

FIAP GRADUAÇÃO

SISTEMAS DE INFORMAÇÃO

MICROSERVICE AND WEB ENGINEERING

Prof^a. Aparecida Castello Rosa
profaparecida.rosa@fiap.com.br

Material de apoio:
PROF. PEDRO IVO CORREIA
PROF. LUCAS FURLANETO

Agenda

- Explicação de Servlet e seu ciclo de vida
- Java Server Pages - JSP
- Exercícios JSP e Servlet

Objetivos

- Entender o funcionamento base do ciclo de vida dos *requests* e *responses* em um projeto web utilizando *Servlet* e JSP.

Servlet

■ Para saber mais

Java Servlet Technology Overview:

<https://www.oracle.com/java/technologies/servlet-technology.html>

Alura

<https://www.alura.com.br/apostila-java-web/servlets>

- A API Java Servlet (do pacote javax.servlet) proporciona ao desenvolvedor a possibilidade de adicionar conteúdo dinâmico em um servidor Web usando a plataforma Java.
- Esta tecnologia disponibiliza ao programador da linguagem Java uma interface para o servidor Web (ou servidor de aplicação), através de uma API. As aplicações baseadas no Servlet geram conteúdo dinâmico (normalmente **HTML**) e interagem com os clientes, utilizando o modelo *request-response* (requisição-resposta).
- Os *servlets* normalmente utilizam o protocolo **HTTP**, apesar de não serem restritos a ele. Um **Servlet** necessita de um **Container Web** para ser executado.

A maior parte do conteúdo da internet é baseado em conteúdo dinâmico.

Utilizamos **Servlet** para criar conteúdo dinâmico.

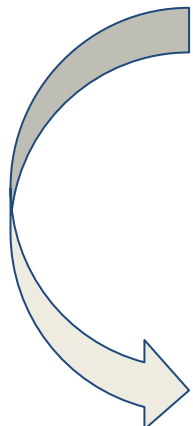
Servlet é uma classe Java que processa informações e produz conteúdo.

A resposta será sempre um conteúdo que o navegador pode interpretar, como HTML, por exemplo.

Servlet - Ciclo de Vida e Métodos

- **init(ServletConfig config)**
- **service(HttpServletRequest, HttpServletResponse)**
 - doGet(HttpServletRequest, HttpServletResponse)
 - doPost(HttpServletRequest, HttpServletResponse)
 - doPut(HttpServletRequest, HttpServletResponse)
 - doDelete(HttpServletRequest, HttpServletResponse)
 - demais
- **void destroy()**

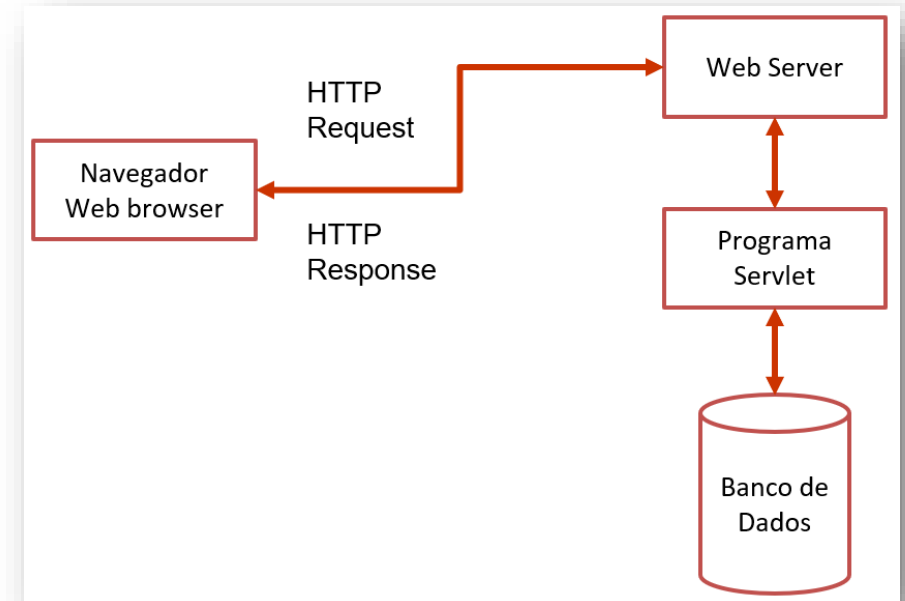
Se o método service for declarado, os demais métodos não serão executados.



Servlet

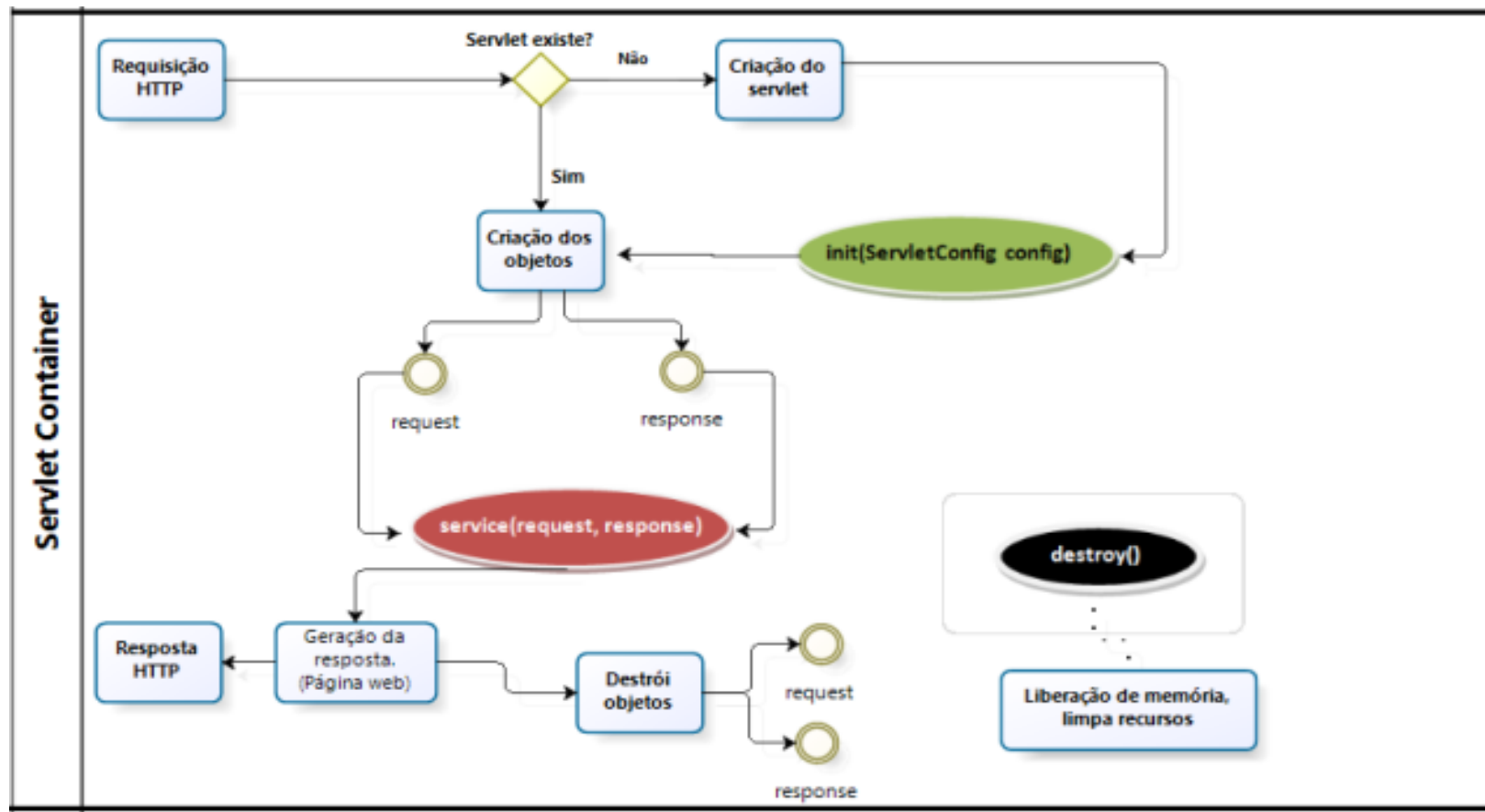
A execução de **Servlets** envolve basicamente seis etapas básicas:

1. Os clientes enviam a solicitação ao servidor Web.
2. O servidor Web recebe a solicitação.
3. O servidor Web passa a solicitação para o **Servlet** correspondente.
4. O **Servlet** processa a solicitação e gera a resposta na forma de saída.
5. O **Servlet** envia a resposta de volta ao servidor Web.
6. O servidor Web envia a resposta de volta ao cliente e o navegador do cliente a exibe na tela.



Fonte: <https://www.geeksforgeeks.org/introduction-java-servlets/>

Servlet



Fonte: <https://www.devmedia.com.br/ciclo-da-vida-do-servlet/27919>

Estrutura do Servlet

```
public class Index extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    → public void init(ServletConfig config) throws ServletException {  
        System.out.println("init");  
    }  
  
    public void destroy() {  
        System.out.println("destroy");  
    }  
  
    → protected void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
        System.out.println("doGet");  
        response.getWriter().append("Served at: ").append(request.getContextPath());  
    }  
  
    → protected void doPost(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
        System.out.println("doPost");  
        doGet(request, response);  
    }  
}
```

Configuração do Servlet

- Servlet 2.5 ou menor - web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <display-name>projeto-jsp</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>

  <!-- Red arrow pointing to this tag -->
  <servlet>
    <servlet-name>Index</servlet-name>
    <servlet-class>br.com.fiap.projeto.Index</servlet-class>
  </servlet>

  <!-- Red arrow pointing to this tag -->
  <servlet-mapping>
    <servlet-name>Index</servlet-name>
    <url-pattern>/Index</url-pattern>
  </servlet-mapping>

</web-app>
```

Todos os Servlets criados na versão 2.4 ou inferior necessitam estar declarados no arquivo **web.xml** por meio de duas tags xml: **servlet** e **servlet-mapping**.

Configuração do Servlet

- Servlet 3.0 ou superior – utiliza *annotation*

```
3 // código omitido
4 import java.io.IOException;
5 import javax.servlet.ServletConfig;
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 @WebServlet("/index")
13 public class Index extends HttpServlet {
14     private static final long serialVersionUID = 1L;
15
16     public void init(ServletConfig config) throws ServletException {
17         System.out.println("init");
18     }
19 // código omitido
```

Servlet – Apache Tomcat Versions

Apache Tomcat® is an open-source software implementation of a subset of the Jakarta EE (formally Java EE) technologies. Different versions of Apache Tomcat are available for different versions of the specifications. The mapping between [the specifications](#) and the respective Apache Tomcat versions is:

Servlet Spec	JSP Spec	EL Spec	WebSocket Spec	Authentication (JASPIC) Spec	Apache Tomcat Version	Latest Released Version	Supported Java Versions
6.1	4.0	6.0	TBD	TBD	11.0.x	11.0.0-M16 (alpha)	21 and later
6.0	3.1	5.0	2.1	3.0	10.1.x	10.1.18	11 and later
5.0	3.0	4.0	2.0	2.0	10.0.x (superseded)	10.0.27 (superseded)	8 and later
4.0	2.3	3.0	1.1	1.1	9.0.x	9.0.85	8 and later
3.1	2.3	3.0	1.1	1.1	8.5.x	8.5.98	7 and later
3.1	2.3	3.0	1.1	N/A	8.0.x (superseded)	8.0.53 (superseded)	7 and later
3.0	2.2	2.2	1.1	N/A	7.0.x (archived)	7.0.109 (archived)	6 and later (7 and later for WebSocket)
2.5	2.1	2.1	N/A	N/A	6.0.x (archived)	6.0.53 (archived)	5 and later
2.4	2.0	N/A	N/A	N/A	5.5.x (archived)	5.5.36 (archived)	1.4 and later
2.3	1.2	N/A	N/A	N/A	4.1.x (archived)	4.1.40 (archived)	1.3 and later
2.2	1.1	N/A	N/A	N/A	3.3.x (archived)	3.3.2 (archived)	1.1 and later

Each version of Tomcat is supported for any stable Java release that meets the requirements of the final column in the table above.

Fonte: <https://tomcat.apache.org/whichversion.html>



Java Server Pages - JSP

Java Server Pages - JSP

- **Java Server Pages (JSP)** é uma tecnologia que ajuda os desenvolvedores de software a criarem páginas Web geradas dinamicamente baseadas em **HTML**, **XML** ou outros tipos de documentos.
- Lançada em 1999 pela Sun Microsystems, **JSP** é similar ao PHP, mas usa a **linguagem de programação Java**.



Java Server Pages - JSP

- Foram criadas para contornar algumas limitações ao desenvolvimento no **Servlet**.
- É um documento **HTML** que permite execução de Java através do *Scriptlet*;
- As páginas **JSP** são transformadas em **Servlets**.
- Utilizamos **JSP** para interface e a lógica no **Servlets**.



Java Server Pages - JSP

- A tecnologia JavaServer Pages - **JSP**, possibilita juntar conteúdo gerado dinamicamente com conteúdo estático **HTML**.
- Podemos pensar em **JSP** como páginas **HTML** em que se inseriu código Java, geralmente, com uso de tags `<%` e `%>` que delimitam um *scriptlet*.



- Um ***scriptlet*** pode conter qualquer número de instruções de linguagem JAVA, declarações de variáveis ou métodos ou expressões que sejam válidas na linguagem de *script* de página.

Scriptlet

```
<% code fragment %>
```



JSP - Sintaxe

Comentários em JSP

```
<%-- Comentário JSP --%>
```

Declaration Tag - criar variáveis e métodos

```
<%! campo ou método %>
```

ou XML equivalente

```
<jsp:declaration>  
    campo ou método  
</jsp:declaration>
```

```
<%! private int contador = 0; %>
```

Expression Tag - mostrar algum resultado

```
<%= contador %>
```

Directive Tag - informações ao motor do JSP

```
<%@ include file="pagina.jsp" %>
```

Scriptlet tag - código comum

```
<% for(int i = 1; i < 4; i++){ %>  
    <p>Este número é <%= i %></p>  
<% } %>
```

Action tag - uso com JavaBeans ou redirecionamentos

```
<jsp:useBean id="calc" class="br.com.fiap.util.Calculator" />  
<jsp:forward page="index.jsp"></jsp:forward>
```

Projeto login-jsp

Formulário JSP

Exercício 1

Simular Login

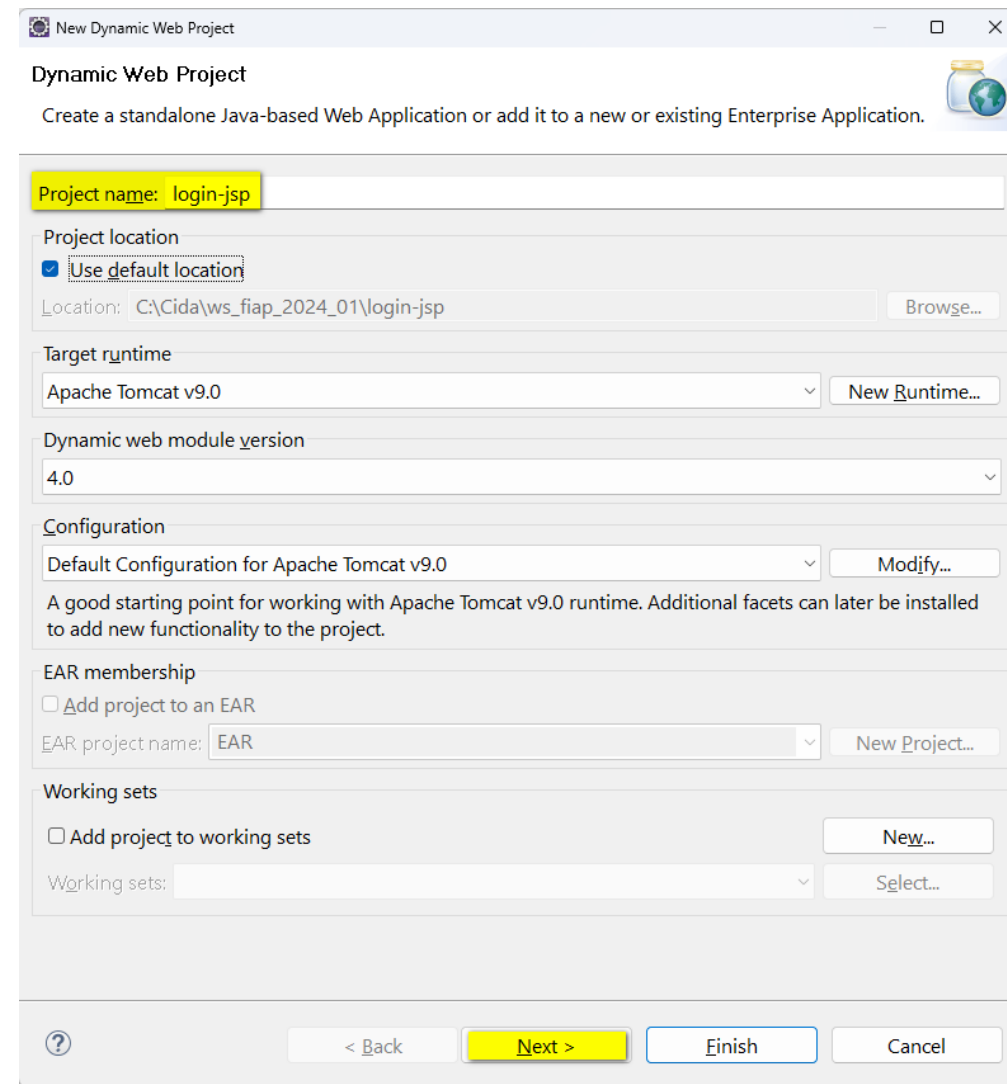
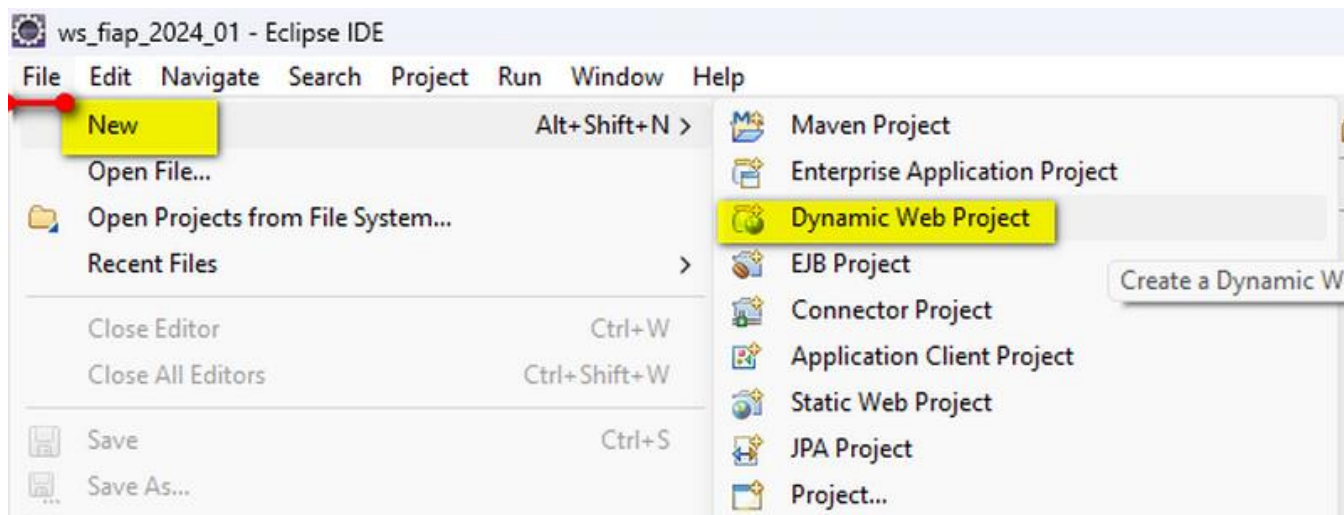
Observação

- Parar o servidor executando o arquivo shutdown.bat
- Local dos arquivos na FIAP
- C:\OpenSource\apache-tomcat-9.0.78\bin\ shutdown.bat

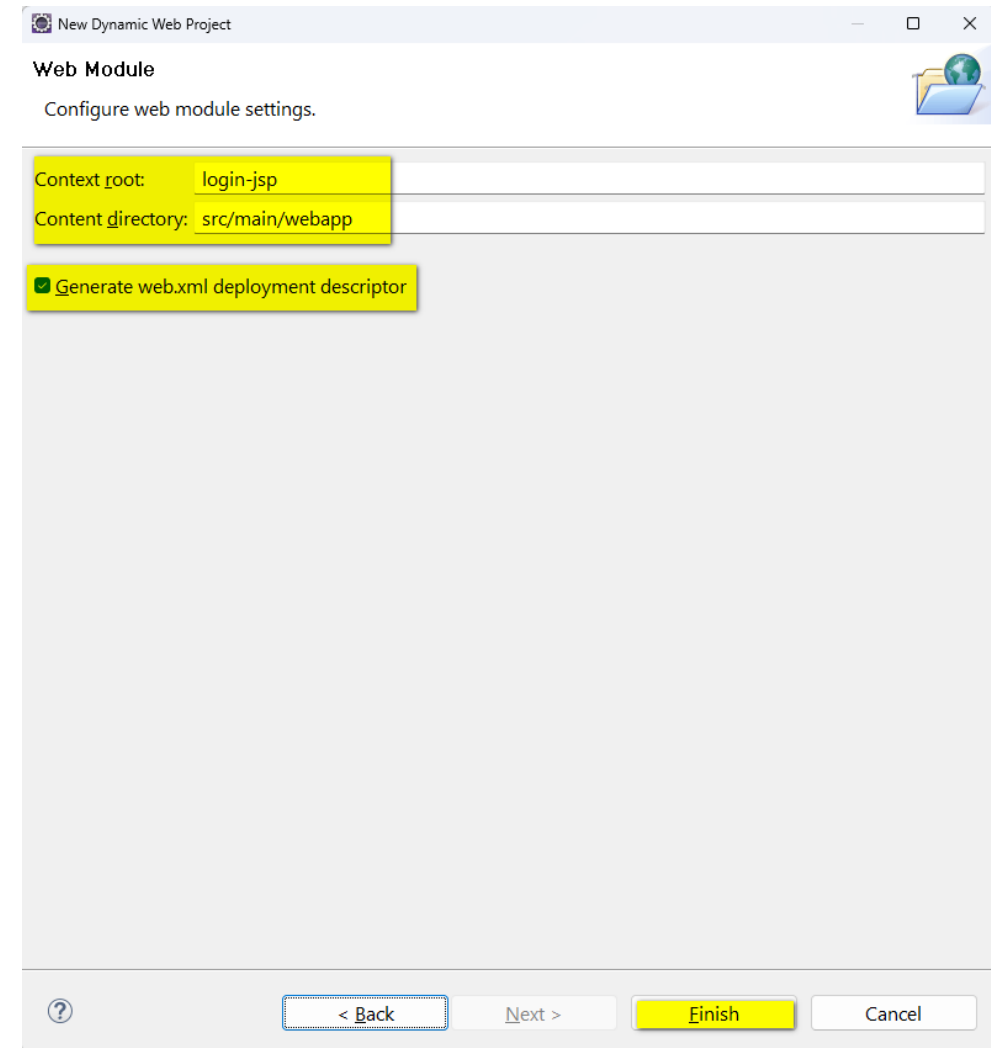
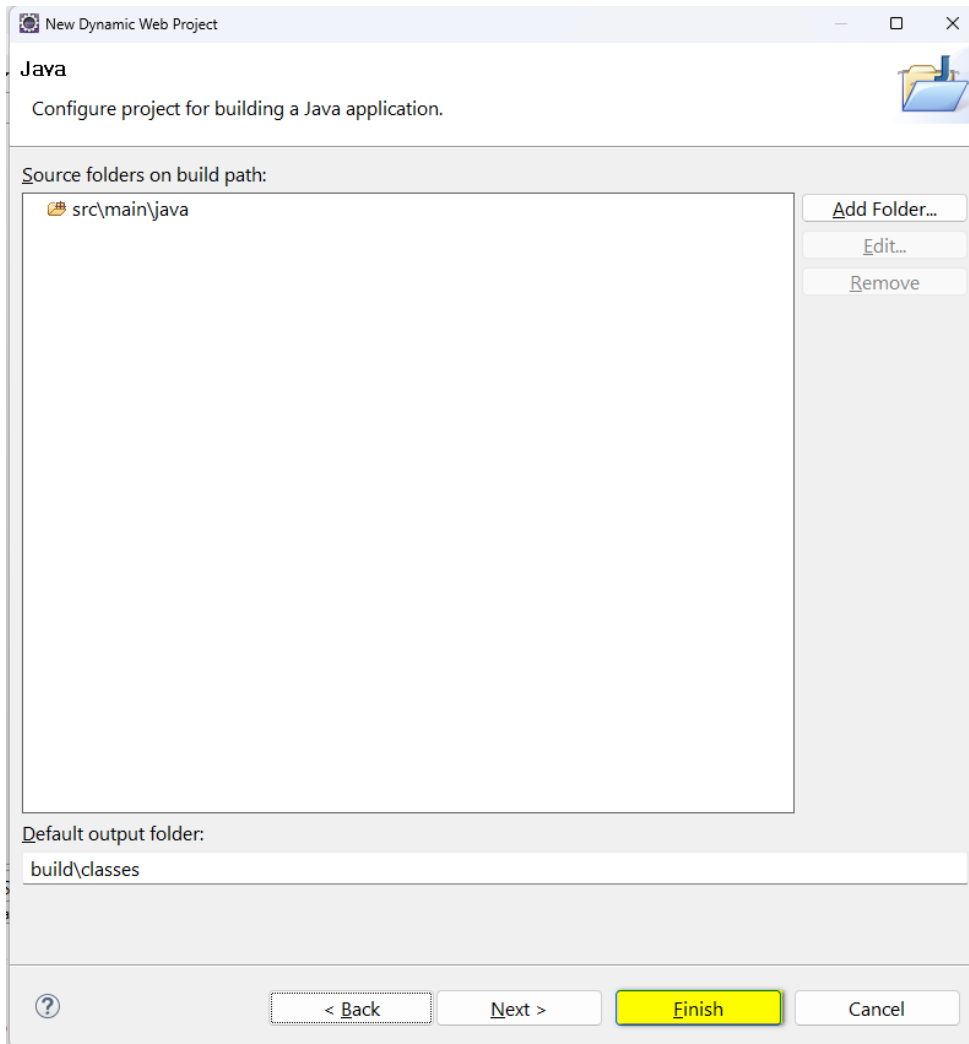
Projeto login-jsp - Passos

1. Criar projeto **login-jsp**.
2. Criar a página **index.jsp** e inserir um formulário com **nome**, **senha** e **botão enviar**.
3. O formulário da página **index.jsp** deverá ter sua propriedade **action** apontada para a página **home.jsp**.
4. Crie um novo **jsp** com o nome **home.jsp** na pasta **webapp** da aplicação.
5. Na página **home.jsp** insira um ***scriptlet*** para validar se a senha digitada é igual a 123456 e imprimir uma mensagem de sucesso ou erro.

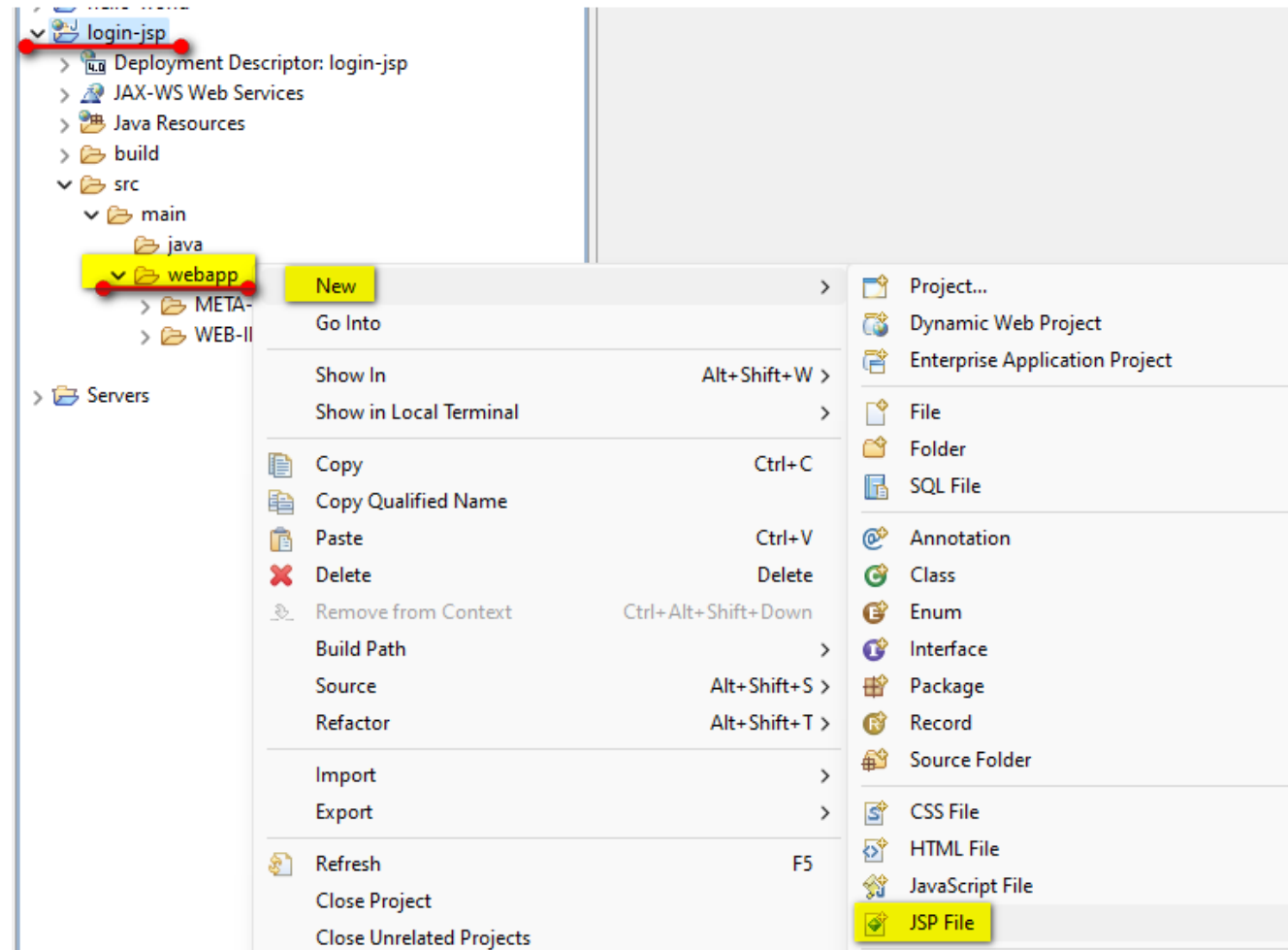
Projeto login-jsp



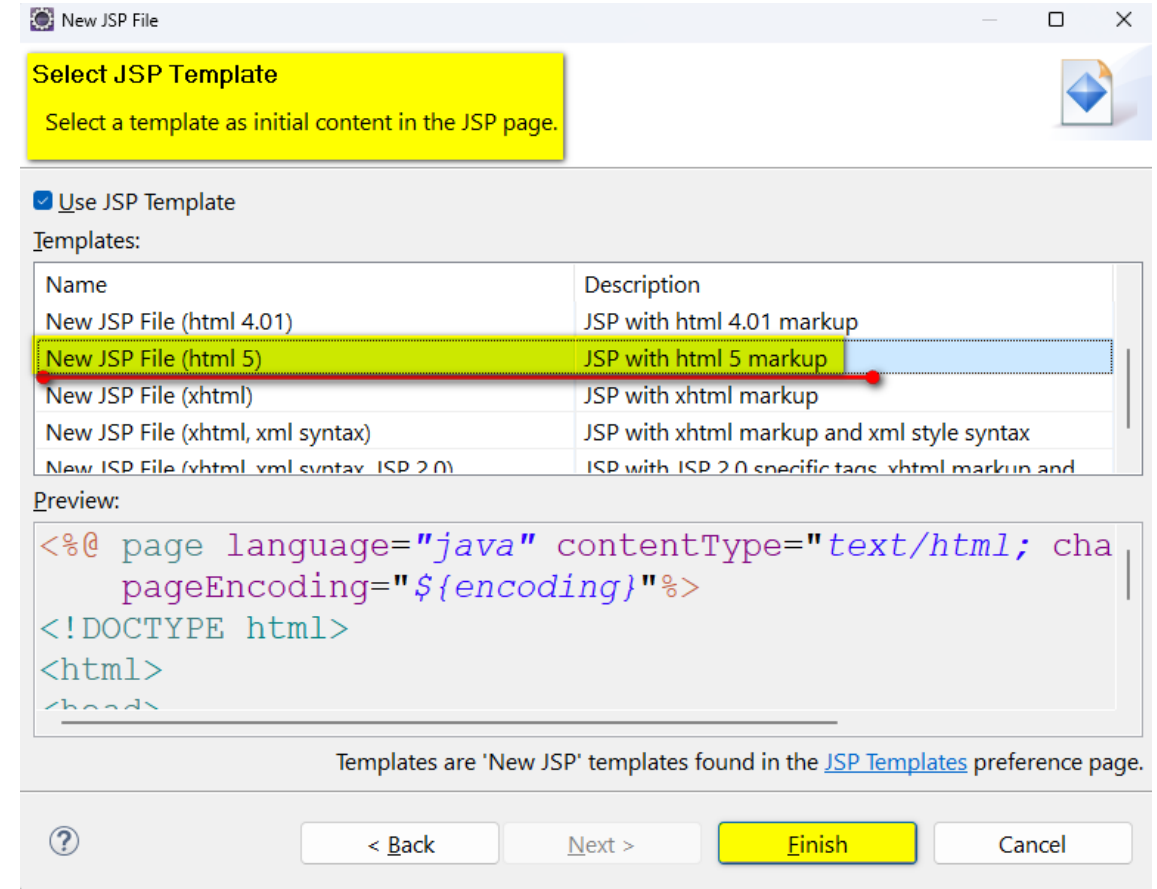
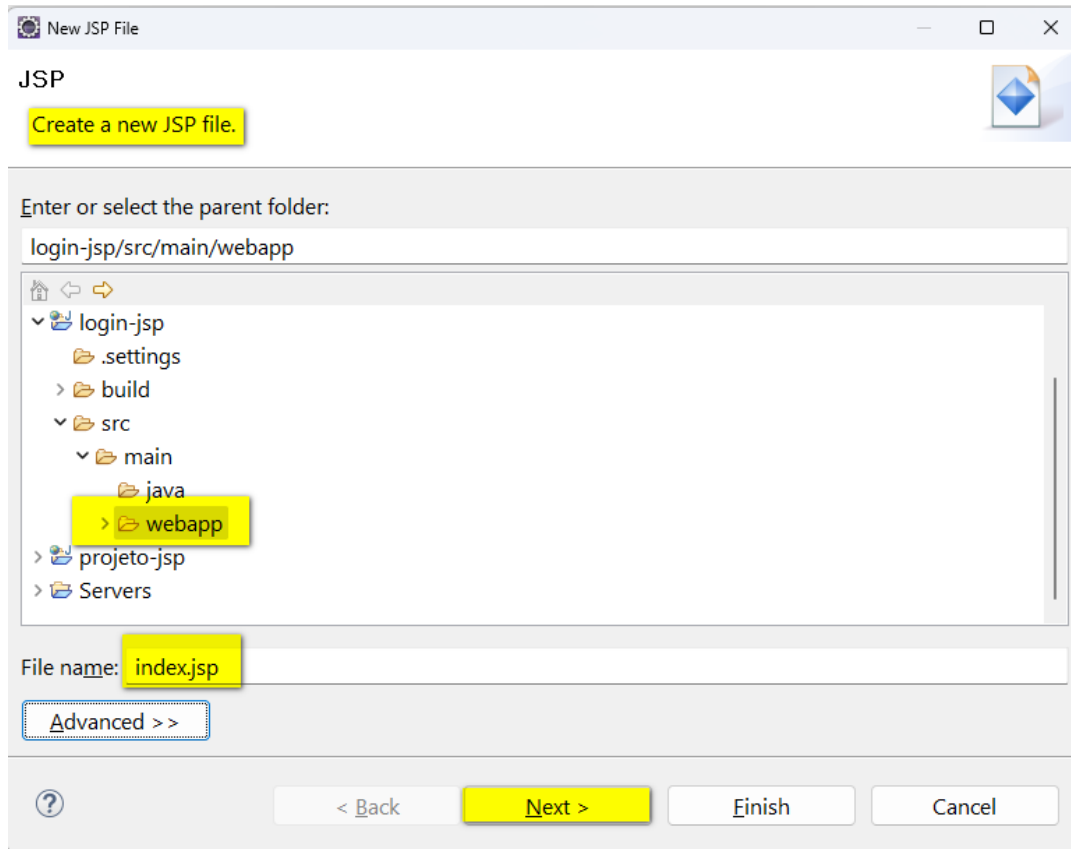
Projeto login-jsp



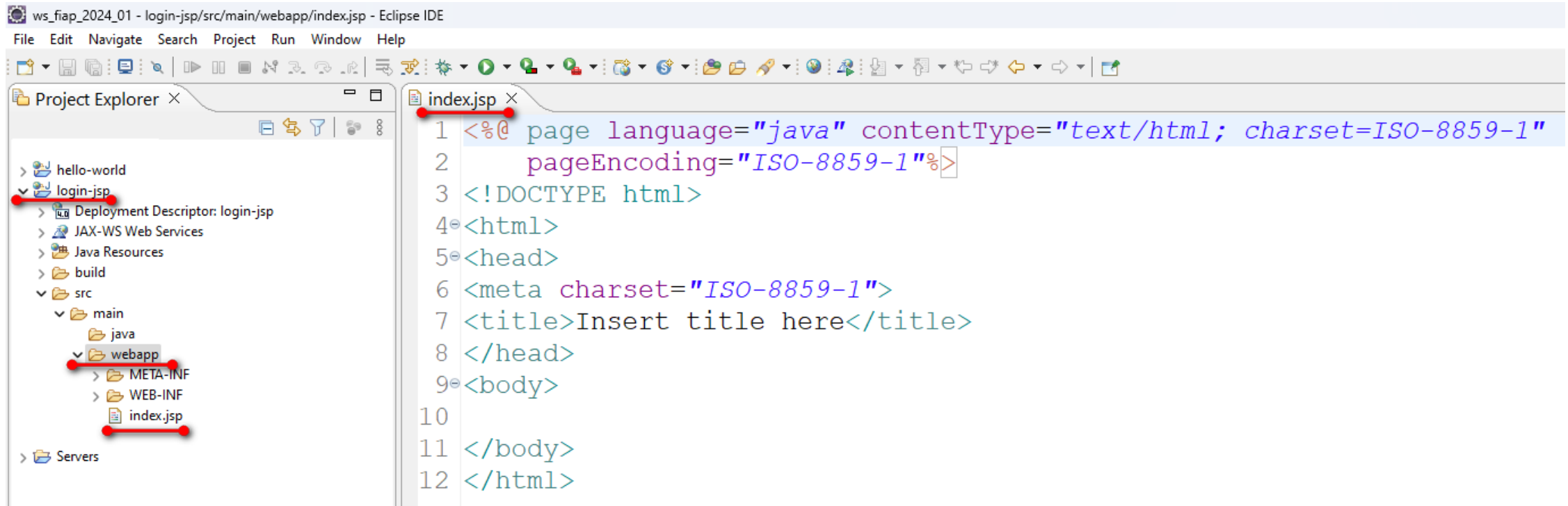
Projeto login-jsp



Projeto login-jsp



Projeto login-jsp



The screenshot shows the Eclipse IDE interface. The title bar indicates the project is 'ws_fiap_2024_01 - login-jsp/src/main/webapp/index.jsp - Eclipse IDE'. The menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and development tools. The Project Explorer on the left shows the project structure: 'hello-world' (expanded), 'login-jsp' (expanded), 'Deployment Descriptor: login-jsp', 'JAX-WS Web Services', 'Java Resources', 'build', 'src' (expanded), 'main' (expanded), 'java', 'webapp' (expanded), 'META-INF', 'WEB-INF', and 'index.jsp'. The editor on the right shows the content of 'index.jsp' with line numbers 1 through 12. The code is as follows:

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

Projeto login-jsp

```
index.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Index</title>
8 </head>
9 <body>
10
11 <form name="form" action="home.jsp" method="post">
12   <label for="nome">Nome:</label>
13   <input type="text" id="nome" name="nome" value="" size="30" placeholder="Nome"> <br /><br />
14   <label for="senha">Senha:</label>
15   <input type="password" id="senha" name="senha" value="" size="20" placeholder="Senha"> <br /><br />
16   <input type="submit" value="Entrar">
17 </form>
18
19 </body>
20 </html>
```

Projeto login-jsp

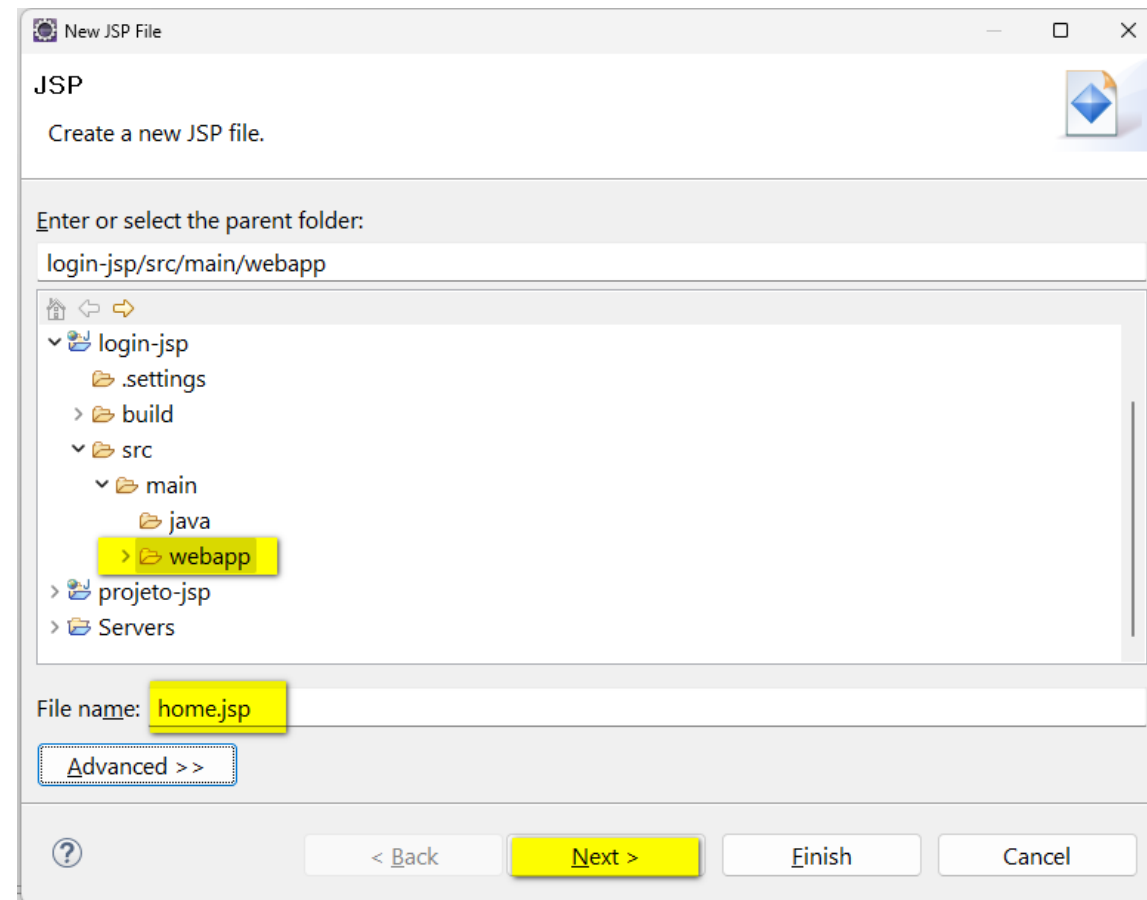
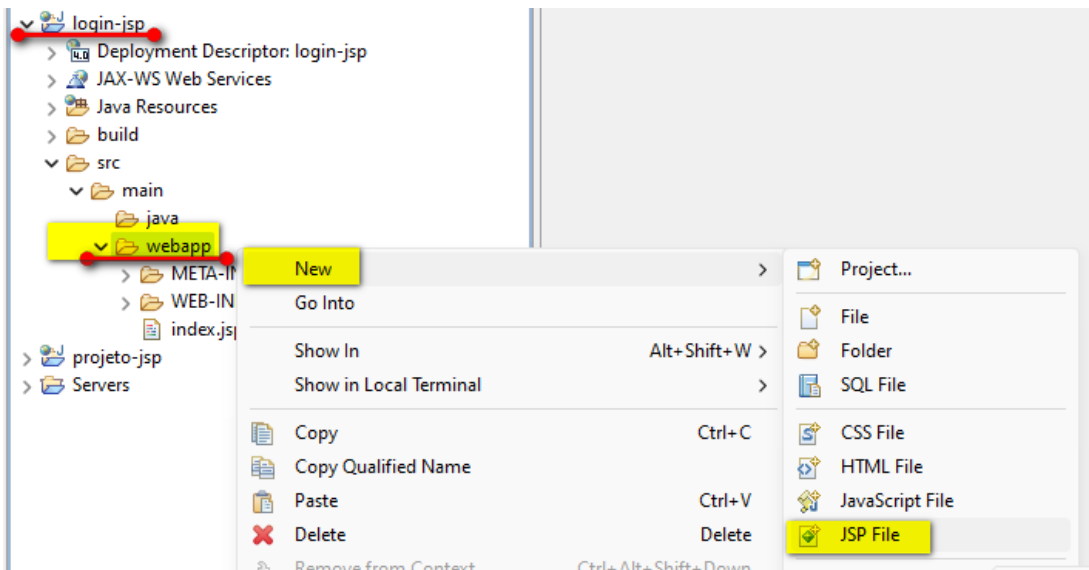
- Na tag <label>, colocamos o atributo **for="nome"** na tag de abertura, para indicar que essa label está associada ao campo com o ID "nome" da tag <input>.
- Na tag <input> o ID dentro da tag, como, por exemplo, **<input id="nome">**, para corresponder ao atributo **for** da label. Dessa forma, o navegador sabe que essa label é para o input específico.
- Dentro da tag **<input>**, definimos o atributo **name** com o valor **"nome"**, no nosso exemplo. Todos os campos em uma página **HTML** devem ter um atributo **name** para que seja possível recuperar as informações no código Java.

```
<label for="nome">Nome:</label>
```

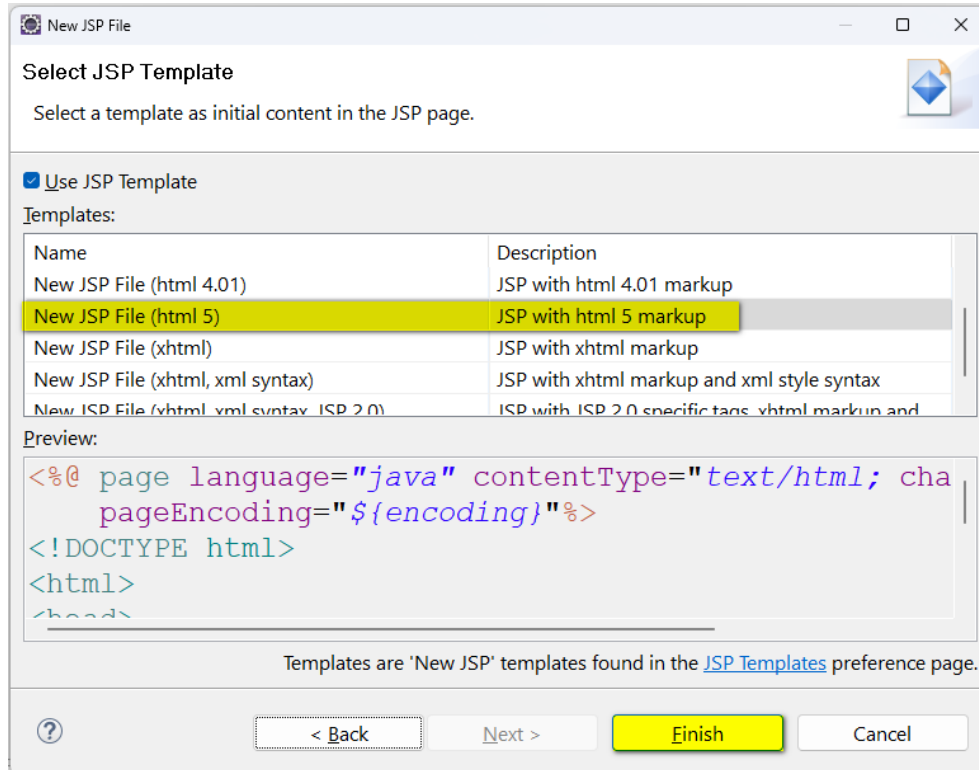
```
<input type="text" id="nome" name="nome" value="" size="30" placeholder="Nome"> <br /><br />
```

recuperar as informações no código Java.

Projeto login-jsp



Projeto login-jsp



```
*home.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

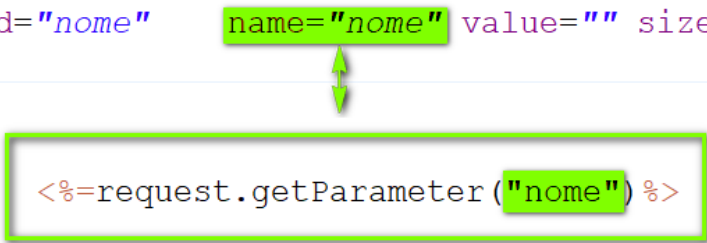
Projeto login-jsp

```
*home.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Home</title>
8 </head>
9 <body>
10
11     Nome:
12     <%=request.getParameter("nome") %>
13     <br>
14     <%
15     String senha = request.getParameter("senha");
16     if (senha.equals("123456")) {
17     %>
18     <b>Login efetuado com sucesso!!</b>
19     <%
20     } else {
21     %>
22     <b> Senha inválida!! </b>
23     <%
24     }
25     %>
26
27 </body>
28 </html>
```

Projeto login-jsp

```
<label for="nome">Nome:</label>
```

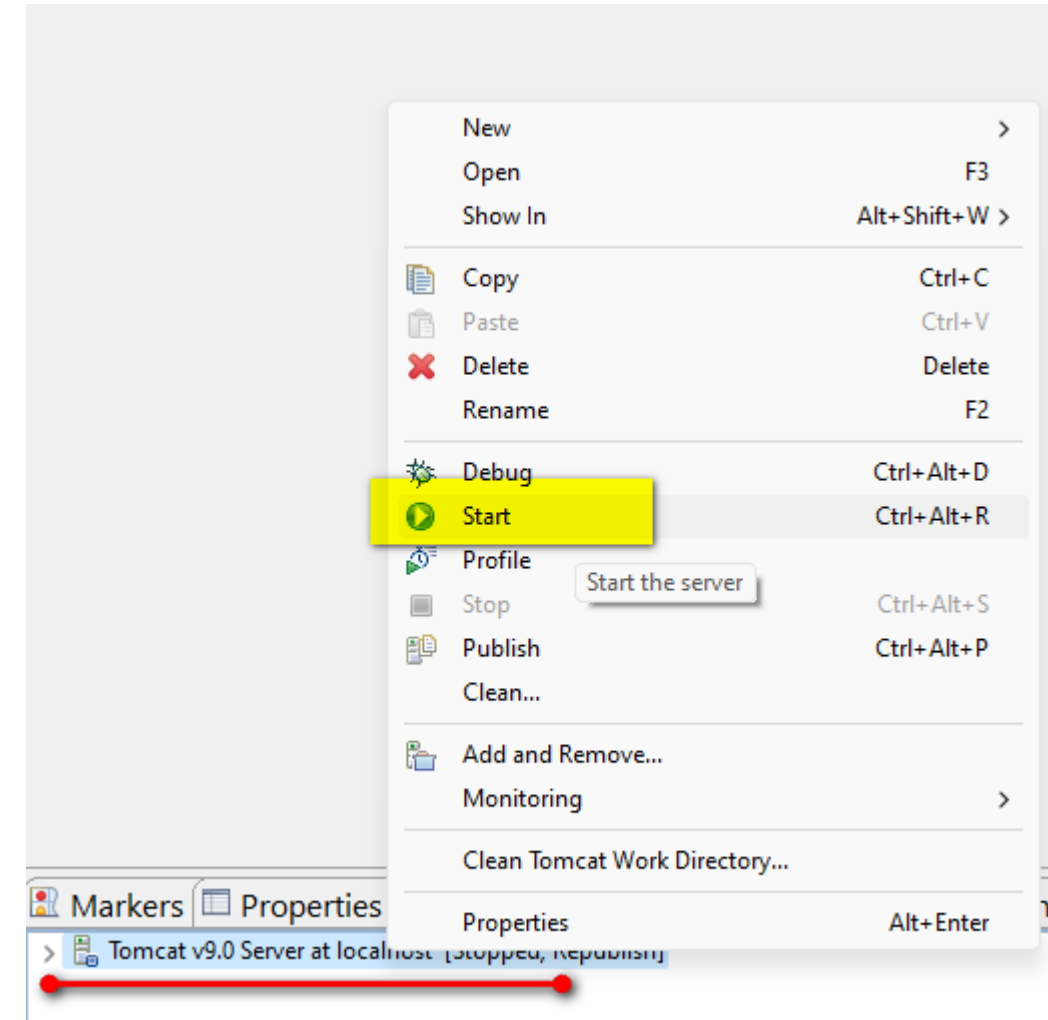
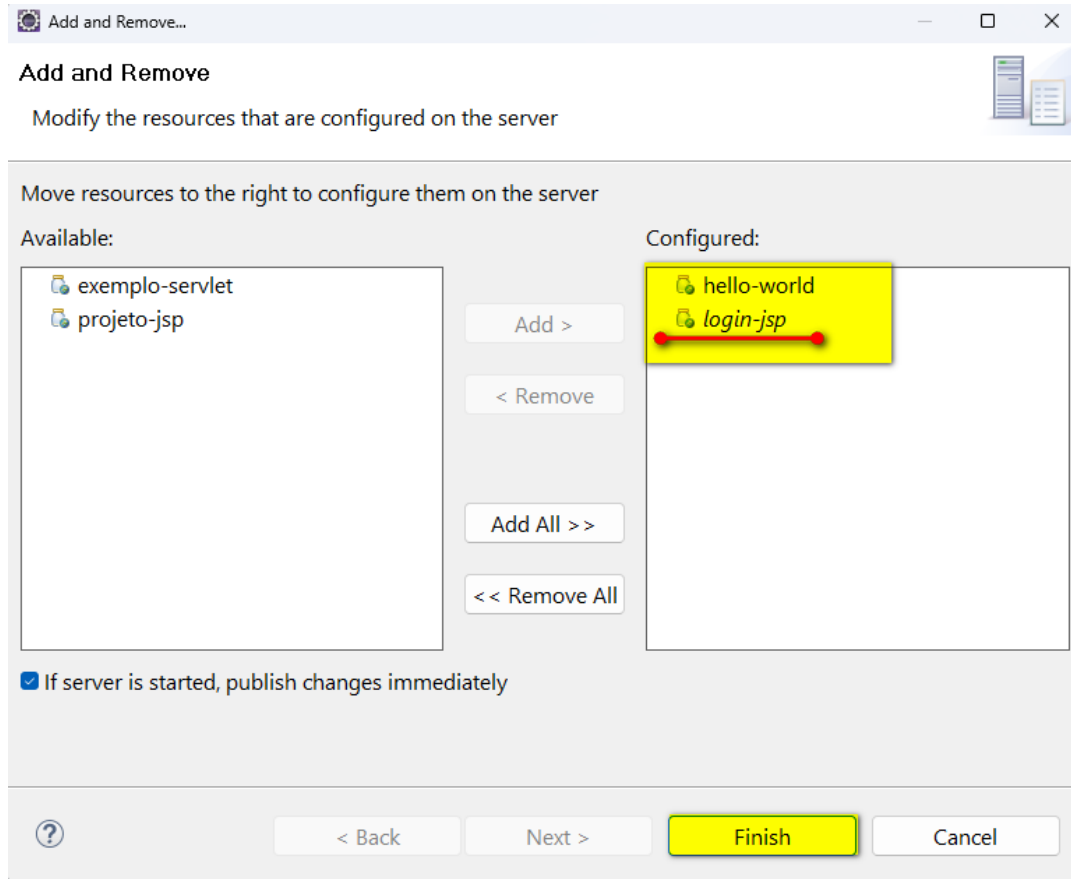
```
<input type="text" id="nome" name="nome" value="" size="30" placeholder="Nome"> <br /><br />
```



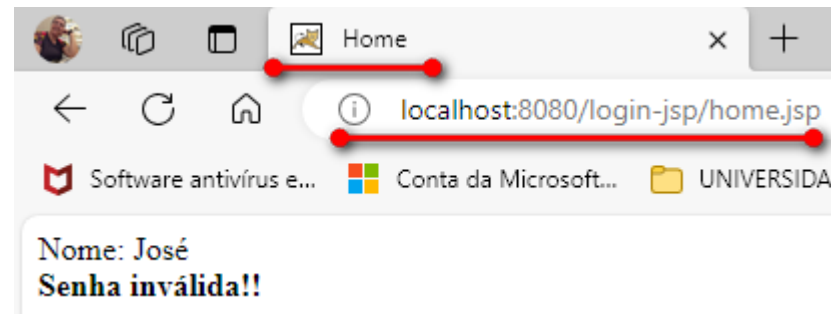
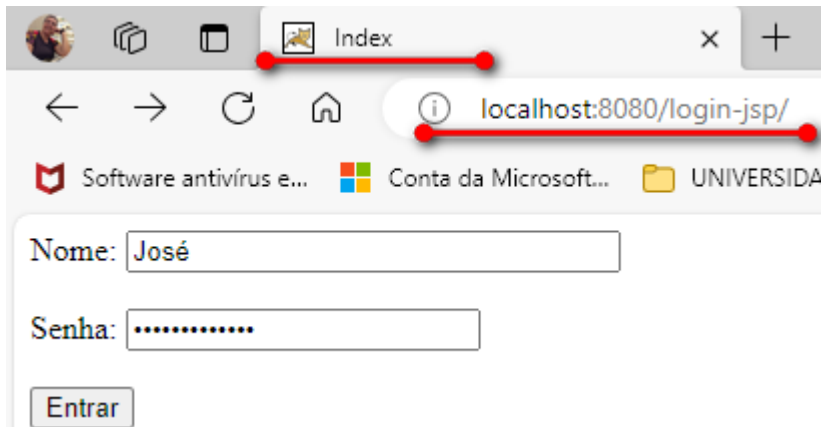
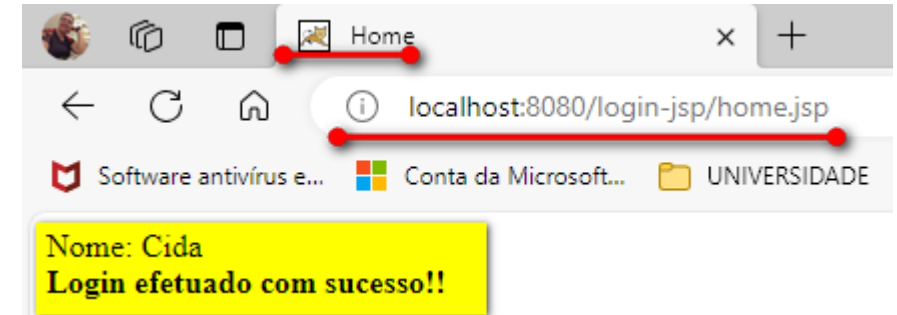
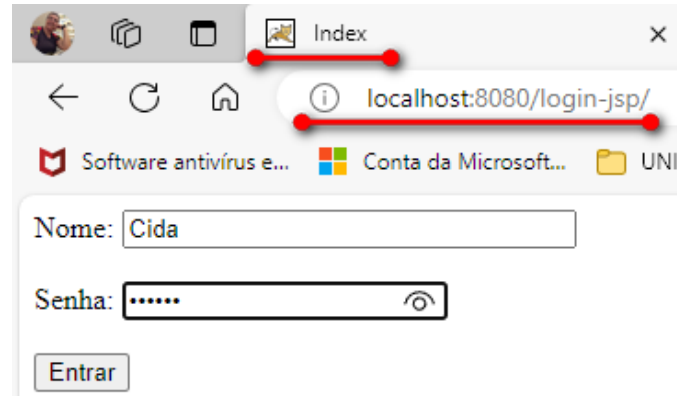
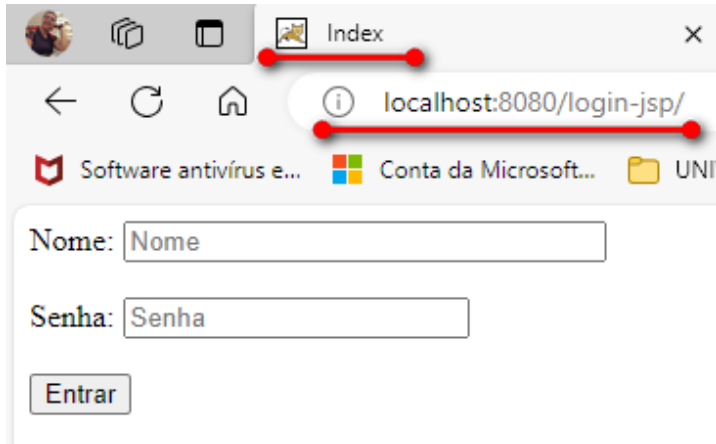
The diagram illustrates the data flow from the HTML input field to the JSP output. A green box highlights the `name="nome"` attribute in the `<input>` tag. A double-headed green arrow points from this box to another green box below it, which highlights the `"nome"` parameter in the `<%=request.getParameter("nome") %>` JSP expression. A horizontal line separates the HTML code from the JSP code.

```
<%=request.getParameter("nome") %>
```

Projeto login-jsp



Projeto login-jsp



Projeto login-jsp - Formulário Servlet

Criar um Formulário Servlet

Exercício 2

Simular Login

Projeto login-jsp – Formulário Servlet

1. Criar a **classe Index.java** e configurá-la para ser um **Servlet**:

```
@WebServlet("/index")  
public class Index extends HttpServlet
```

2. Criar o método **doPost** e realizar o redirect para a home:

```
request.getRequestDispatcher("home").forward(request, response);
```

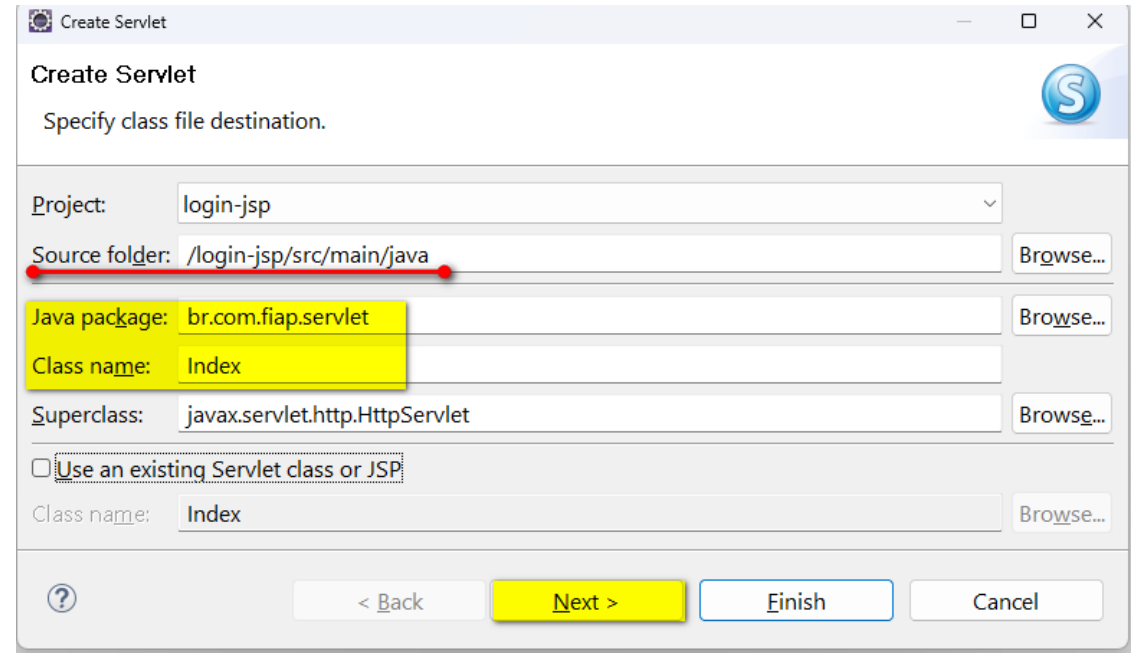
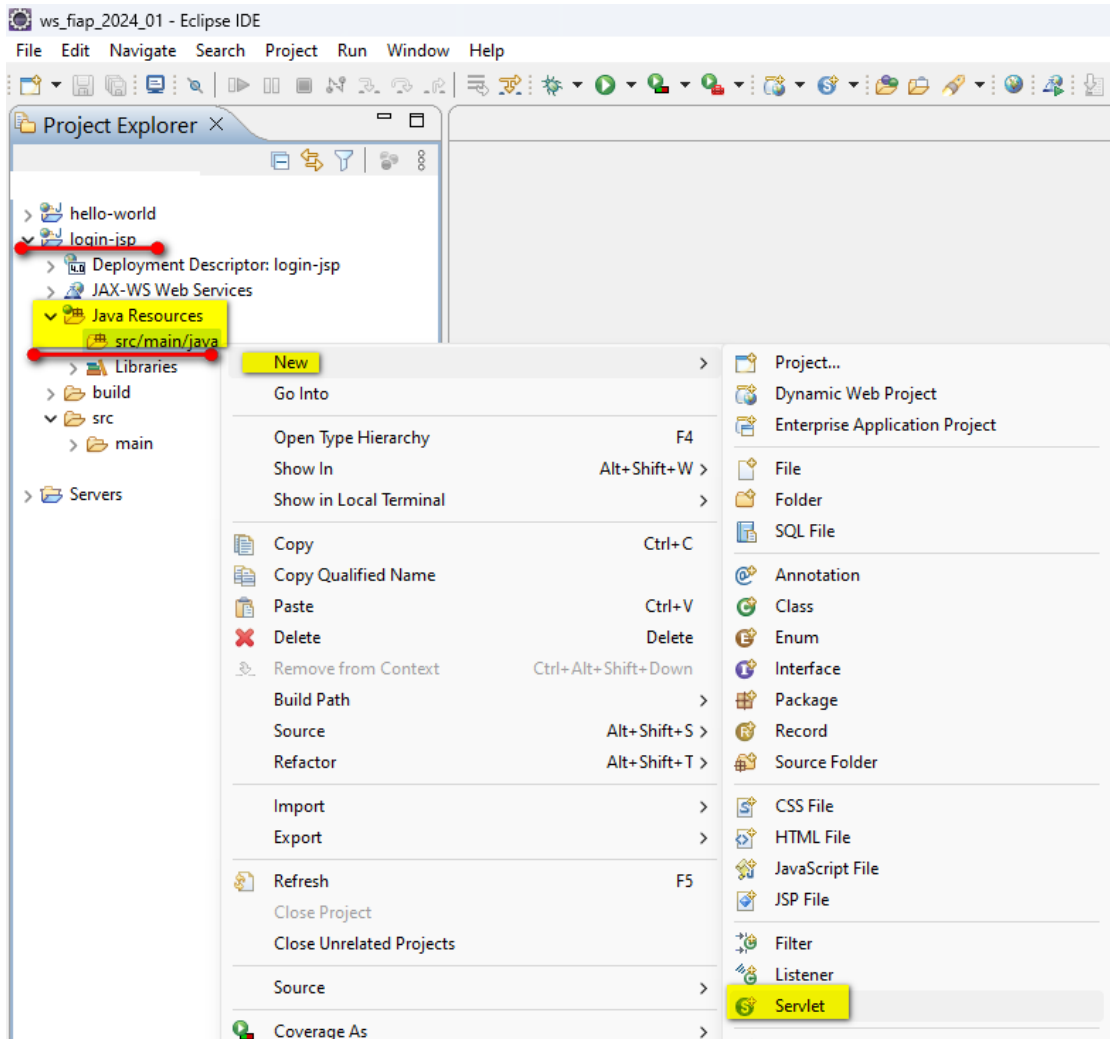
3. Alterar o **action** do formulário **index.jsp** para acessar o servlet index

```
<form name="form" action="index" method="post">
```

4. Criar a classe **Home.java** e configurá-la para ser um **Servlet**
5. Criar o método **doPost** e fazer a **lógica de validação de senha** que antes estava no JSP.

OBS: Parar o servidor

Projeto login-jsp – Formulário Servlet



Projeto login-jsp – Formulário Servlet

Create Servlet

Enter servlet deployment descriptor specific information.

Name:

Description:

Initialization parameters:

Name
/Index

URL mappings:

- /Index

☐ Asynchronous Support

< Back Next > Finish Cancel

Create Servlet

Specify modifiers, interfaces, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces:

Which method stubs would you like to create?

- ☒ Constructors from superclass
- ☒ Inherited abstract methods
- ☐ init
- ☐ destroy
- ☐ getServletConfig
- ☐ getServletInfo
- ☐ service
- ☒ doGet
- ☒ doPost
- ☐ doPut
- ☐ doDelete
- ☐ doHead
- ☐ doOptions
- ☐ doTrace

< Back Next > Finish Cancel

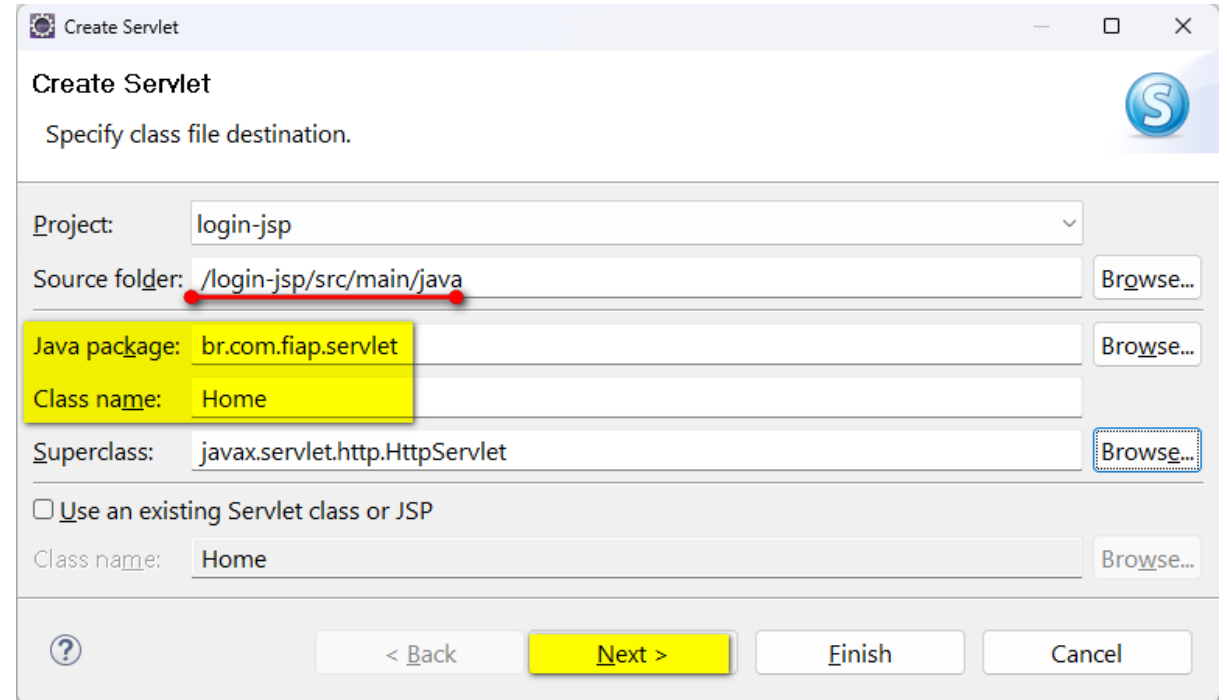
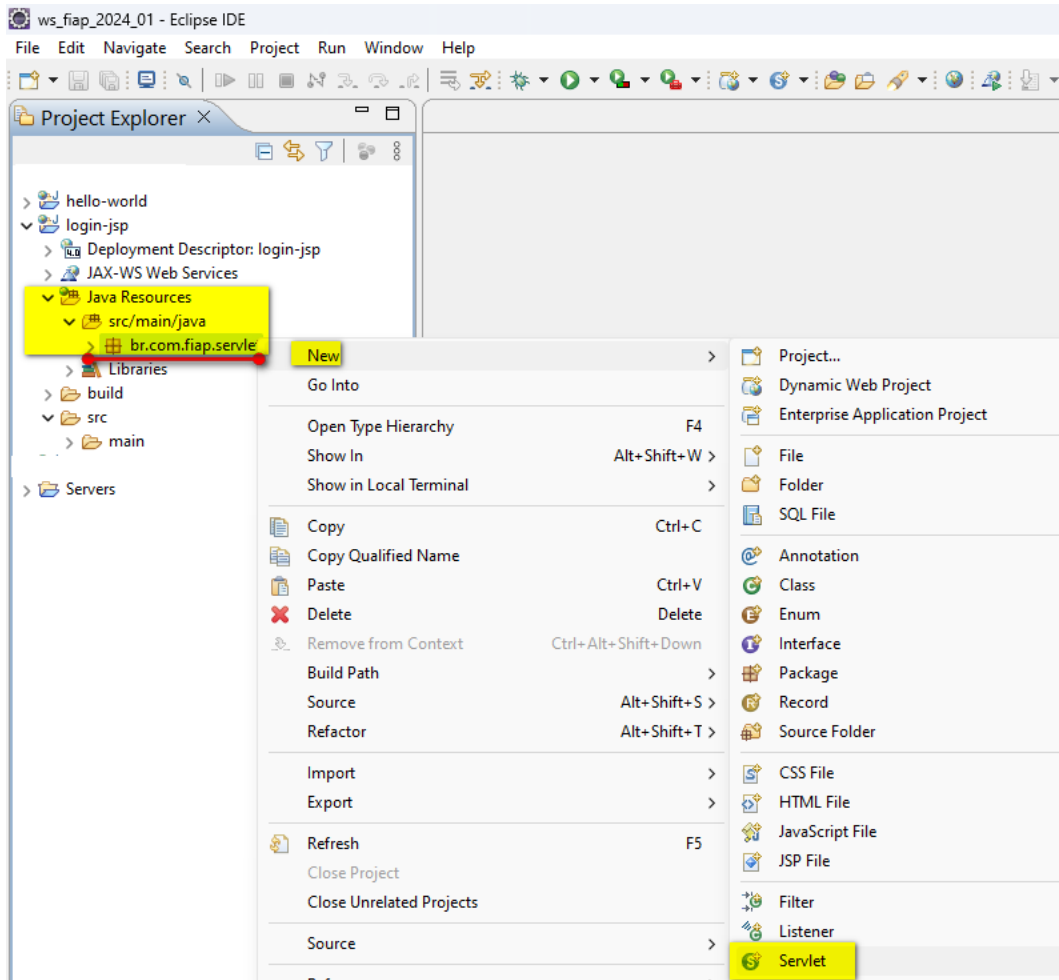
Projeto login-jsp – Formulário Servlet

```
Index.java x
1 package br.com.fiap.servlet;
2
3 import java.io.IOException;
4
5
6
7
8
9
10 @WebServlet("/index")
11 public class Index extends HttpServlet {
12     private static final long serialVersionUID = 1L;
13
14     public Index() {
15         super();
16     }
17
18     protected void doGet(HttpServletRequest request, HttpServletResponse response)
19         throws ServletException, IOException {
20
21         response.getWriter().append("Served at: ").append(request.getContextPath());
22     }
23
24     protected void doPost(HttpServletRequest request, HttpServletResponse response)
25         throws ServletException, IOException {
26
27         request.getRequestDispatcher("home").forward(request, response);
28     }
29 }
```

Projeto login-jsp – Alterar Formulário Servlet

```
index.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Index</title>
8 </head>
9 <body>
10
11 <form name="form" action="index" method="post">
12   <label for="nome">Nome:</label>
13   <input type="text" id="nome" name="nome" value="" size="30" placeholder="Nome"> <br /><br />
14   <label for="senha">Senha:</label>
15   <input type="password" id="senha" name="senha" value="" size="20" placeholder="Senha"> <br /><br />
16   <input type="submit" value="Entrar">
17 </form>
18
19 </body>
20 </html>
```

Projeto login-jsp – Formulário Servlet



Projeto login-jsp – Formulário Servlet

Create Servlet

Enter servlet deployment descriptor specific information.

Name: Home

Description:

Initialization parameters:

URL mappings:

/Home

Asynchronous Support

URL Mappings dialog:

Pattern: /home

OK Cancel

Navigation: < Back Next > Finish Cancel

Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces:

Which method stubs would you like to create?

☒ Constructors from superclass

☒ Inherited abstract methods

☐ init ☐ destroy ☐ getServletConfig

☐ getServletInfo ☐ service ☒ doGet

☒ doPost ☐ doPut ☐ doDelete

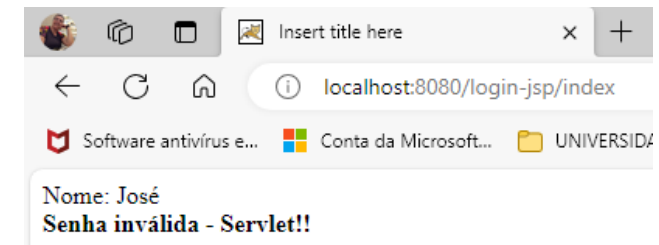
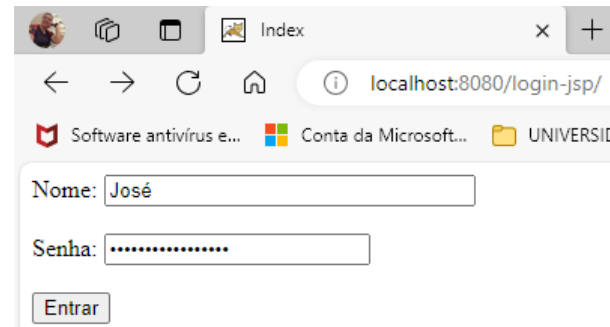
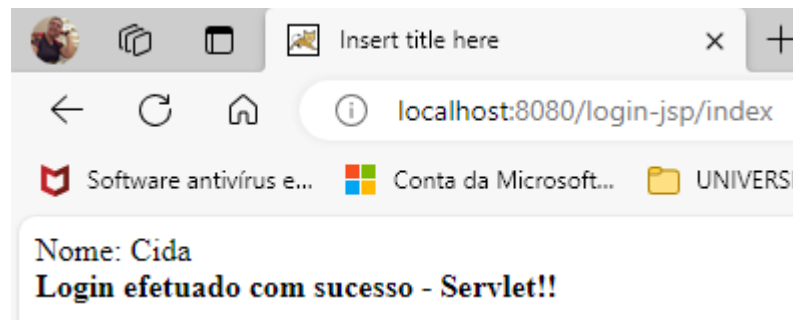
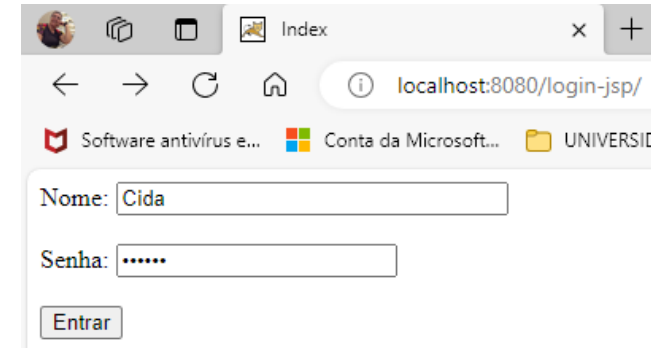
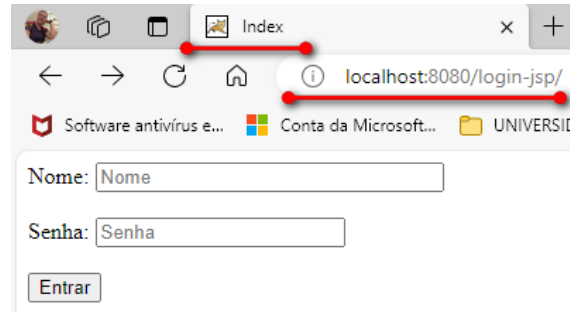
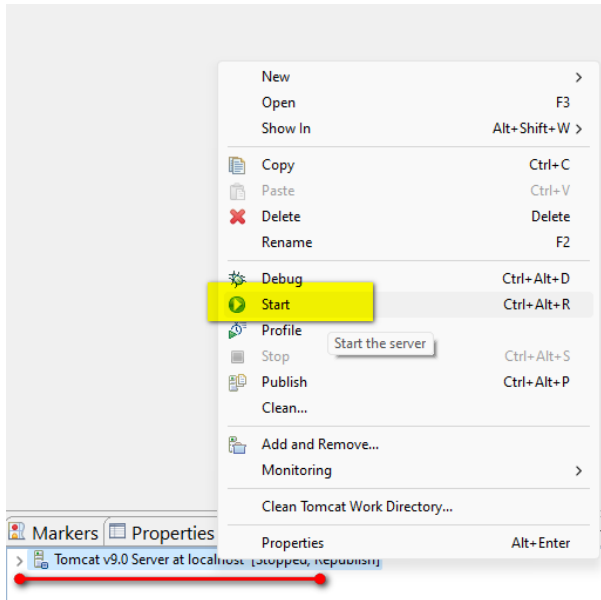
☐ doHead ☐ doOptions ☐ doTrace

Navigation: < Back Next > Finish Cancel

Projeto login-jsp – Formulário Servlet

```
12 @WebServlet("/home")
13 public class Home extends HttpServlet {
14     private static final long serialVersionUID = 1L;
15
16     public Home() {}
17
18
19
20     protected void doGet(HttpServletRequest request, HttpServletResponse response) {}
21
22
23
24
25     protected void doPost(HttpServletRequest request, HttpServletResponse response)
26         throws ServletException, IOException {
27
28         String nome = request.getParameter("nome");
29         String senha = request.getParameter("senha");
30         String mensagem = "Login efetuado com sucesso - Servlet!!";
31
32         if(! senha.equals("123456")) {
33             mensagem = "Senha inválida - Servlet!!";
34         }
35         response.setContentType("text/html");
36
37         PrintWriter out = response.getWriter();
38         out.println("<html> ");
39         out.println("<head> ");
40         out.println("<meta http-equiv='Content-Type' content='text/html; charset=ISO-8859-1'> ");
41         out.println("<title>Insert title here</title> ");
42         out.println("</head> ");
43         out.println("<body> ");
44         out.println(" Nome: " + nome + " <br> ");
45         out.println(" <b> " + mensagem + "</b> ");
46         out.println("</body> ");
47         out.println("</html> ");
48     }
49 }
```

Projeto login-jsp – Formulário Servlet





Desafios

Vamos praticar?

Retângulo

Retângulo

Vamos calcular a área e o perímetro do retângulo?

Base:

Altura:

Calcular

Cancelar

Retângulo

Vamos calcular a área e o perímetro do retângulo?

Base:

Altura:

Calcular

Cancelar

Resultado

Área: 6.0

Perímetro: 2.5

Salve! Salve!

Salve! Salve!

Nome:

Sobrenome:

E-mail:

Salve! Salve!

Nome:

Sobrenome:

E-mail:

Bem vindo(a)

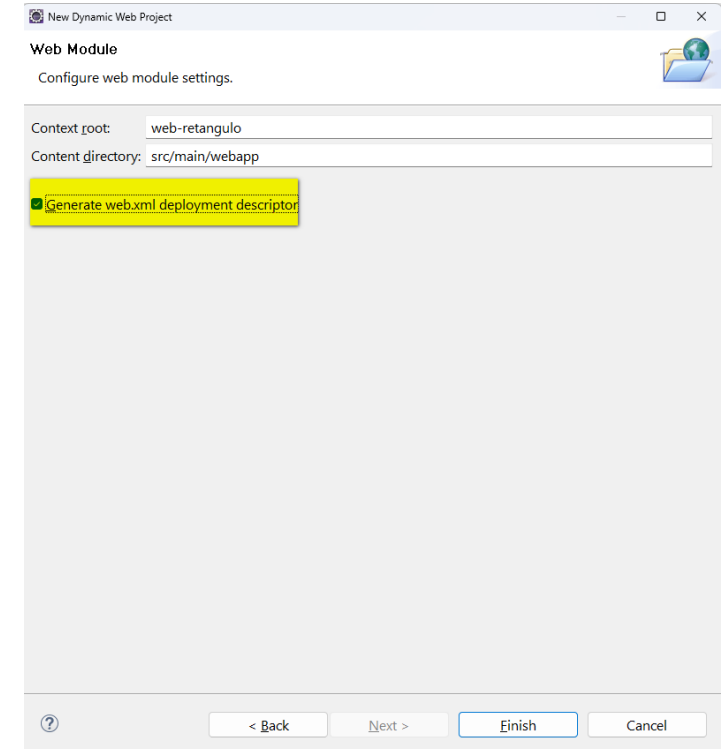
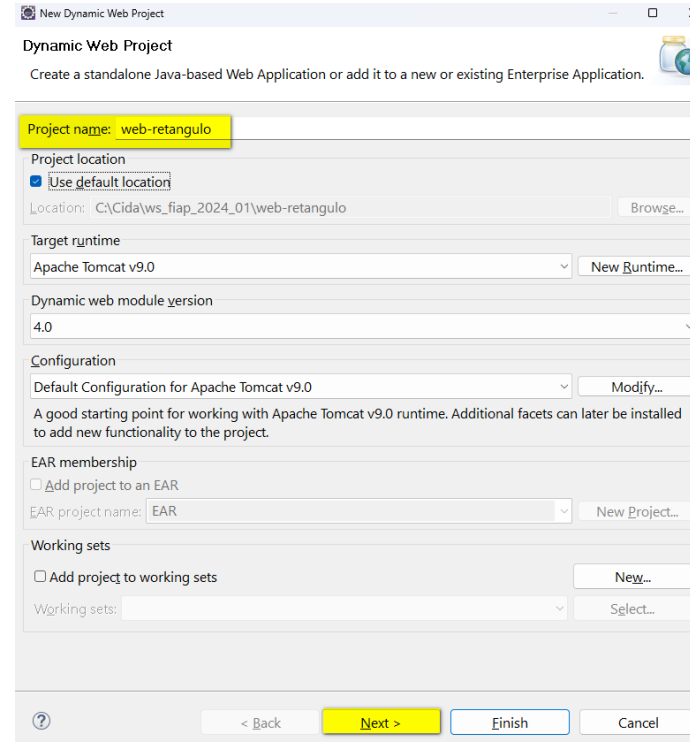
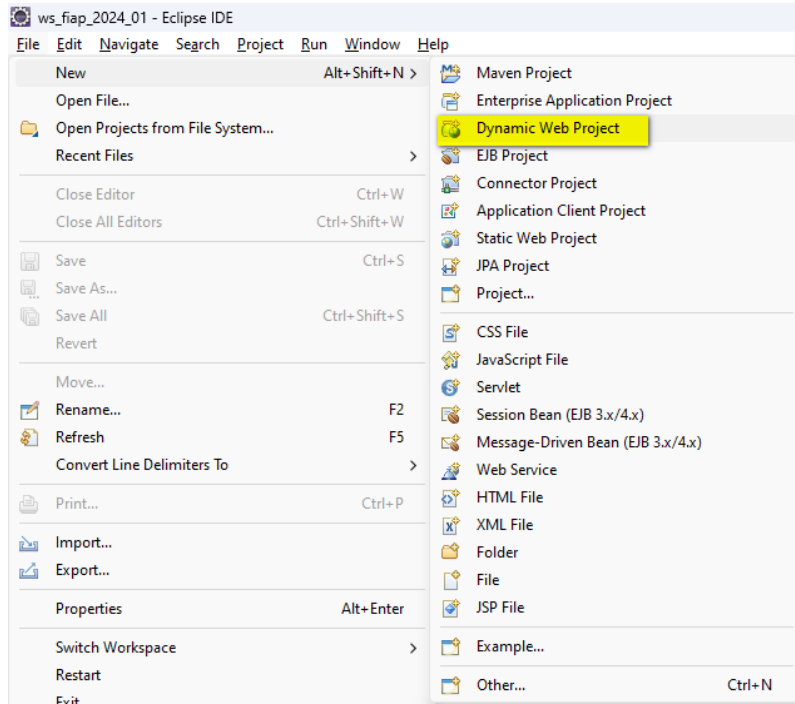
Olá, Cida Castello

E-mail: cida@gmail.com

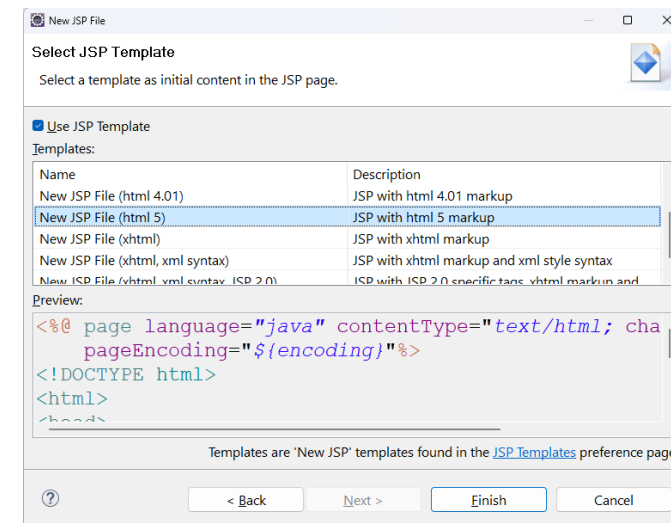
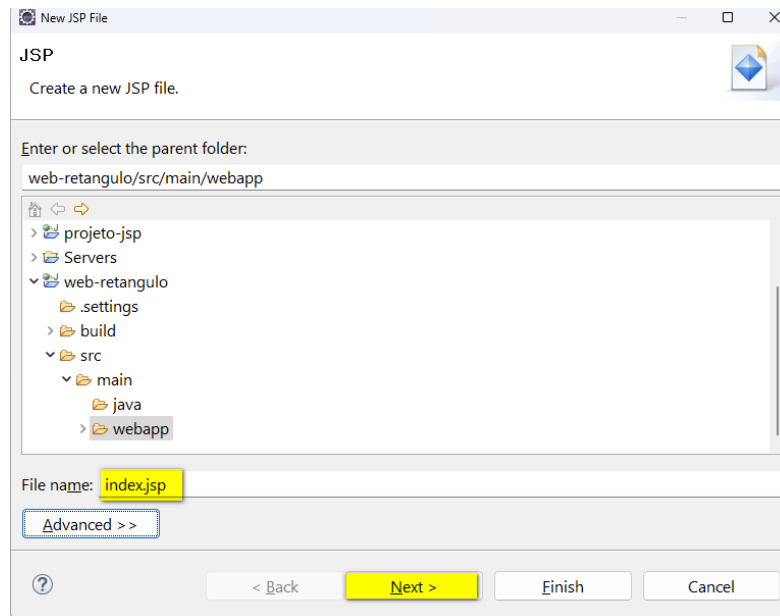
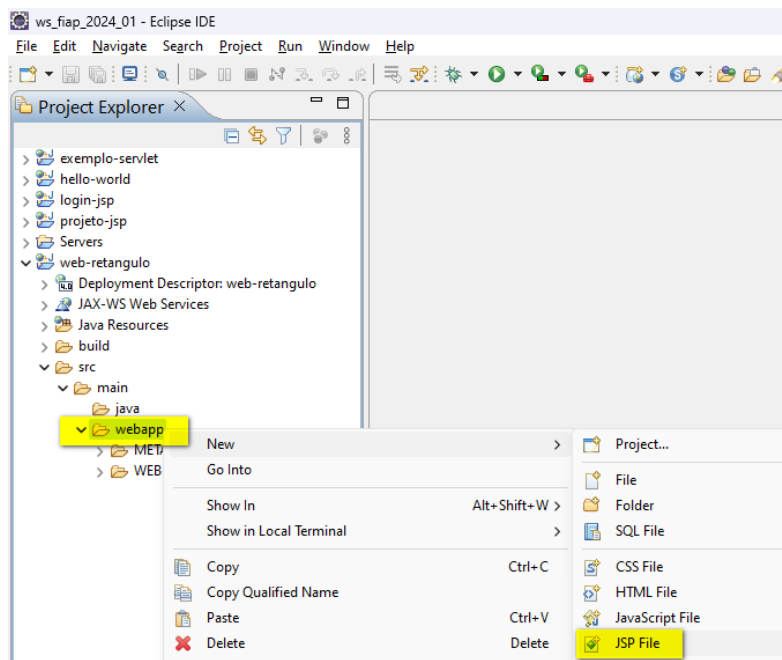
web-retangulo

Solução

web-retangulo



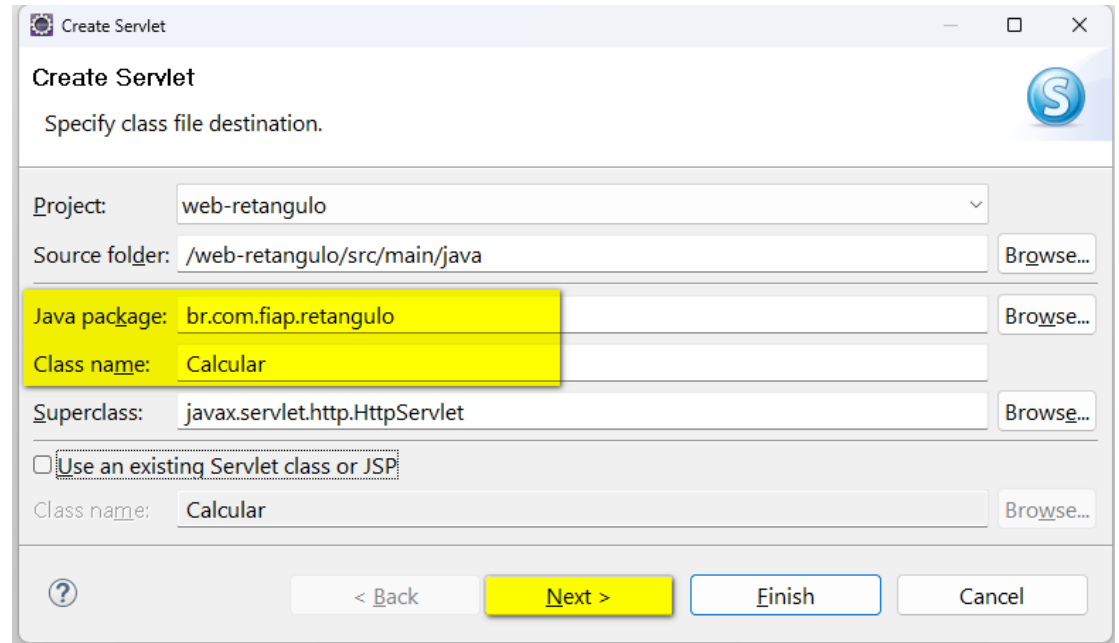
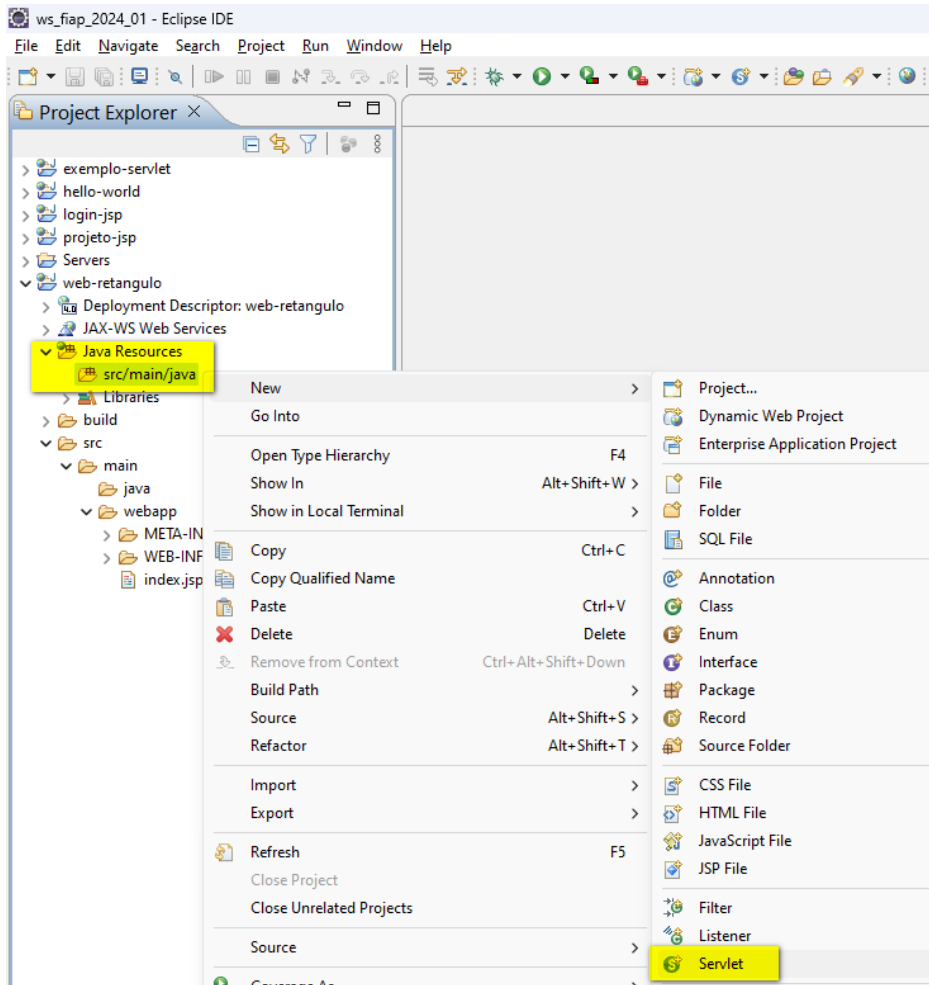
web-retângulo – index.jsp



web-retângulo – index.jsp

```
*index.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Web-Retângulo</title>
8 </head>
9 <body>
10
11 <h1>Retângulo</h1>
12   <h3>Vamos calcular a área e o perímetro do retângulo?</h3>
13
14   <form name="form" action="calcular" method="post">
15     <label for="base">Base:</label>
16     <input type="text" id="base" name="base" placeholder="Base" required>
17     <br><br>
18     <label for="altura">Altura:</label>
19     <input type="text" id="altura" name="altura" placeholder="Altura" required>
20     <br><br>
21     <input type="submit" value="Calcular"><br><br>
22     <input type="reset" value="Cancelar">
23   </form>
24
25 </body>
26 </html>
```


web-retângulo – Servlet calcular



web-retângulo – Servlet calcular

Create Servlet

Enter servlet deployment descriptor specific information.

Name:

Description:

Initialization parameters:

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

URL mappings:

/Calcular

☐ Asynchronous Support

< Back Next > Finish Cancel

Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces:

Which method stubs would you like to create?

☒ Constructors from superclass

☒ Inherited abstract methods

<input type="checkbox"/> init	<input type="checkbox"/> destroy	<input type="checkbox"/> getServletConfig
<input type="checkbox"/> getServletInfo	<input type="checkbox"/> service	<input checked="" type="checkbox"/> doGet
<input checked="" type="checkbox"/> doPost	<input type="checkbox"/> doPut	<input type="checkbox"/> doDelete
<input type="checkbox"/> doHead	<input type="checkbox"/> doOptions	<input type="checkbox"/> doTrace

? < Back Next > Finish Cancel

class Calcular – doPost()

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

    Double base = Double.parseDouble(request.getParameter("base"));
    Double altura = Double.parseDouble(request.getParameter("altura"));

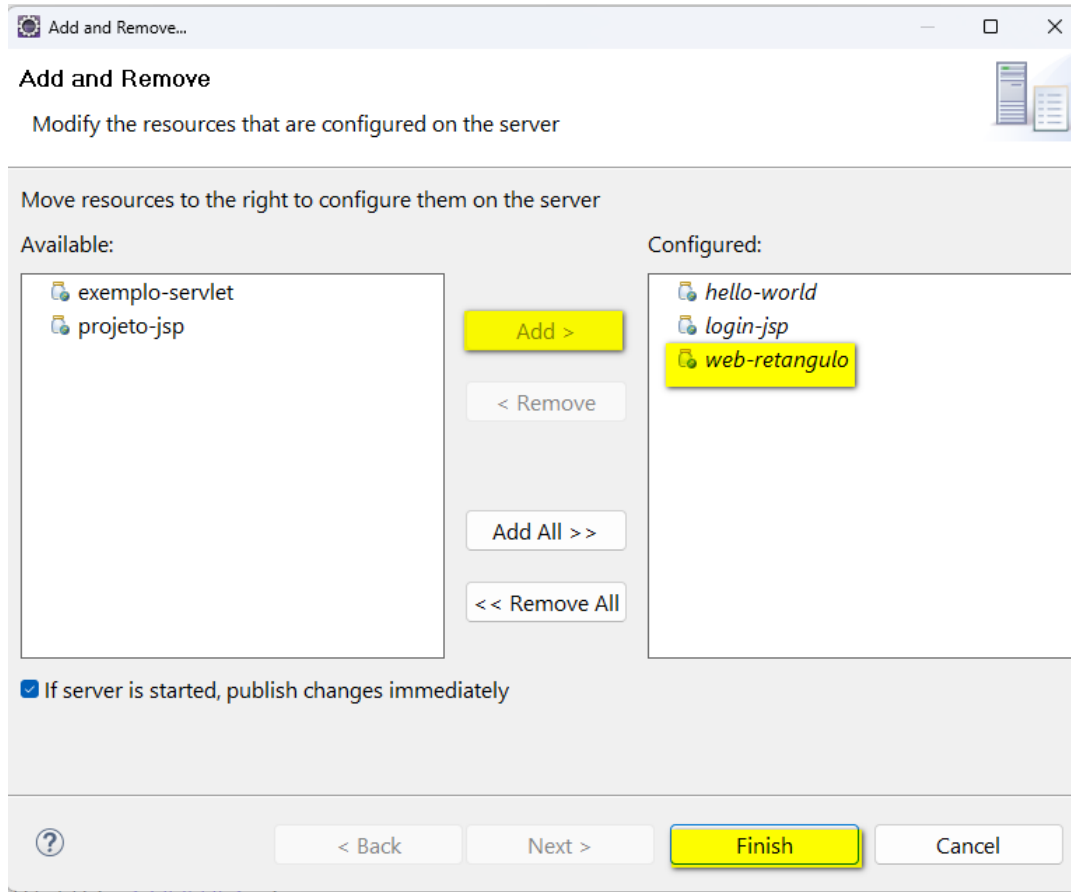
    Double area = base * altura;
    Double perimetro = (base + altura) / 2;

    response.setContentType("text/html");

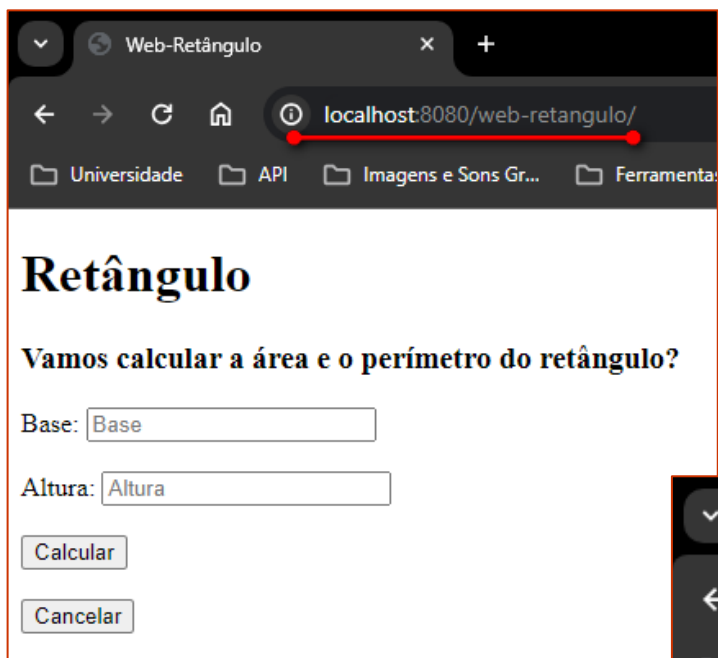
    PrintWriter out = response.getWriter();

    out.println("<!DOCTYPE html>");
    out.println("<html>");
    out.println("<head>");
    out.println("<meta charset=ISO-8859-1>");
    out.println("<title>App Retângulo</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Resultado</h1>");
    out.println("<h3>Área: " + area + "</h3>");
    out.println("<h3>Perímetro: " + perimetro + "</h3>");
    out.println("</body>");
    out.println("</html>");
}
```

Executando a aplicação web-retângulo



web-retângulo



Web-Retângulo

localhost:8080/web-retangulo/

Universidade API Imagens e Sons Gr... Ferramenta

Retângulo

Vamos calcular a área e o perímetro do retângulo?

Base:

Altura:



App Retângulo

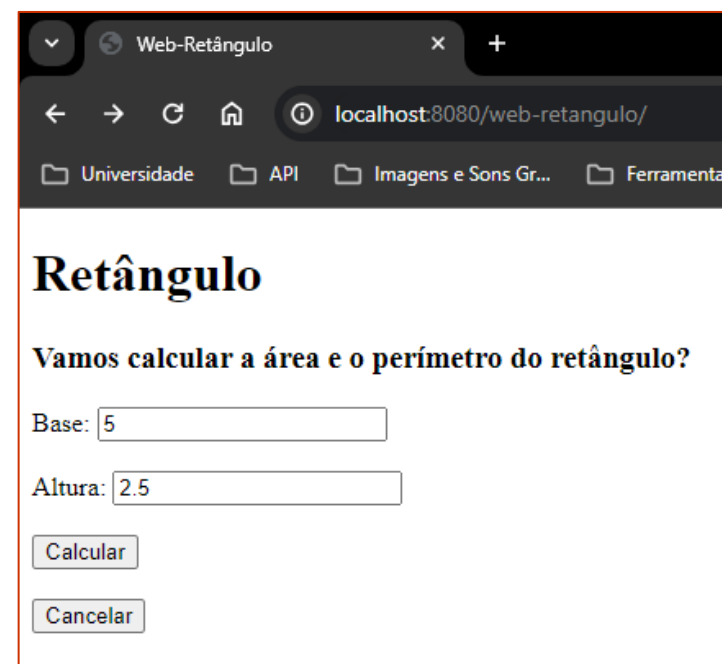
localhost:8080/web-retangulo/calcular

Universidade API Imagens e Sons Gr... Ferramentas

Resultado

Área: 12.5

Perímetro: 3.75



Web-Retângulo

localhost:8080/web-retangulo/

Universidade API Imagens e Sons Gr... Ferramenta

Retângulo

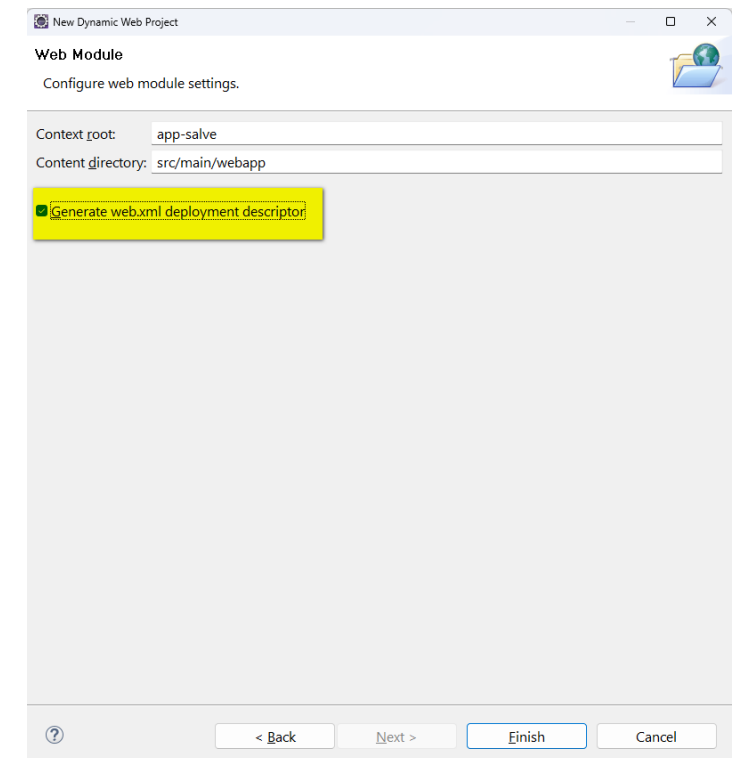
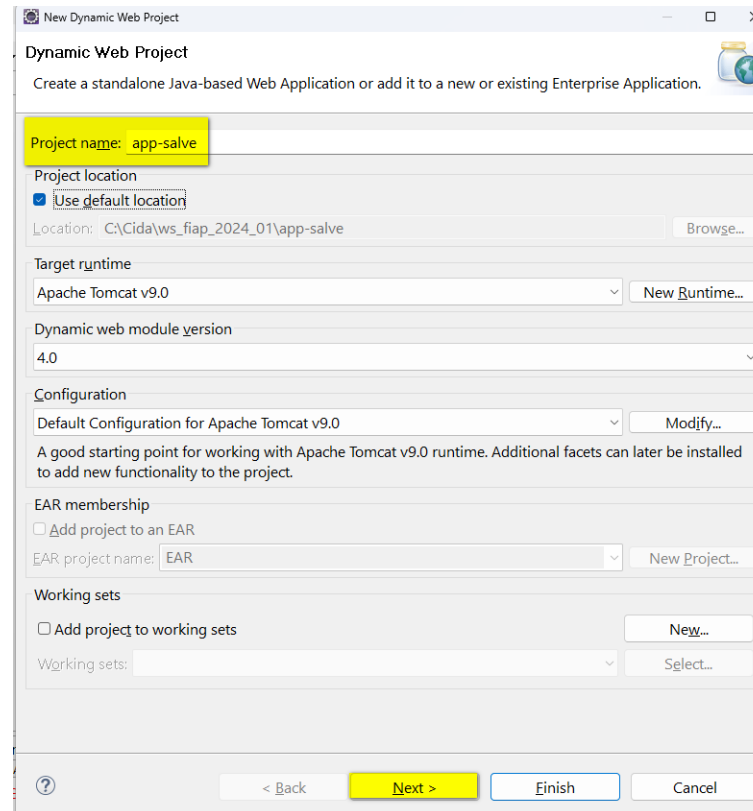
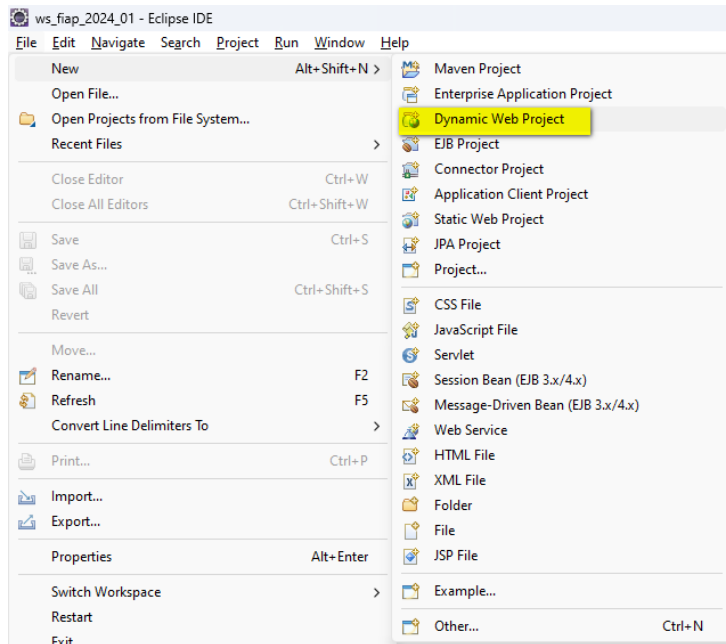
Vamos calcular a área e o perímetro do retângulo?

Base:

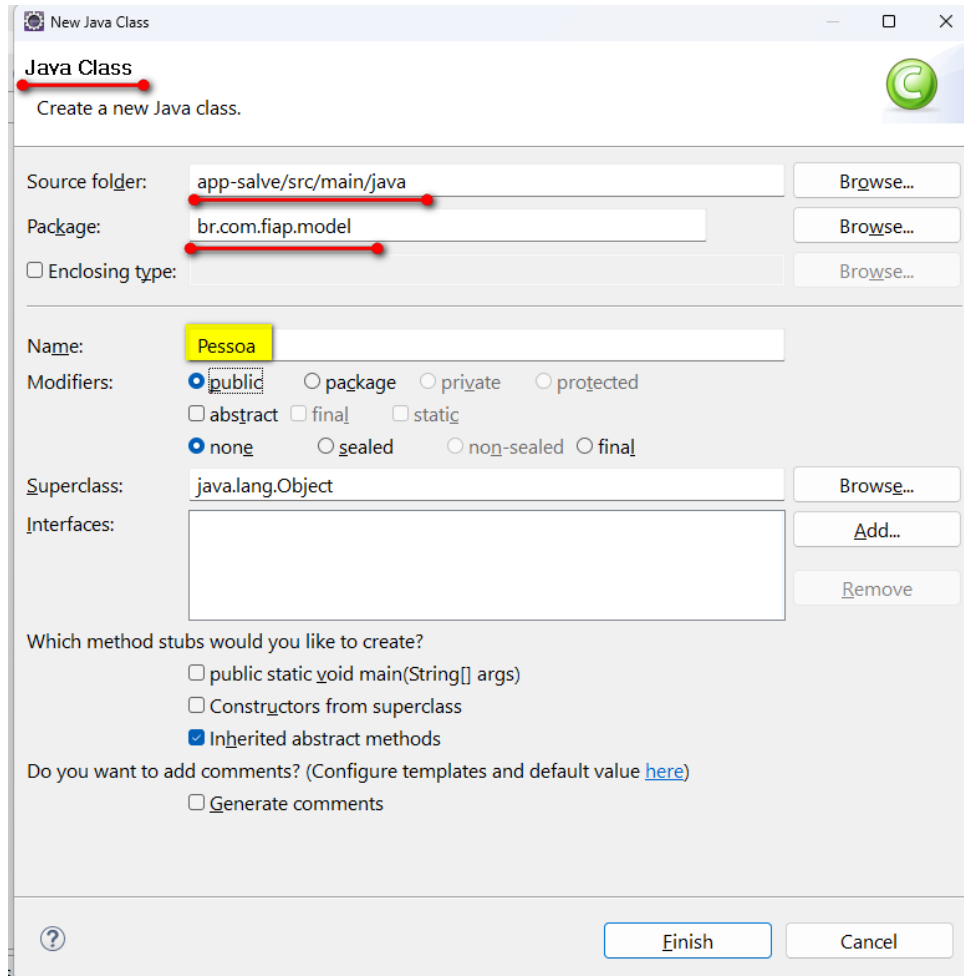
Altura:

app-salve

Solução

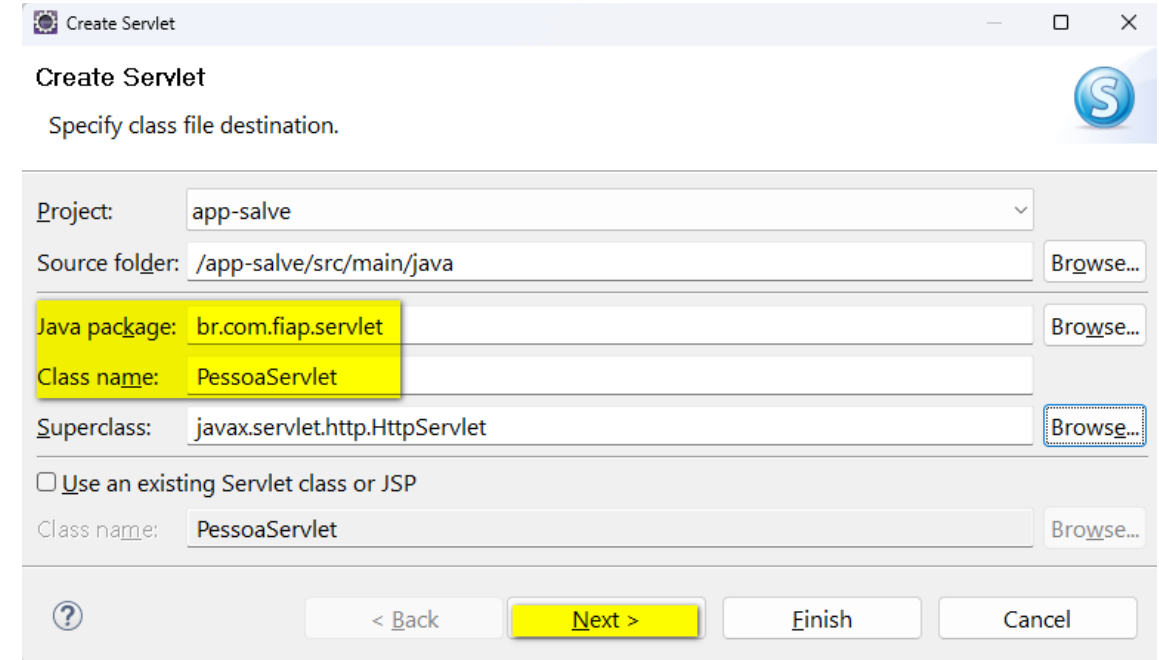
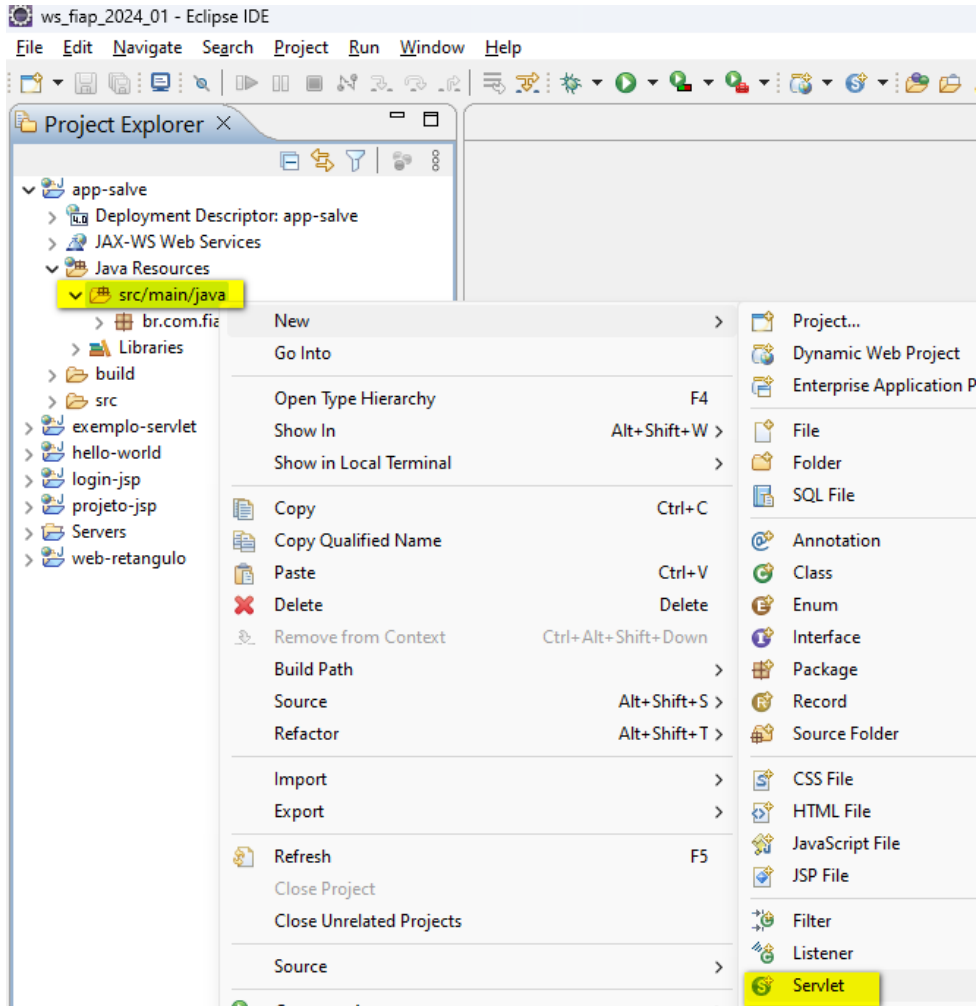


app-salve – Classe Pessoa (JavaBean)



```
1 package br.com.fiap.model;
2
3 public class Pessoa {
4
5     private String nome;
6     private String sobrenome;
7     private String email;
8
9     //incluir construtores com e sem argumentos
10    //incluir getters e setter
11
12    //incluir o método abaixo
13    public String getNomeCompleto() {
14        return nome + " " + sobrenome;
15    }
16 }
```


app-salve – Servlet PessoaServlet



app-salve – Servlet PessoaServlet

Create Servlet

Enter servlet deployment descriptor specific information.

Name: PessoaServlet

Description:

Initialization parameters:

URL Mappings

Pattern: /pessoa-servlet

OK

Cancel

URL mappings:

/PessoaServlet

Add...

Edit...

Remove

☐ Asynchronous Support

< Back

Next >

Finish

Cancel

Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces:

Add...

Remove

Which method stubs would you like to create?

☐ Constructors from superclass

☒ Inherited abstract methods

☐ init ☐ destroy ☐ getServletConfig

☐ getServletInfo ☐ service ☒ doGet

☒ doPost ☐ doPut ☐ doDelete

☐ doHead ☐ doOptions ☐ doTrace

< Back

Next >

Finish

Cancel

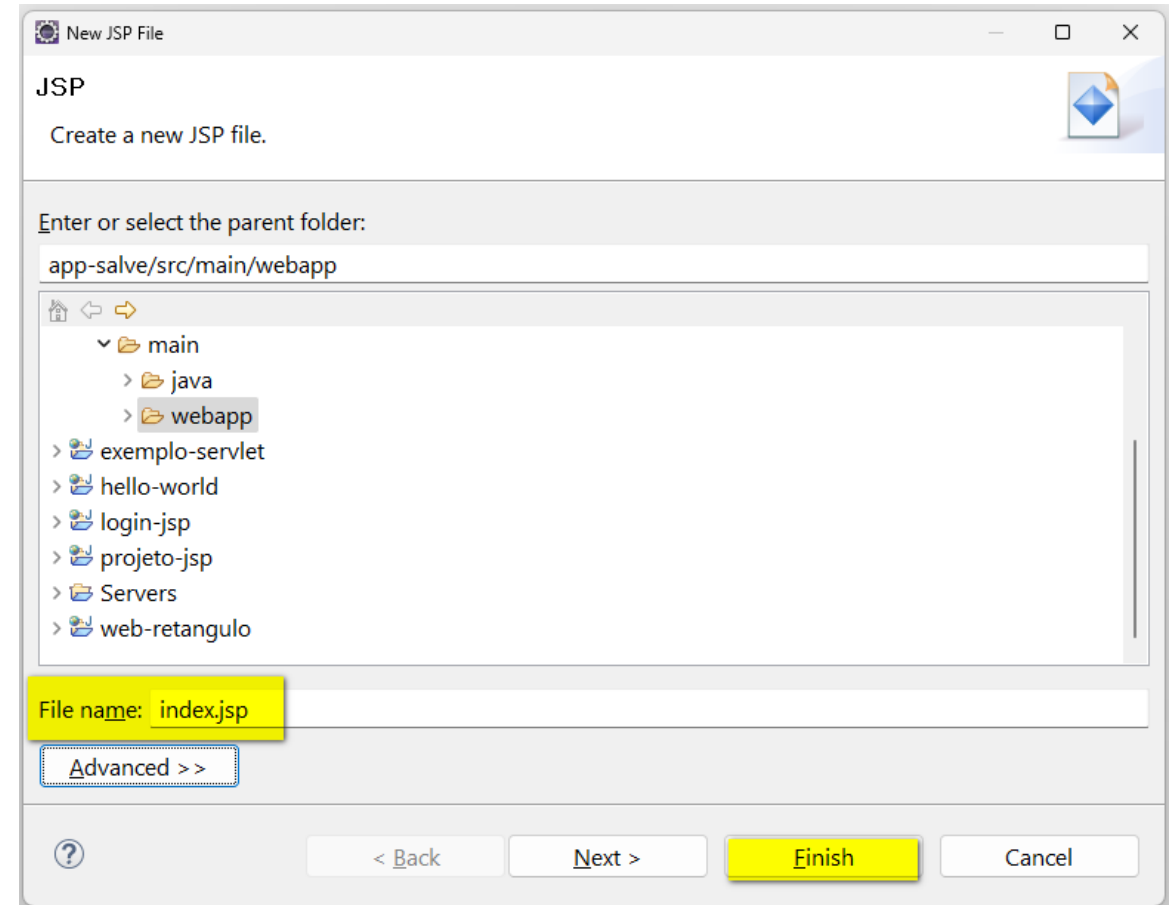
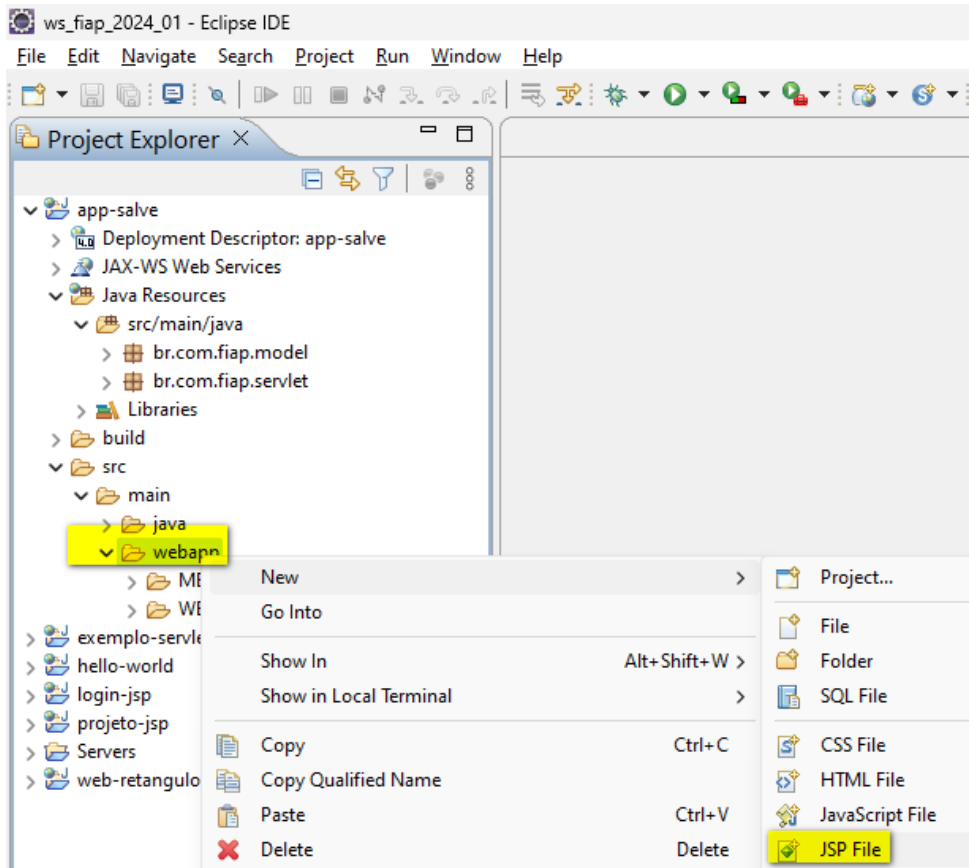
class PessoaServlet

```
*PessoaServlet.java ×
1 package br.com.fiap.servlet;
2
3 import java.io.IOException;
4
5 import javax.servlet.RequestDispatcher;
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 import br.com.fiap.model.Pessoa;
13
14 @WebServlet("/pessoa-servlet")
15 public class PessoaServlet extends HttpServlet {
16     private static final long serialVersionUID = 1L;
17
18     protected void doGet(HttpServletRequest request, HttpServletResponse response)
19         throws ServletException, IOException {
20         this.processaRequisicao(request, response);
21     }
22
23     protected void doPost(HttpServletRequest request, HttpServletResponse response)
24         throws ServletException, IOException {
25         doGet(request, response);
26     }
27
28 }
```

class PessoaServlet

```
30 public void processaRequisicao(HttpServletRequest request, HttpServletResponse response)
31     throws ServletException, IOException {
32
33     String nome = request.getParameter("nome");
34     String sobrenome = request.getParameter("sobrenome");
35     String email = request.getParameter("email");
36
37     Pessoa bean = new Pessoa();
38     bean.setNome(nome);
39     bean.setSobrenome(sobrenome);
40     bean.setEmail(email);
41
42     request.setAttribute("PessoaBean", bean);
43
44     RequestDispatcher view = request.getRequestDispatcher("saudacao.jsp");
45     view.forward(request, response);
46 }
47 }
```

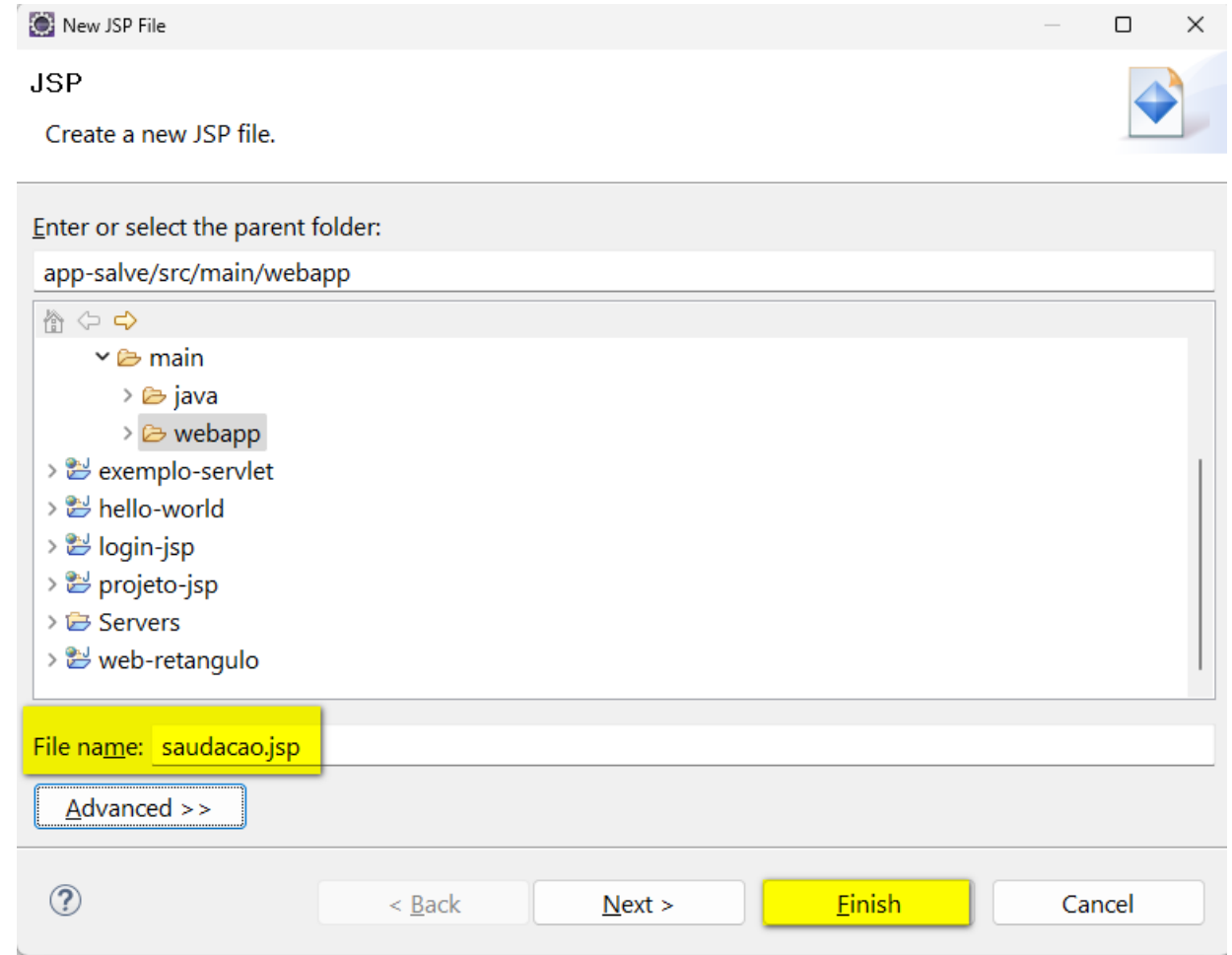
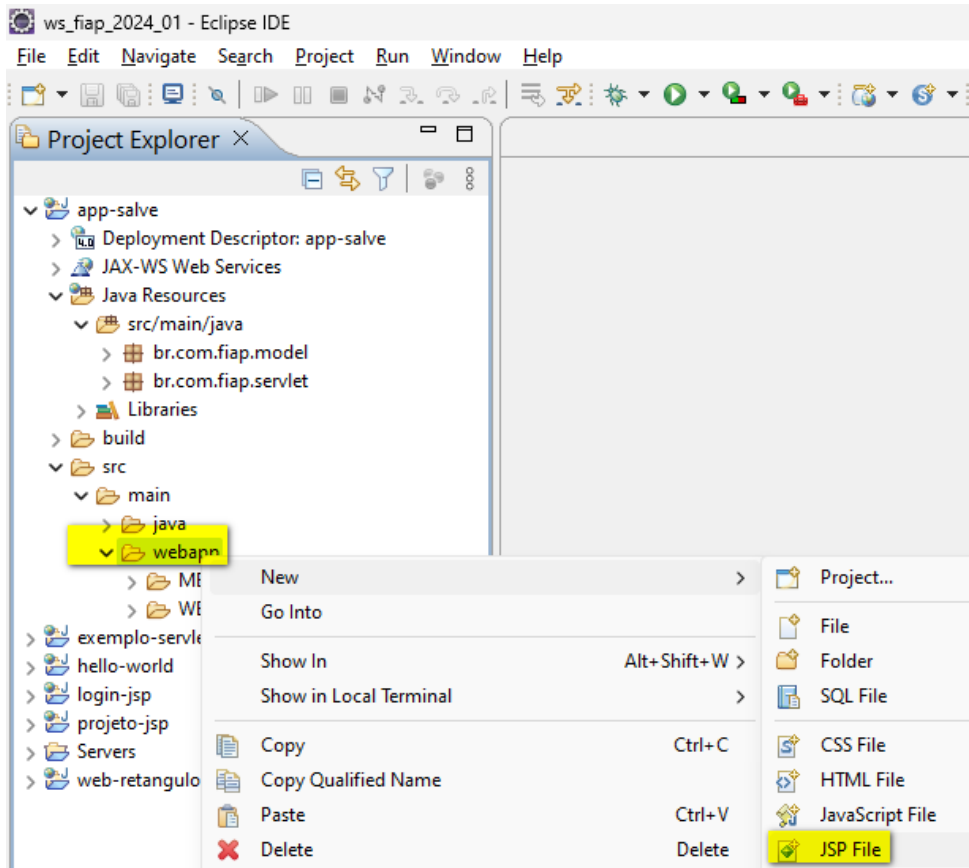
app-salve – index.jsp



app-salve – index.jsp

```
index.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Saudação</title>
8 </head>
9 <body>
10   <h1>Salve! Salve!</h1>
11
12   <form name="form" action="pessoa-servlet" method="post">
13
14       <label for="nome">Nome:</label><br>
15       <input type="text" id="nome" name="nome" placeholder="Nome" required><br> <br>
16
17       <label for="sobrenome">Sobrenome:</label><br>
18       <input type="text" id="sobrenome" name="sobrenome" placeholder="Sobrenome" required><br> <br>
19
20       <label for="email">E-mail:</label><br>
21       <input type="email" id="email" name="email" placeholder="E-mail" required><br> <br>
22
23       <input type="submit" value="Enviar dados"> <br><br>
24       <input type="reset" value="Cancelar">
25
26   </form>
27
28 </body>
29 </html>
```

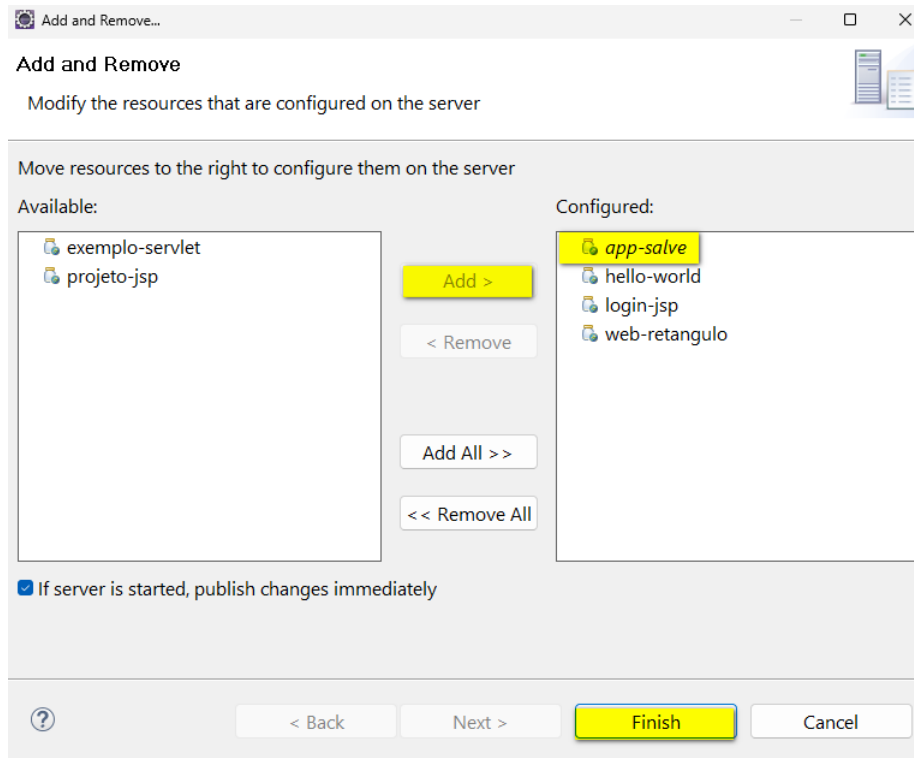
app-salve – saudacao.jsp



app-salve – saudacao.jsp

```
saudacao.jsp x
1 <%@page import="br.com.fiap.model.Pessoa"%>
2 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
3   pageEncoding="ISO-8859-1"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="ISO-8859-1">
8   <title>Saudações</title>
9 </head>
10 <body>
11 <h1>Bem vindo(a)</h1>
12
13 <%
14     Pessoa pessoa = (Pessoa) request.getAttribute("PessoaBean");
15 %>
16
17 <div>
18     <h3>Olá, <%= pessoa.getNomeCompleto() %> </h3>
19     <p>E-mail: <%= pessoa.getEmail() %></p>
20 </div>
21
22 <form action="index.jsp" method="post">
23     <input type="submit" value="Voltar">
24 </form>
25
26 </body>
27 </html>
```


Executando app-salve



app-salve

Saudação

localhost:8080/app-salve/

Universidade API Imagens e Sons Gr...

Salve! Salve!

Nome:

Sobrenome:

E-mail:

Saudação

localhost:8080/app-salve/

Universidade API Imagens e Sons Gr...

Salve! Salve!

Nome:

Sobrenome:

E-mail:

Saudações

localhost:8080/app-salve/pessoa-servlet

Universidade API Imagens e Sons Gr... Ferramentas

Bem vindo(a)

Olá, Cida Castello

E-email: cida@gmail.com

Saudação

localhost:8080/app-salve/index.jsp

Universidade API Imagens e Sons Gr... Ferrame

Salve! Salve!

Nome:

Sobrenome:

E-mail:

Copyright © 2024
Prof^a. Aparecida de Fátima Castello Rosa

Copyright © 2020
Prof^o. Pedro Ivo Correia e Prof^o. Lucas Furlaneto

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).