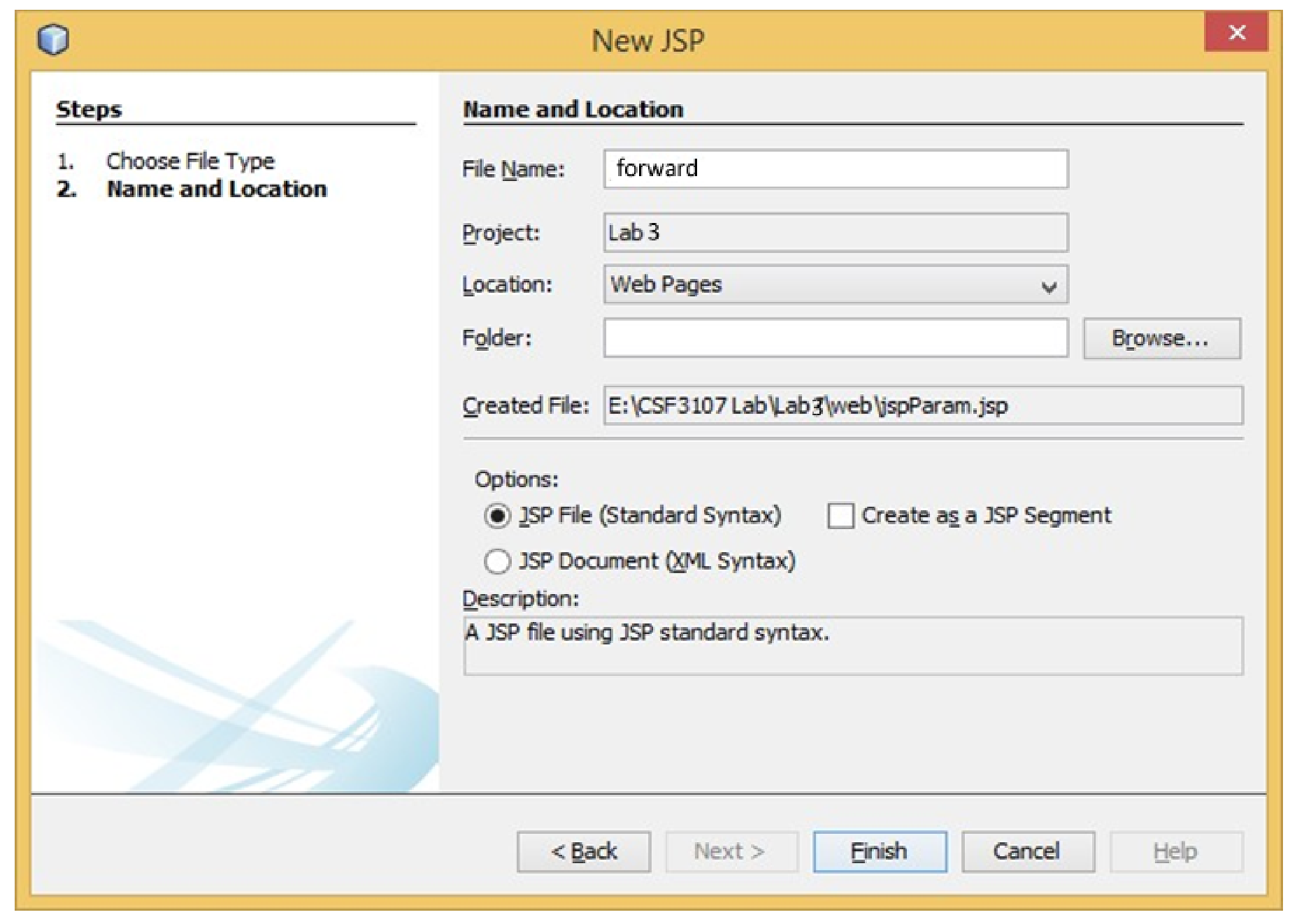
m

inute

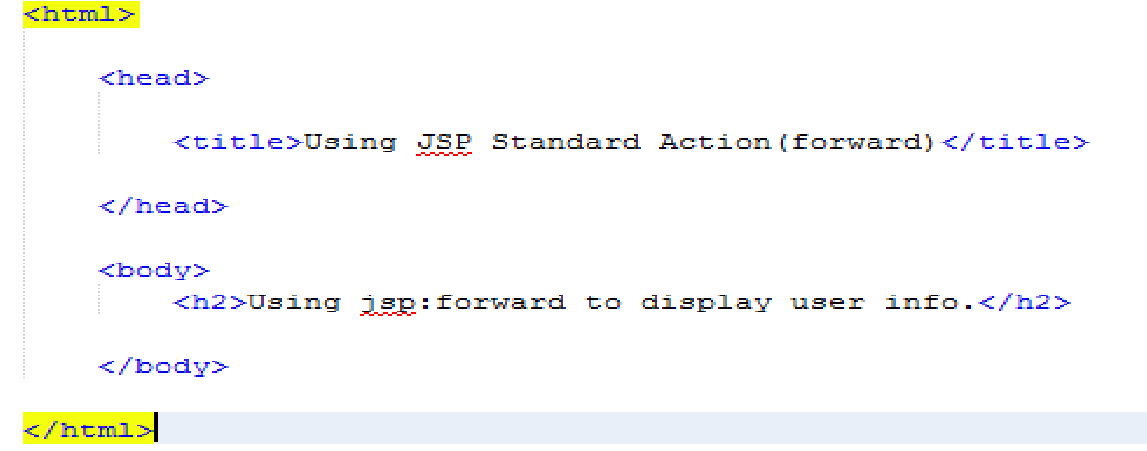
s

1. Go to Project *Lab3*.

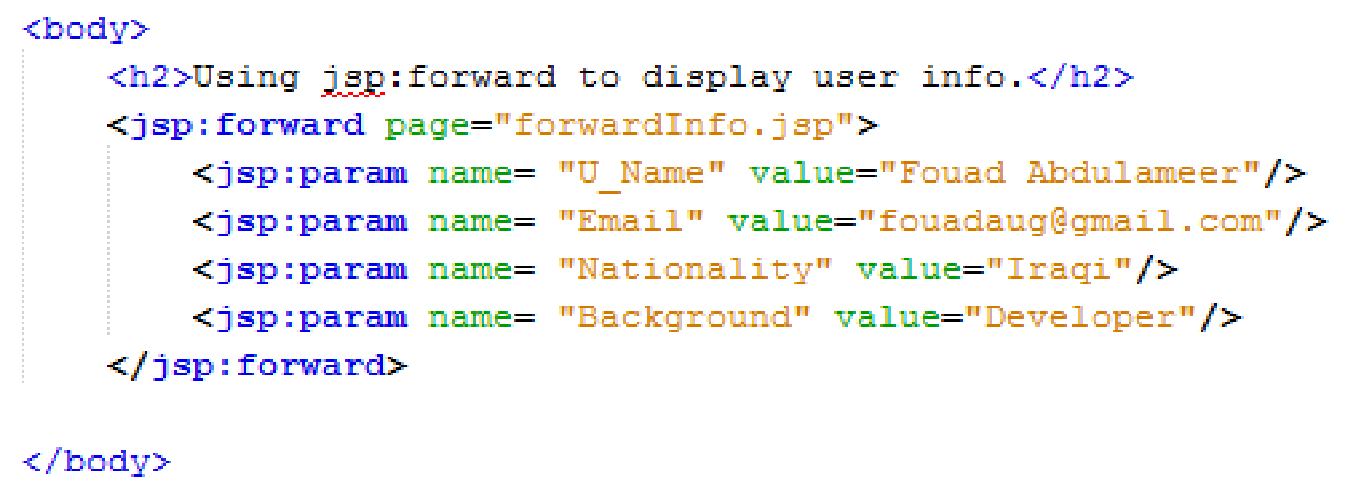
1. Create a new JSP’s file known as *forward*.



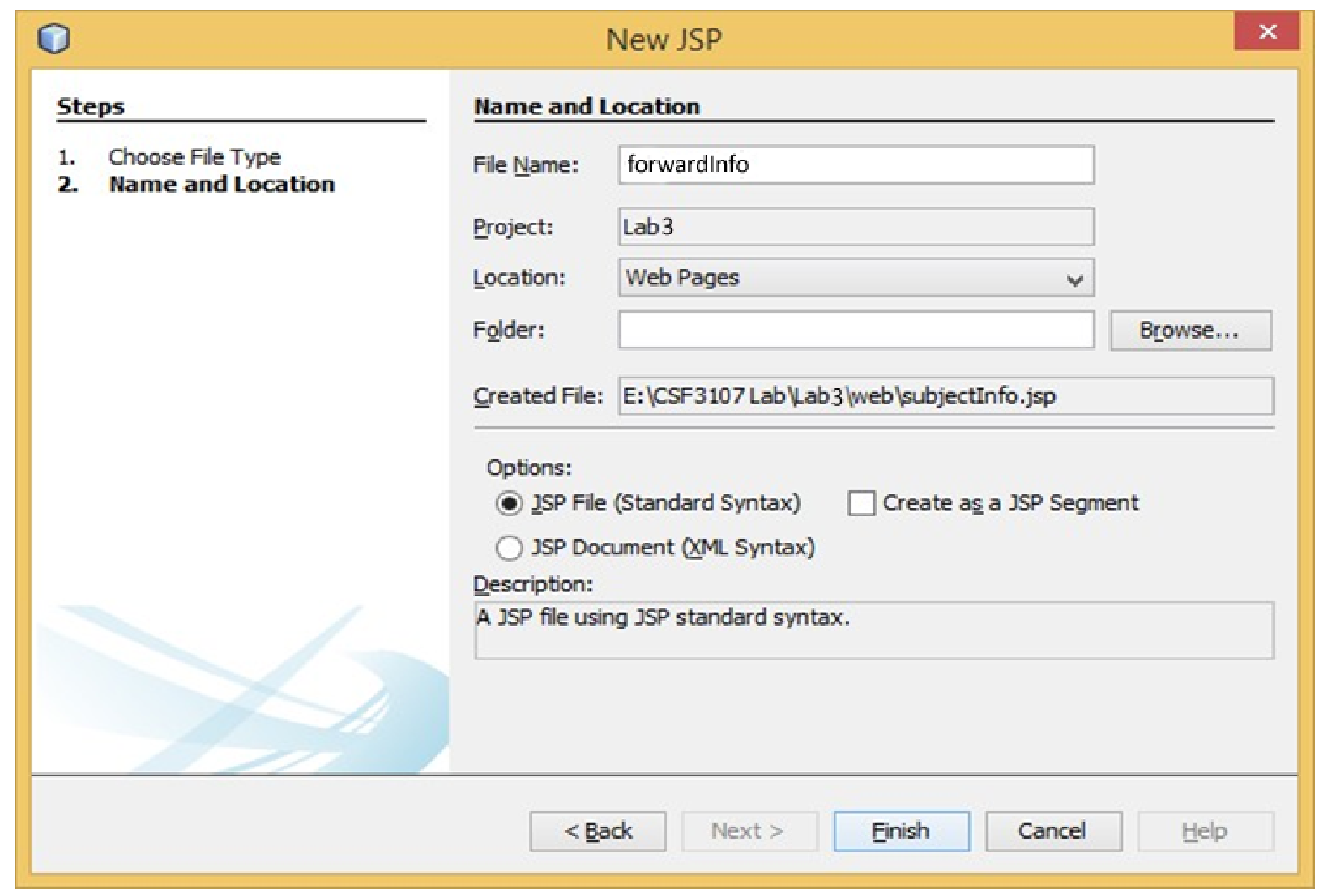
1. Prepare the following HTML’s syntax.



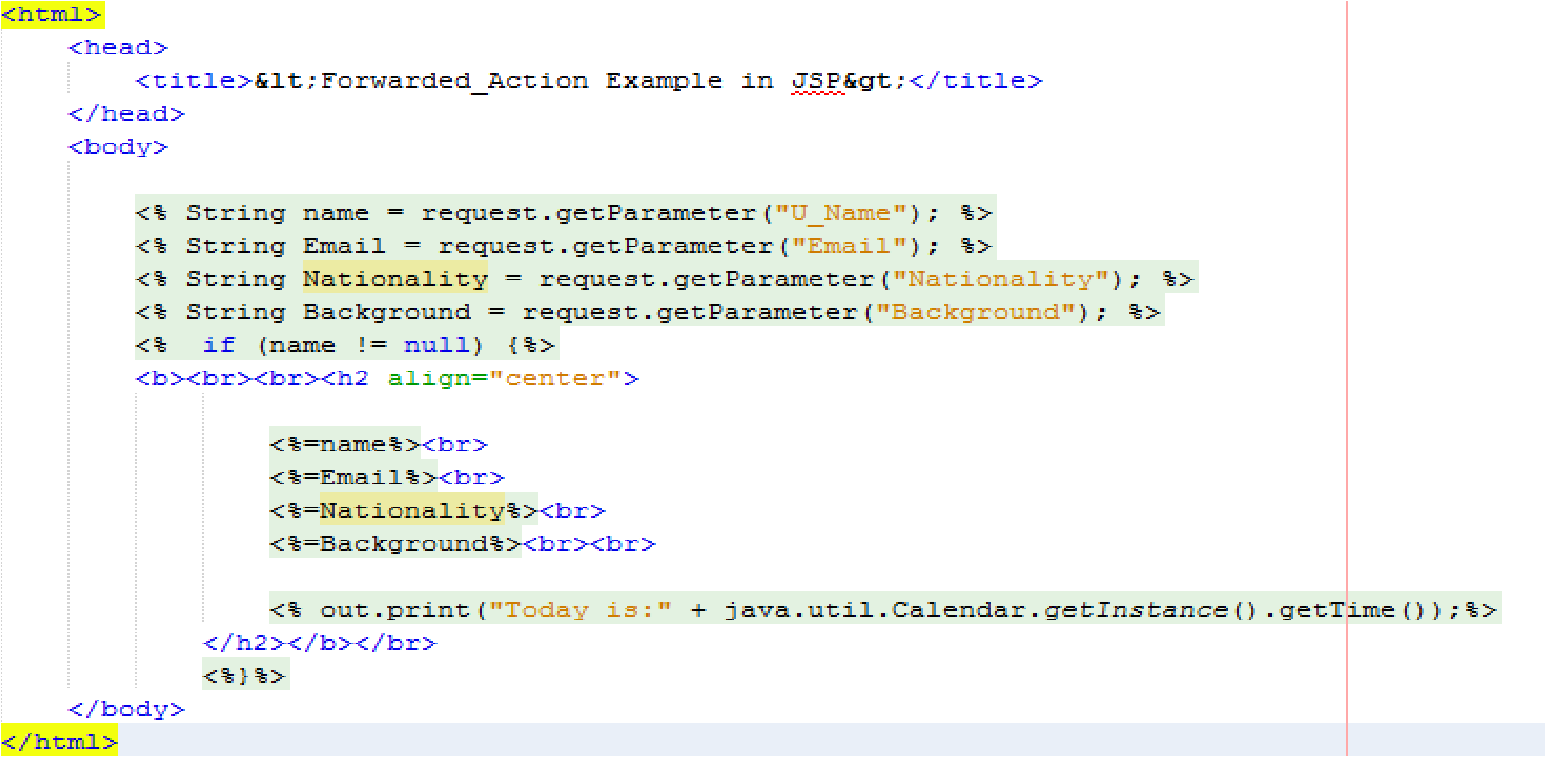
1. Add JSP Standard Action *<jsp:forward>* to call *forwardInfo.jsp’s* page and *<jsp:parameter>* to store the user’s information.



1. Save *forward.jsp*’s file.
2. Create another JSP’s file known as *forwardInfo*.



1. Write the following code.



8

.

S

a

ve

a

l

l

f

i

l

e

s

.

1. Compile and run *forward.jsp*’s file.

1. Output will appear in web browser.

### Reflection

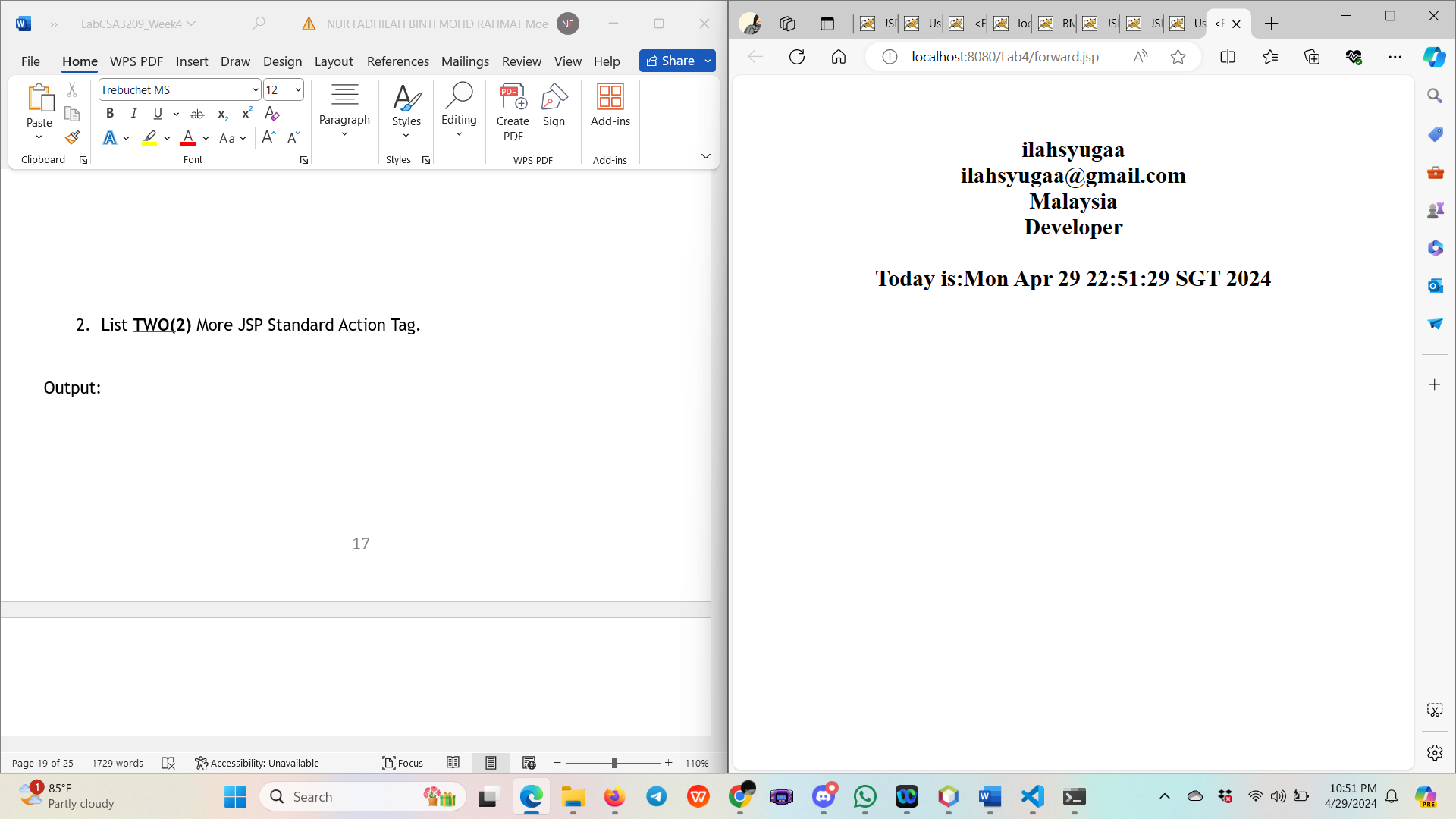
1. What you have learnt from this exercise?

**Forwarding Requests**: I learned how to use the <jsp:forward> standard action to forward HTTP requests from one JSP page to another within the same server context. This allows for better control of request flow and enables modularization of code.

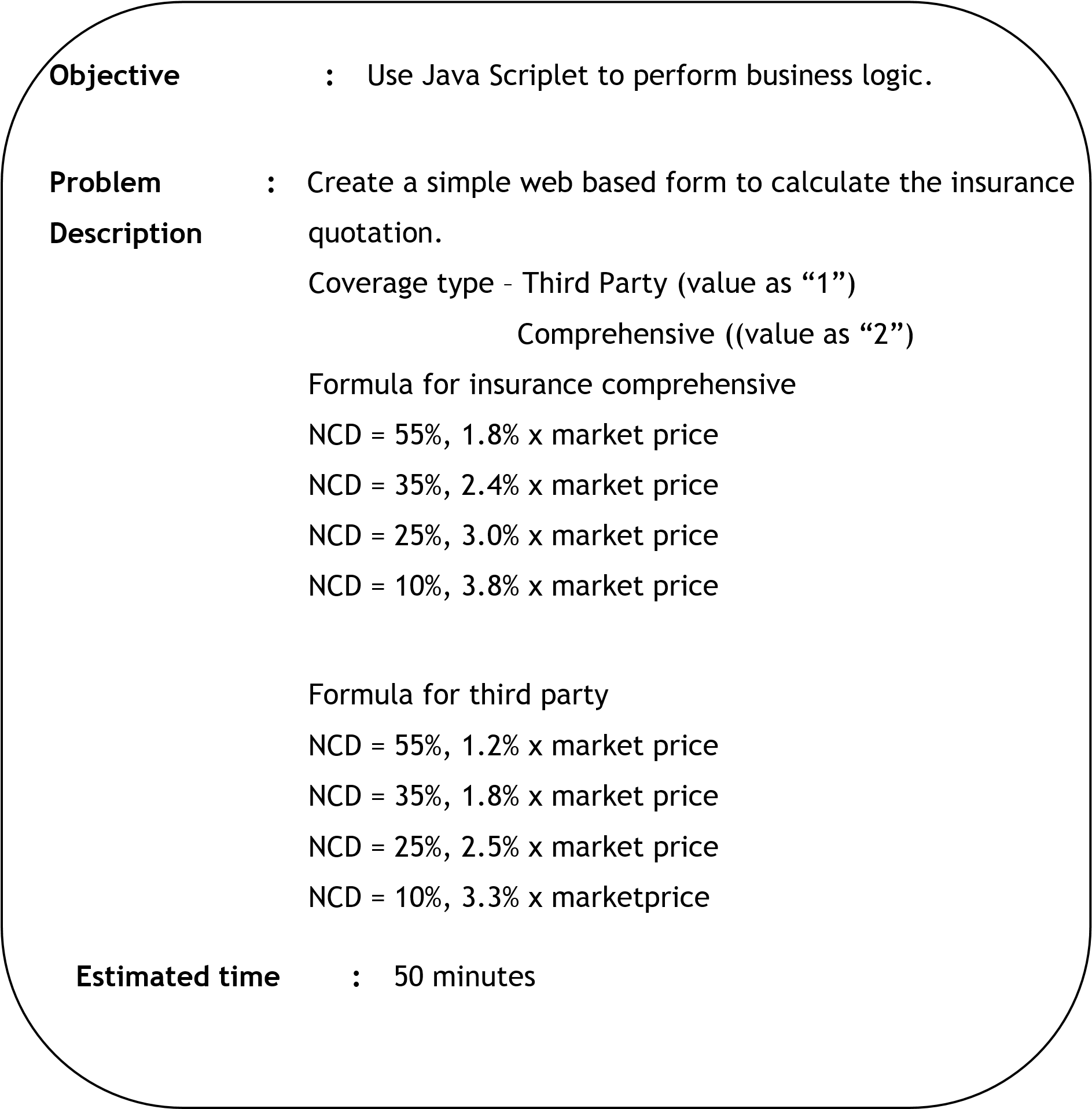
1. List **TWO(2)** More JSP Standard Action Tag.

* <jsp:useBean>
* <jsp:setProperty>

Output:



# Task 5: Use Java Scriptlet To Construct Business Logic



1. Go to project *Lab3*.

1. Create a new JSP’s file as *insuranceQuotation*.

1. Prepare the following Graphical User Interface (GUI).



1. You need to ensure all front-end validation take place.

1. Creating another JSP’s file known as *processInsuranceQuo.jsp*.

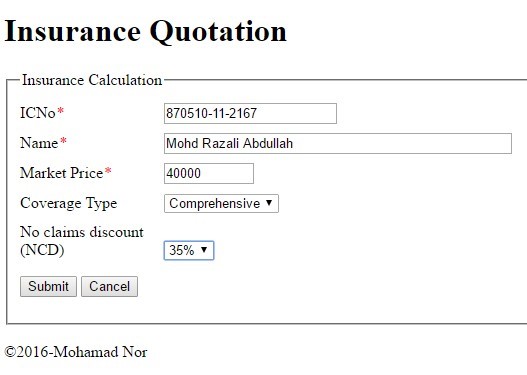
1. Use Java Scriplet to perform the business logic for the application in page *processInsuranceQuo.jsp*.

1. Final insurance amount must be added with 6% GST.

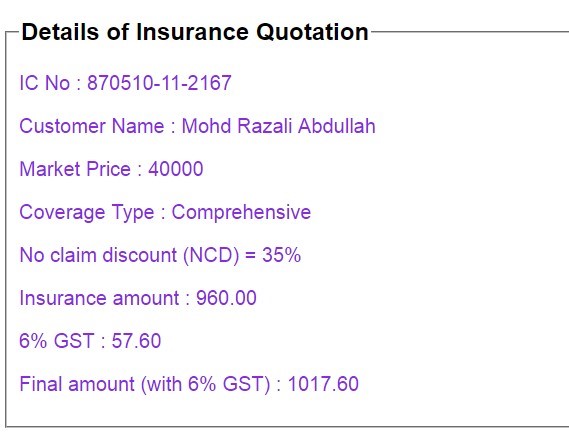
1. Save your file.

1. Right click *insuranceQuotation.jsp* and compile the program.

1. Test your application by key-in the following information.



1. You should get the following output.



### Reflection

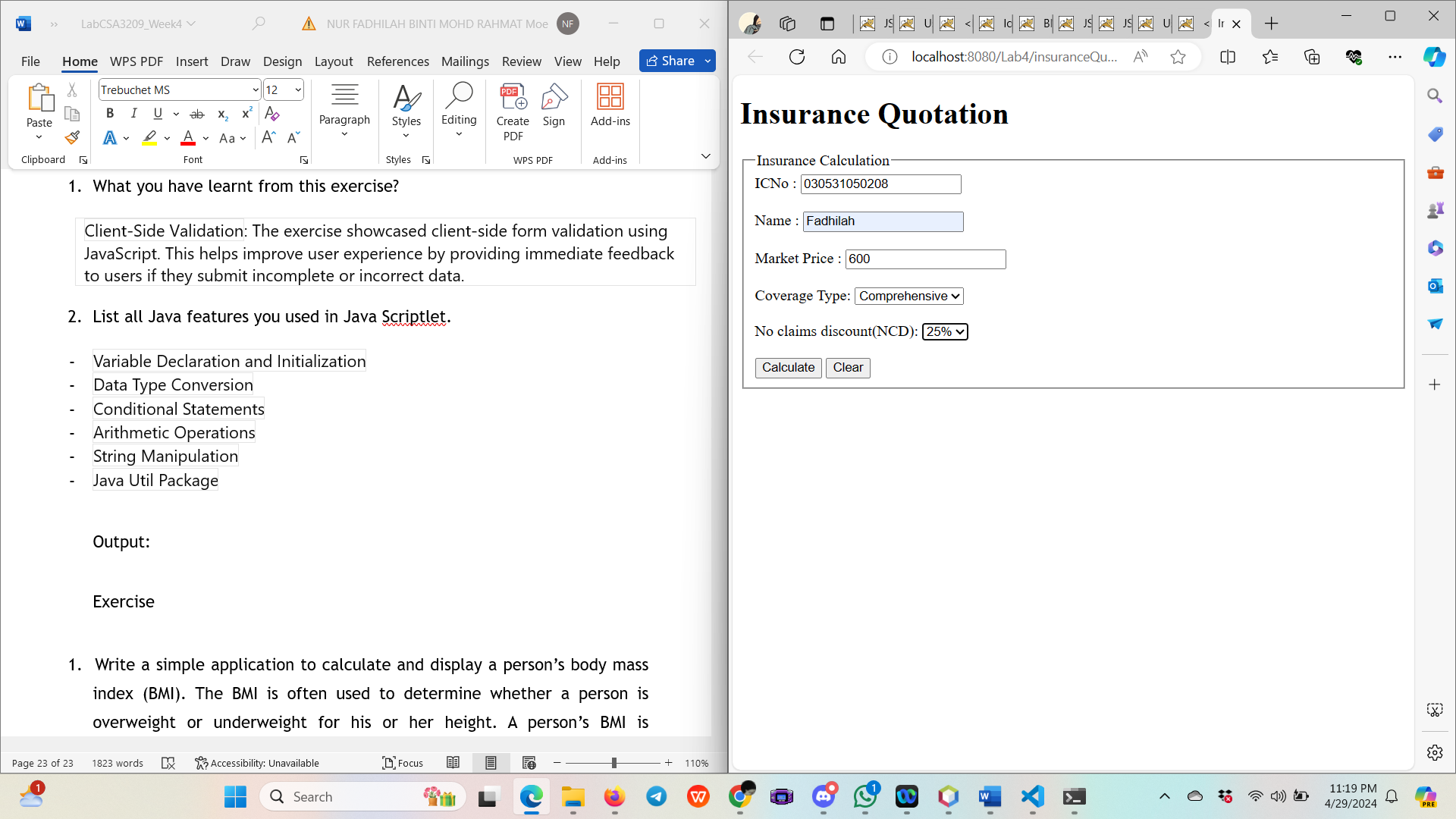
1. What you have learnt from this exercise?

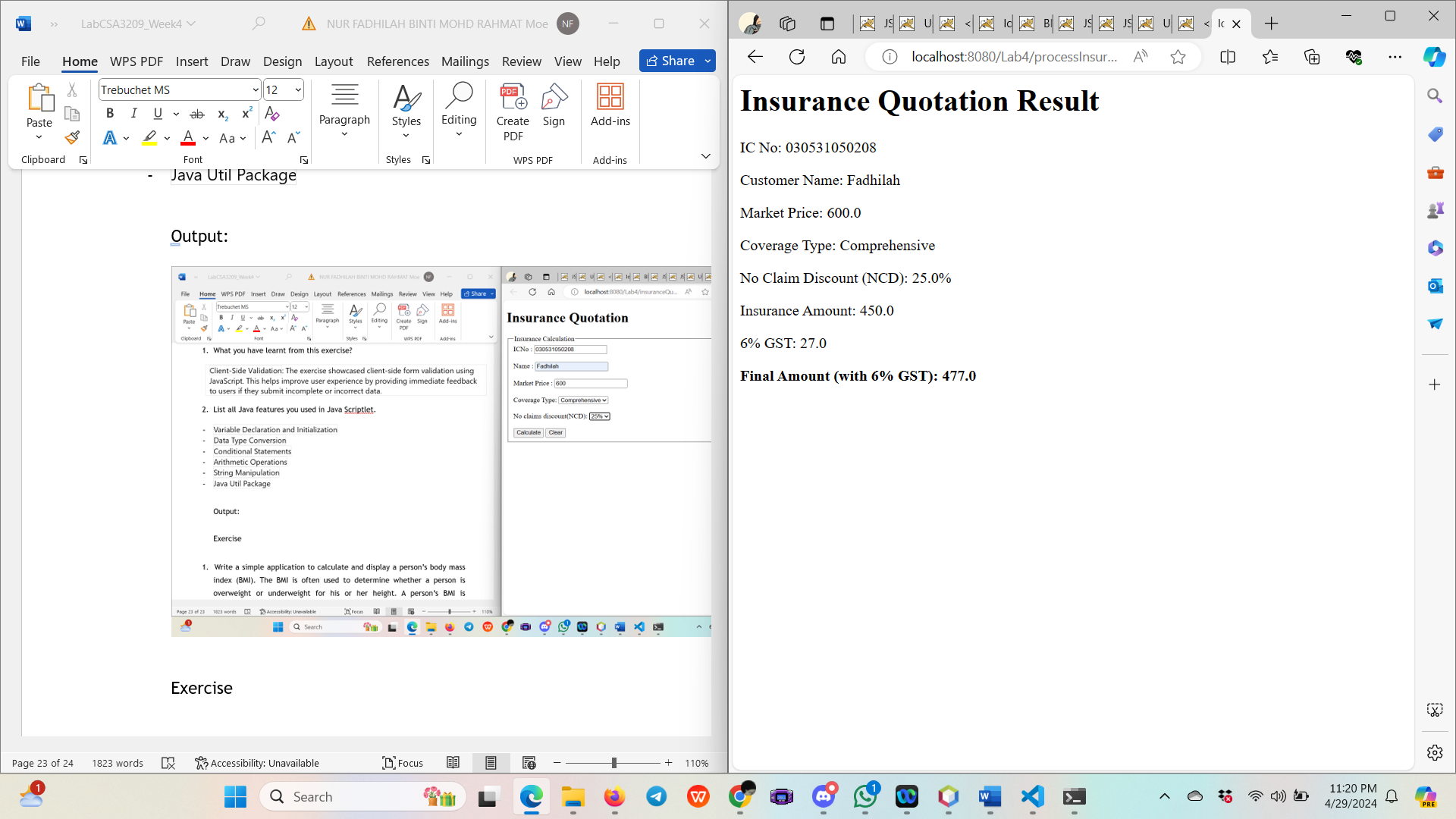
**Client-Side Validation**: The exercise showcased client-side form validation using JavaScript. This helps improve user experience by providing immediate feedback to users if they submit incomplete or incorrect data.

1. List all Java features you used in Java Scriptlet.

* **Variable Declaration and Initialization**
* **Data Type Conversion**
* **Conditional Statements**
* **Arithmetic Operations**
* **String Manipulation**
* **Java Util Package**

Output:





Exercise

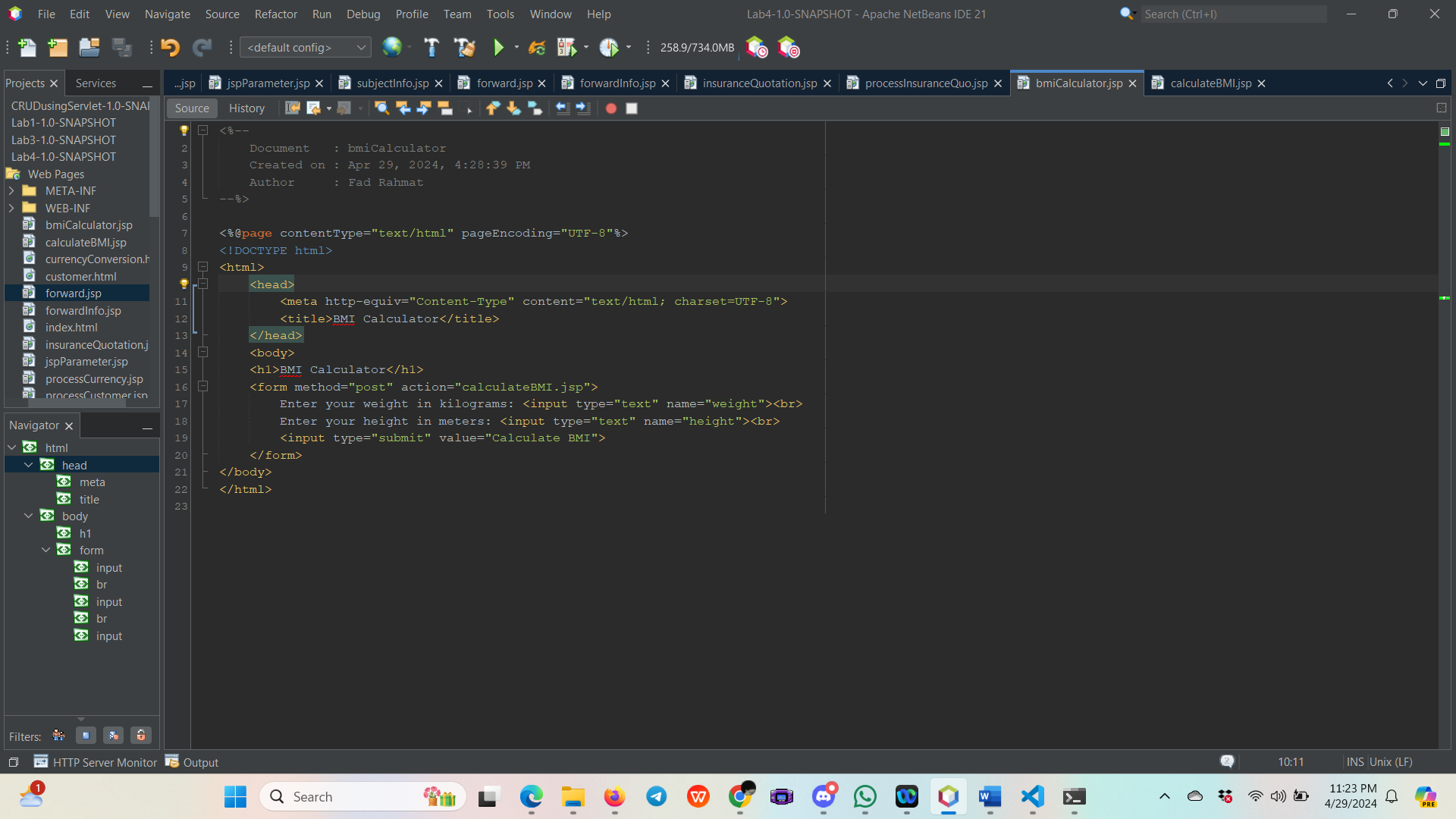
1. Write a simple application to calculate and display a person’s body mass index (BMI). The BMI is often used to determine whether a person is overweight or underweight for his or her height. A person’s BMI is calculated with the following formula:

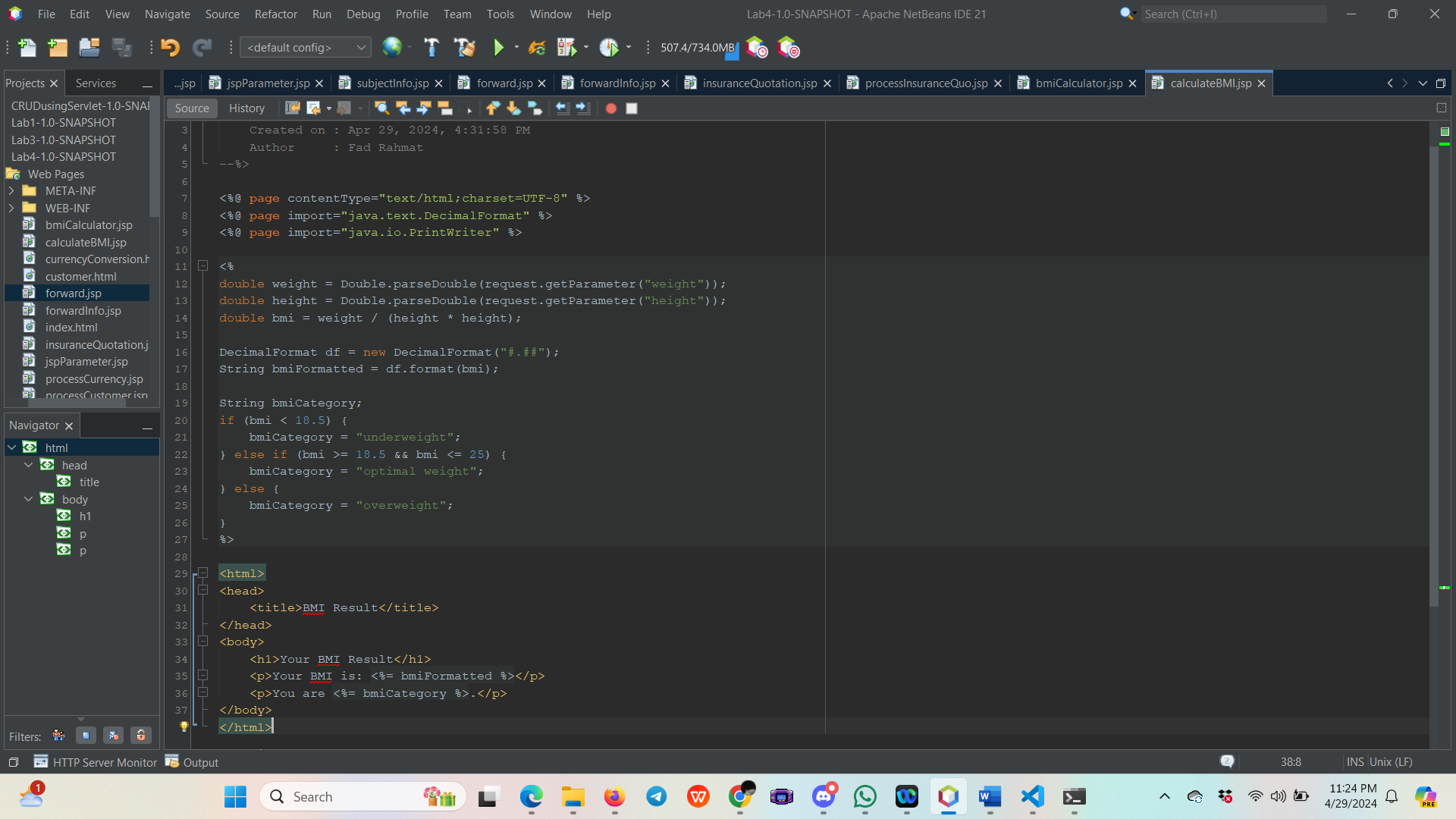
**BMI = weight /height2**

where weight is measured in kilogram and height is measured in meter. User should enter his or her weight and height and then display the user’s BMI.

The program should also display a message indicating whether the person has optimal weight, is underweight, or is overweight. A person’s weight is considered to be optimal if his or her BMI is between 18.5 and 25. If the BMI is less than 18.5, the person is considered to be underweight. If the BMI value is greater than 25, the person is considered to be overweight.

Code:





Output:

