



REGISTRO ELETTRONICO

Macaluso Giulio - Smal Philippe

Indice

- 1 Installazione IDE
- 2 Pagina JSP e Java
- 3 Problemi riscontrati



01

Installazione IDE

Eclipse -
Tomcat

Installazione IDE



Per sviluppare un progetto web, c'è bisogno di un software adeguato. Noi abbiamo usato eclipse, un ambiente di sviluppo integrato utilizzato principalmente per lo sviluppo di java.

Eclipse - installazione

Andare nel sito di Eclipse e scaricare la versione “Eclipse IDE for Enterprise Java and Web Developers”.

<https://www.eclipse.org/downloads/packages/release/2021-12/r>

The Eclipse Installer 2024-03 R now includes a JRE for macOS, Windows and Linux.

Try the Eclipse **Installer** 2024-03 R

The easiest way to install and update your Eclipse Development Environment.

[Find out more](#)

📥 203,904 Installer Downloads

📦 185,876 Package Downloads and Updates


Download

macOS x86_64 | [AArch64](#)

Windows x86_64

Linux x86_64 | [AArch64](#)

Eclipse IDE 2021-12 R Packages




Eclipse IDE for Java Developers

306 MB | 482,694 DOWNLOADS

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration

Windows x86_64
macOS x86_64 | [AArch64](#)
Linux x86_64 | [AArch64](#)



Eclipse IDE for Enterprise Java and Web Developers

509 MB | 447,686 DOWNLOADS

Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, Yaml, Markdown, Web Services, JPA and Data Tools, Maven and Gradle, Git, and more.

Windows x86_64
macOS x86_64 | [AArch64](#)
Linux x86_64 | [AArch64](#)

[Click here to file a bug against Eclipse Web Tools Platform.](#)

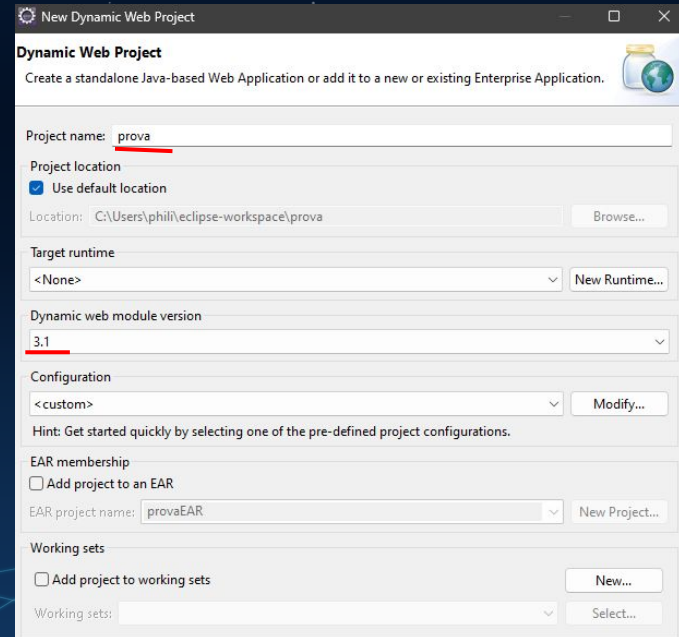
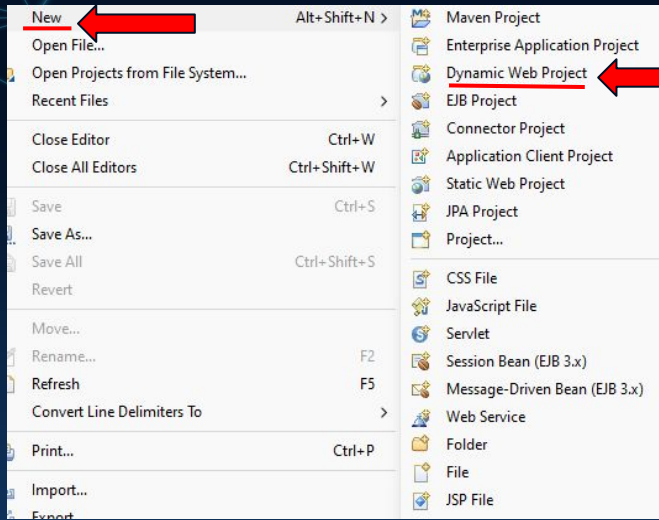
[Click here to file a bug against Eclipse Platform.](#)

[Click here to file a bug against Maven integration for web projects.](#)

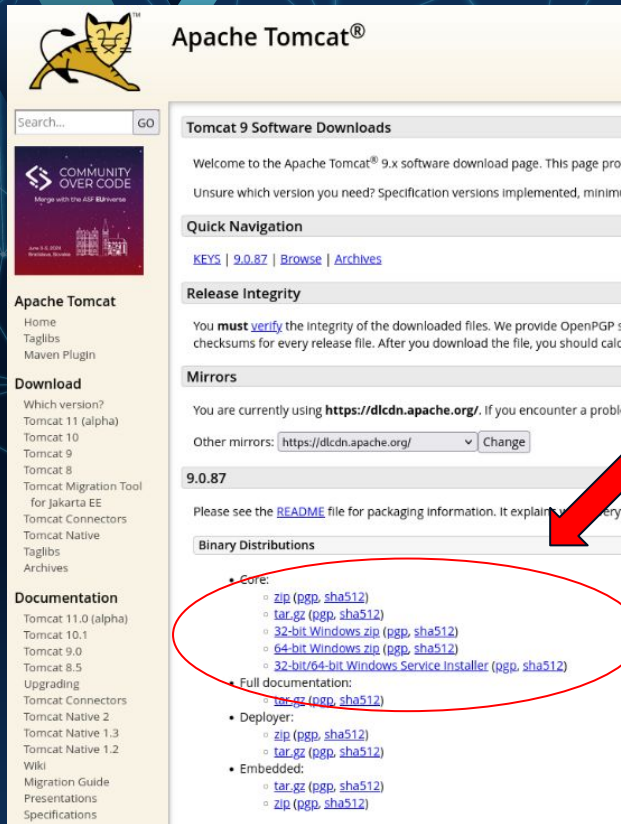
[Click here to report an issue against Eclipse Wild Web Developer \(incubating\).](#)

Eclipse - Creazione

Per riuscire ad utilizzare Eclipse è necessario disporre le cartelle e i file in un ordine preciso.
Per creare un progetto bisogna cliccare su “New” e su “Dynamic Web Project”.



Tomcat - installazione



Apache Tomcat®

Search... GO

Tomcat 9 Software Downloads

Welcome to the Apache Tomcat® 9.x software download page. This page provides information on how to obtain the software, and how to verify the integrity of the downloaded files.

Unsure which version you need? Specification versions implemented, minimal requirements, and more.

Quick Navigation

[KEYS](#) | [9.0.87](#) | [Browse](#) | [Archives](#)

Release Integrity

You **must** [verify](#) the integrity of the downloaded files. We provide OpenPGP signatures and checksums for every release file. After you download the file, you should calculate the checksum and compare it to the one provided.

Mirrors

You are currently using <https://dlcdn.apache.org/>. If you encounter a problem, you can switch to another mirror.

Other mirrors: <https://dlcdn.apache.org/>

9.0.87

Please see the [README](#) file for packaging information. It explains the naming convention for the files, and provides links to the source code and documentation.

Binary Distributions

- **Core:**
 - [zip \(pgp, sha512\)](#)
 - [tar.gz \(pgp, sha512\)](#)
 - [32-bit Windows.zip \(pgp, sha512\)](#)
 - [64-bit Windows.zip \(pgp, sha512\)](#)
 - [32-bit/64-bit Windows Service Installer \(pgp, sha512\)](#)
- **Full documentation:**
 - [tar.gz \(pgp, sha512\)](#)
- **Deployer:**
 - [zip \(pgp, sha512\)](#)
 - [tar.gz \(pgp, sha512\)](#)
- **Embedded:**
 - [tar.gz \(pgp, sha512\)](#)
 - [zip \(pgp, sha512\)](#)

Per eseguire il nostro programma abbiamo bisogno di Tomcat, un webserver.
Per scaricarlo andare nel sito:

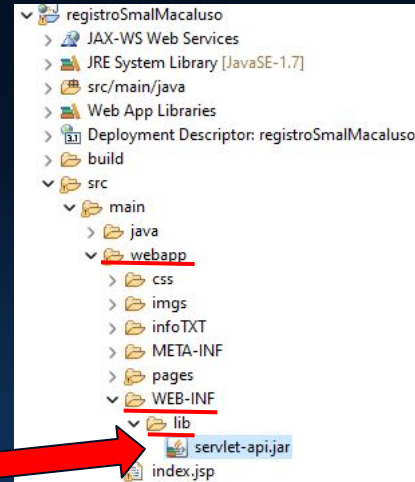
<https://tomcat.apache.org/download-90.cgi>

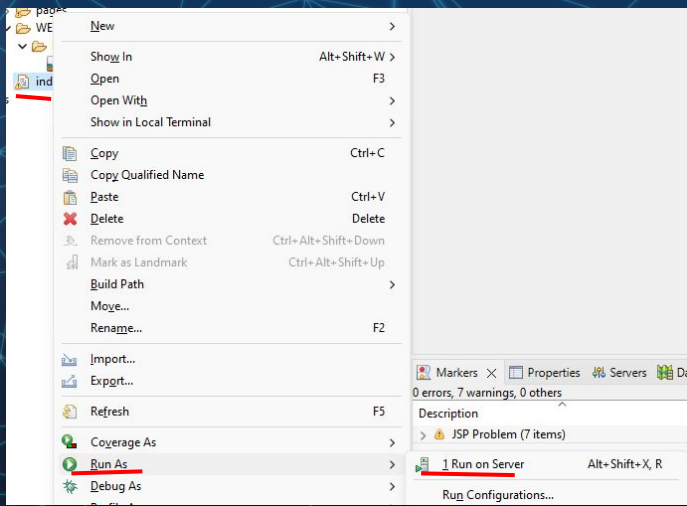
Tomcat - configurazione

Successivamente dobbiamo collegare il webserver al nostro progetto.

Unzip della cartella e dobbiamo inserire il file “servlet-api.jar” della cartella lib di tomcat nella cartella lib della nostra webapp.

websocket-api.jar	23/02/2024 16:37	Executable Jar File	40 KB
tomcat-i18n-fr.jar	23/02/2024 16:37	Executable Jar File	169 KB
tomcat-i18n-ja.jar	23/02/2024 16:37	Executable Jar File	192 KB
tomcat-i18n-ko.jar	23/02/2024 16:37	Executable Jar File	189 KB
tomcat-i18n-pt-BR.jar	23/02/2024 16:37	Executable Jar File	53 KB
tomcat-i18n-ru.jar	23/02/2024 16:37	Executable Jar File	65 KB
tomcat-i18n-zh-CN.jar	23/02/2024 16:37	Executable Jar File	172 KB
tomcat-jdbc.jar	23/02/2024 16:37	Executable Jar File	149 KB
tomcat-jni.jar	23/02/2024 16:37	Executable Jar File	38 KB
tomcat-util.jar	23/02/2024 16:37	Executable Jar File	223 KB
tomcat-util-scan.jar	23/02/2024 16:37	Executable Jar File	223 KB
tomcat-websocket.jar	23/02/2024 16:37	Executable Jar File	243 KB
catalina.jar	23/02/2024 16:37	Executable Jar File	1.751 KB
catalina-storeconfig.jar	23/02/2024 16:37	Executable Jar File	78 KB
catalina-tribes.jar	23/02/2024 16:37	Executable Jar File	333 KB
ecj-4.20.jar	23/02/2024 16:37	Executable Jar File	3.061 KB
el-api.jar	23/02/2024 16:37	Executable Jar File	88 KB
jasper.jar	23/02/2024 16:37	Executable Jar File	563 KB
jasper-el.jar	23/02/2024 16:37	Executable Jar File	170 KB
jaspic-api.jar	23/02/2024 16:37	Executable Jar File	27 KB
jsp-api.jar	23/02/2024 16:37	Executable Jar File	73 KB
servlet-api.jar	23/02/2024 16:37	Executable Jar File	279 KB

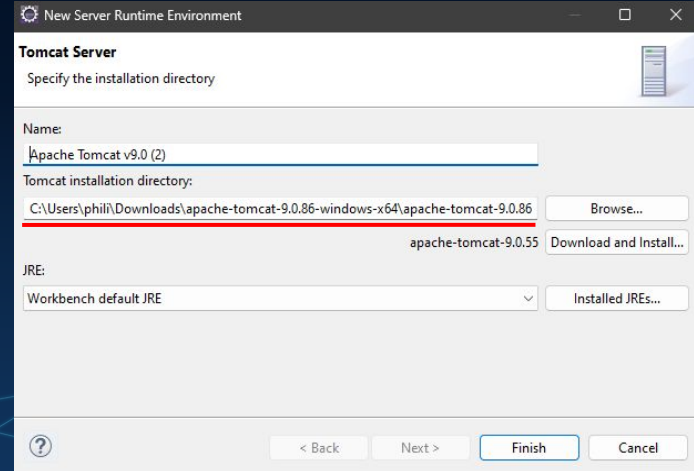
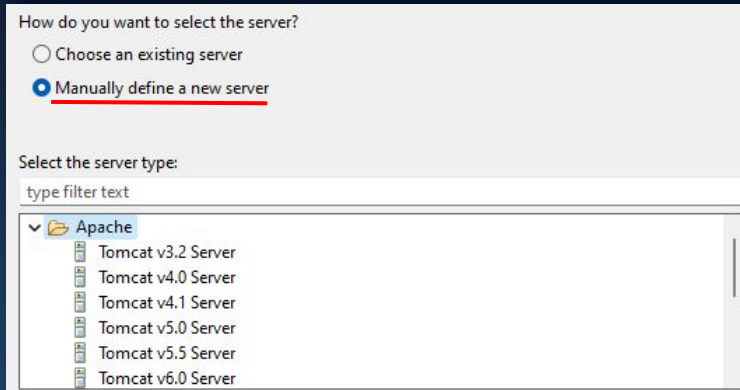




Tomcat - configurazione

Procediamo a collegare la cartella di Tomcat al nostro progetto. Per eseguire seleziona: 'Run As' > 'Run on Server'.

Successivamente, scegli la versione di Tomcat che abbiamo installato e seleziona la cartella.



A cluster of three-dimensional cubes and a network of thin, light-blue lines forming a complex web in the upper-left corner.

02

Pagina JSP

A cluster of three-dimensional cubes and a network of thin, light-blue lines forming a complex web in the lower-right corner.

Pagina JSP - struttura

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<%@ page import="com.tools_user.Docente" %>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Registro | Home - docente</title>
<link rel="stylesheet" href="/registroSmaMacaluso/css/style_docente.css">
</head>
<body>
<%
Docente docente = (Docente)session.getAttribute("docenteTake");
%>
<form class="card">
<h2 id="title-card">Home - Docente</h2>
<div class="user-card">
<div class="info-div">
<h3 id="nome_docente"><%=docente.getNome() + " " + docente.getCognome() %></h3>
<ul>
<%
for(String materie : docente.getMaterie()) {
%>
<li><%=materie %></li>
<%
}
%>
</ul>
</div>
<div class="input-inserimento-classe">
<h3 id="nome_docente">Voto e Appello</h3>
<select class="drop-down-menu" name="inputSezione">
<option value="" disabled selected hidden>Classe</option>
<option value="3 Informatica">3 Informatica</option>
<option value="4 Informatica">4 Informatica</option>
<option value="5 Informatica">5 Informatica</option>
</select>
<div class="scelta">
<button class="button-scelta" name="inserisciVoto">Inserisci voto</button>
<button class="button-scelta" name="appello">Fai l'appello</button>
</div>
</div>
</form>
<%
if (request.getParameter("inserisciVoto") != null) {
session.setAttribute("classeTake", request.getParameter("inputSezione"));
response.sendRedirect("votoDocente.jsp");
}
%>
</body>
</html>
```

Questo è la struttura di una pagina JSP.

La prima riga serve per il collegamento al server, successivamente vengono importate le classi che si utilizzano. Le righe seguenti sono l'HTML e infine c'è la parte di script.

Pagina JAVA - classi user

Abbiamo creato la classe User, con attributi nome, cognome, nome utente e password. Le classi Studente e Insegnante estendono la classe User e hanno metodi e attributi in più caratteristici dell'utente. Abbiamo anche creato le classi Voto e Classe.

```
package com.tools_user;

public class User {
    private String nome, cognome, utente, password;

    public User(String nome, String cognome, String utente, String password) {
        this.nome = nome;
        this.cognome = cognome;
        this.utente = utente;
        this.password = password;
    }

    public User(String nome, String cognome, String password) {}
        this.nome = nome;
        this.cognome = cognome;
        this.utente = nome.toLowerCase() + "." + cognome.toLowerCase();
        this.password = password;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public void setCognome(String cognome) {
        this.cognome = cognome;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public void setUtente(String utente) {
        this.utente = utente;
    }

    public String getNome() {
        return nome;
    }

    public String getCognome() {
        return cognome;
    }

    public String getPassword() {
        return password;
    }

    public String getUtente() {
        return utente;
    }

    public String toString() {
        return utente + " " + password;
    }
}
```

Pagina JAVA - Gestione file

```
package com.tools_file;

import java.io.FileReader;

public class Reader {
    private FileReader file;
    private Scanner in;

    public Reader(String pathFile) {
        try {
            file = new FileReader(pathFile);
            in = new Scanner(file);
        } catch (IOException e) {
            System.out.println("Errore: Impossibile aprire il file");
            System.exit(0);
        }
    }

    public String getLine() {
        return in.nextLine();
    }

    public boolean isFinished() {
        return in.hasNext();
    }

    public void closeFile() {
        try {
            in.close();
            file.close();
        } catch (IOException e) {
            System.out.println("Errore: Impossibile chiudere il file");
            System.exit(0);
        }
    }
}
```

Abbiamo salvato tutte le informazioni su dei file, abbiamo quindi creato le classi Reader e Writer.

La classe Reader ha i metodi per aprire il file, leggere una riga, controllare la fine del file e chiudere il file.


```

package com.tools_file;

import java.io.BufferedWriter;

public class Writer {
    private FileWriter file;
    private BufferedWriter buffer;
    private PrintWriter print;
    private String pathFile;

    public Writer(String pathFile, boolean append) {
        try {
            file = new FileWriter(pathFile, append);
            buffer = new BufferedWriter(file);
            print = new PrintWriter(buffer);
            this.pathFile = pathFile;
        } catch (IOException e) {
            System.out.println("Errore: Impossibile aprire il file");
            System.exit(0);
        }
    }

    public Writer(String pathFile) {
        try {
            file = new FileWriter(pathFile);
            this.pathFile = pathFile;
        } catch (IOException e) {
            System.out.println("Errore: Impossibile aprire il file");
            System.exit(0);
        }
    }

    void replaceAll(LinkedList<String> lslinee) {
        Writer wf = new Writer(pathFile);
        Iterator<String> it = lslinee.iterator();
        while (it.hasNext()) wf.writeLine(it.next()+"\n");
        wf.closeFile();
    }

    public void writeLine(String line) {
        try {
            file.write(line);
        } catch (IOException e) {
            System.out.println("Errore: Impossibile scrivere sul file");
            System.exit(0);
        }
    }

    public String getPathFile() {
        return pathFile;
    }

    public void addLine(String line) {
        print.print("\n"+line);
    }

    private boolean isNull(Object obj) {
        return obj != null;
    }
}

```

Pagina JAVA - Gestione file

La classe Writer ha due costruttori, il primo serve ad aggiungere qualcosa all'interno del file, il secondo serve a sovrascrivere il file. Ha inoltre il metodo per sovrascrivere, per aggiungere una linea e per creare una nuova linea.

Pagina JAVA - Gestione file

Infine la classe `WriterVoto` serve ad aggiungere un voto alla lista dei voti, attraverso i metodi `addVoto` ed `editLine`.

```
package com.tools_file;

import java.util.LinkedList;

public class WriterVoto extends Writer{
    public WriterVoto(String pathFile, boolean append) {
        super(pathFile, append);
    }

    public void addVoto(int index, Voto voto) {
        Reader read = new Reader(getPathFile());
        LinkedList<String> lsStr = new LinkedList<String>();

        while(read.isFinished()) {
            lsStr.add(read.getLine());
            System.out.println(lsStr.getLast());
        }
        read.closeFile();

        lsStr.set(index, editLine(lsStr.get(index),
            voto.getMateria() + "-" + voto));

        replceAll(lsStr);
    }

    public String editLine(String strVoti, String voto) {
        return voto + " " + strVoti;
    }
}
```



Problemi riscontrati

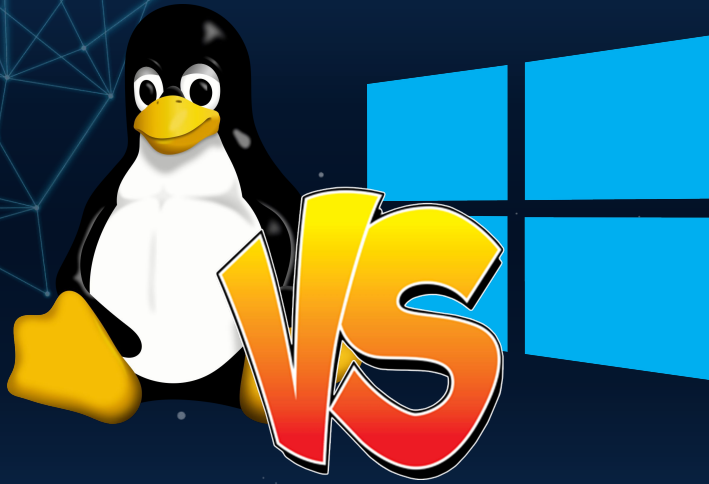
Problemi riscontrati

Eclipse è un ottimo strumento con cui sviluppare delle web app, tuttavia inizialmente volevamo utilizzare netbeans.

Questo IDE è però designato più per lo sviluppo di Desktop app che per le web app, la realizzazione del progetto risultava pertanto più complicata.



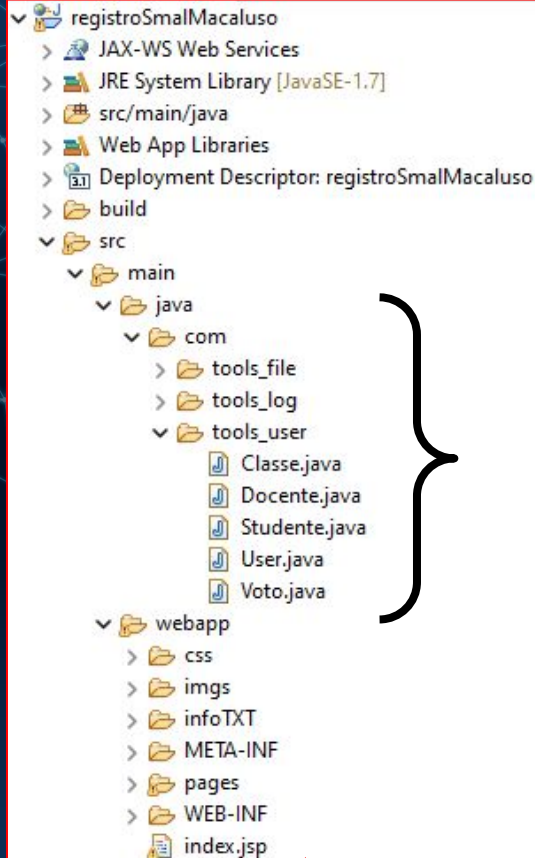
Problemi riscontrati



Inizialmente volevamo sviluppare il nostro progetto su Linux, installato su una macchina virtuale.

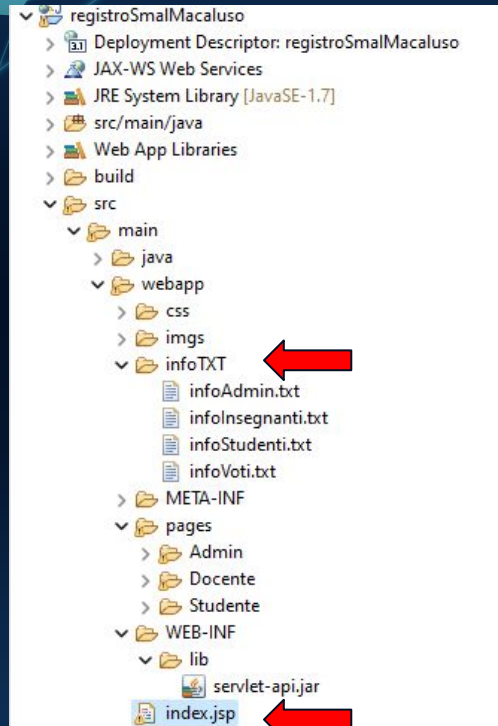
Tuttavia abbiamo scoperto che quest'ultima blocca le porte dalla 8000 in su, quindi abbiamo dovuto spostarci su Windows.

Problemi riscontrati



Abbiamo avuto qualche problema nella gestione delle cartelle e file, specialmente nella comunicazione tra la parte front-end e back-end. Abbiamo risolto inserendo la parte di back-end come source folder.

Problemi riscontrati



Un altro problema è stata la lettura dei file di testo. Su Windows 11, quando creiamo un nuovo progetto, viene copiato interamente in un'altra cartella chiamata 'eclipse-workspace'. Questa cartella è dove viene eseguito il progetto ed è dove sono contenuti i file di testo. Abbiamo risolto questo problema inserendo nel Writer e nel Reader questo metodo che ci restituisce il reale percorso della cartella che contiene i file di testo:

```
request.getServletContext().getRealPath("/NomeCartella")
```

The background is a dark blue gradient. It features several abstract geometric elements: a complex wireframe of interconnected triangles in the top-left and bottom-right corners, and several 3D cubes of varying sizes and orientations in the top-right and bottom-left corners. The cubes are rendered with a blue-to-white gradient, giving them a three-dimensional appearance. A large, white-outlined rectangle with slightly clipped corners is centered on the slide.

FINE

Macaluso Giulio - Smal Philippe