Cento Universitário UNA

Sistemas de Informação

Tecnologias Emergentes

Práticas de Laboratório

Wesley Dias Maciel

2019/02

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

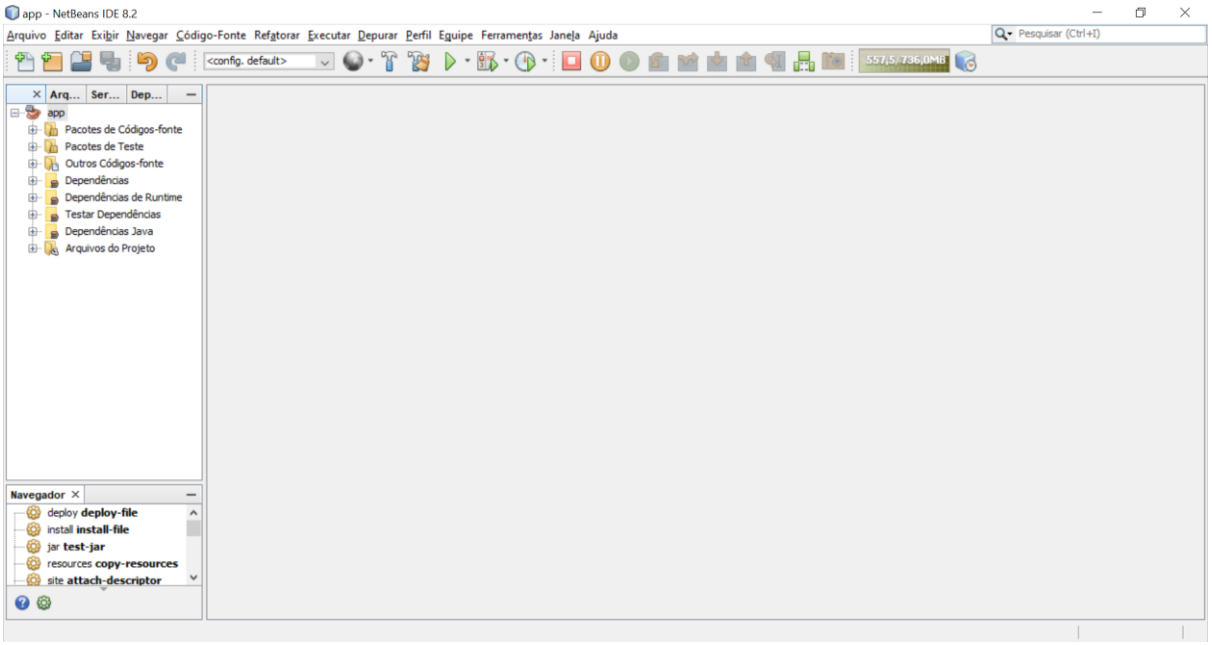
Spring Boot

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

Prática 01

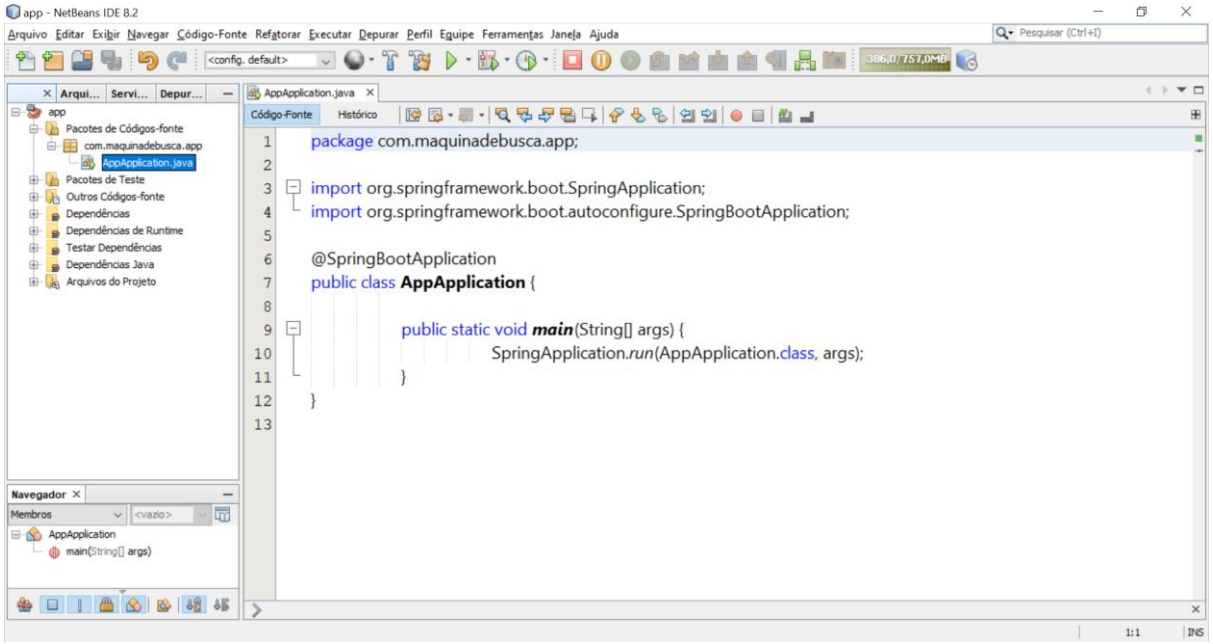
Controlador

1) Abra o projeto gerado pelo Spring Initializr no NetBeans, Eclipse ou Spring Tool Suite (STS).

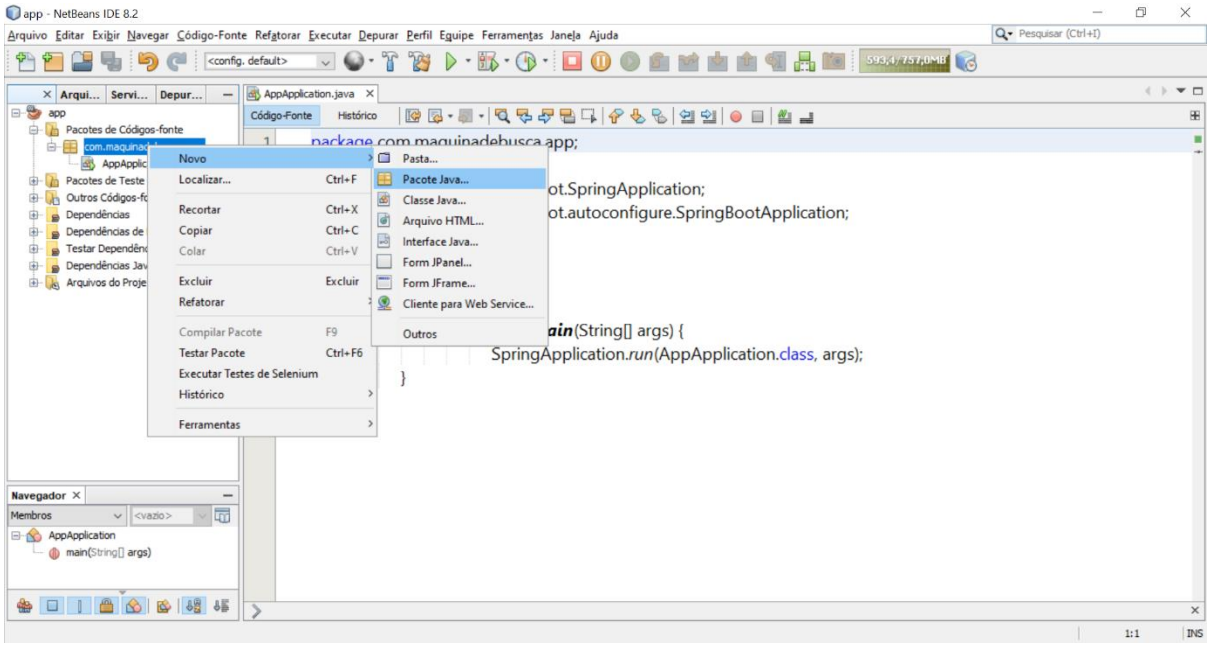


No projeto aberto, clique em Pacotes de Códigos-fonte. Em seguida, clique no pacote com.maquinadebusca.app. Clique na classe AppApplication e observe o conteúdo dessa classe na janela para visualização. Essa é a classe principal do projeto. Ela é a responsável por iniciar a execução do projeto.

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

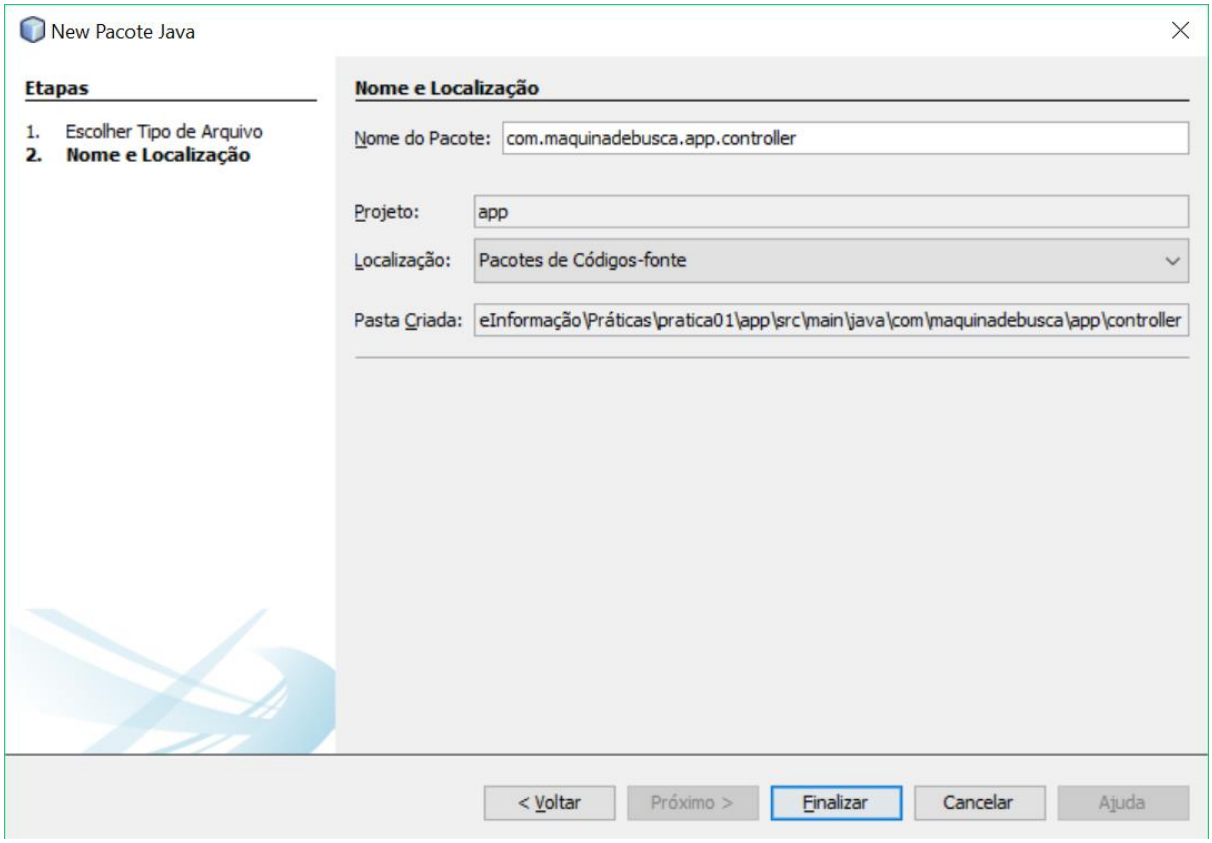


Clique com o botão direito do mouse sobre o pacote com.maquinadebusca.app. Em seguida, clique em Novo. Depois, clique em Pacote Java.

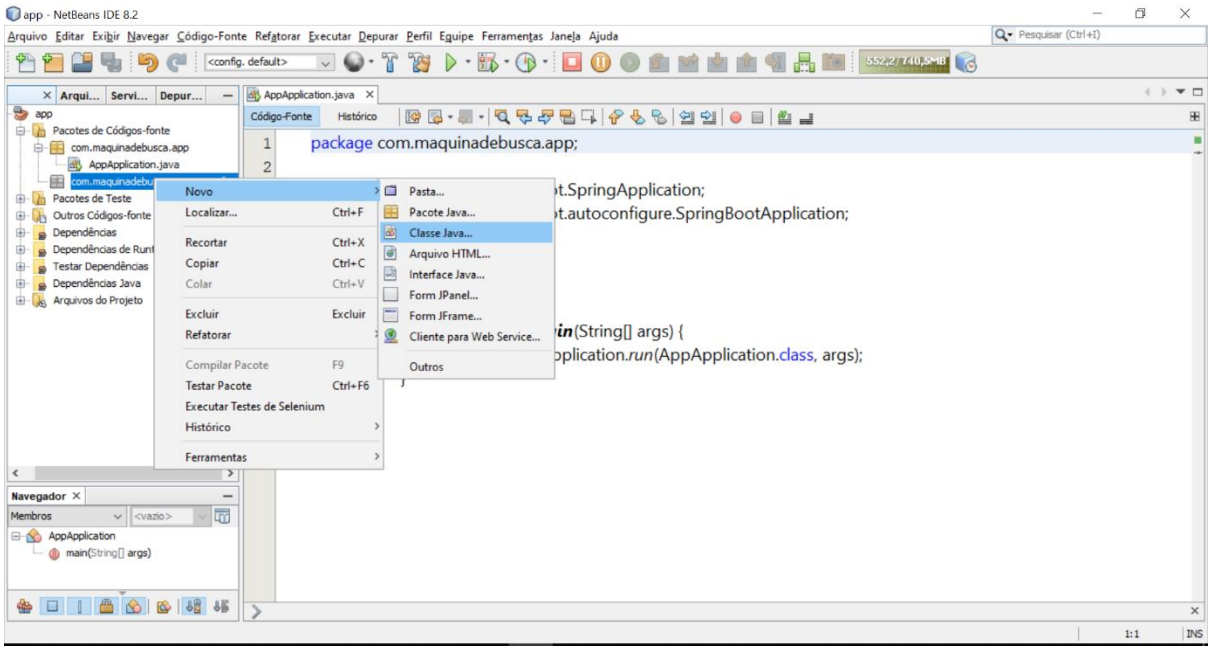


No campo Nome do Pacote da janela que se abre, informe o nome controller, conforme apresentado na figura abaixo (com.maquinadebusca.app.controller). Em seguida, clique em Finalizar.

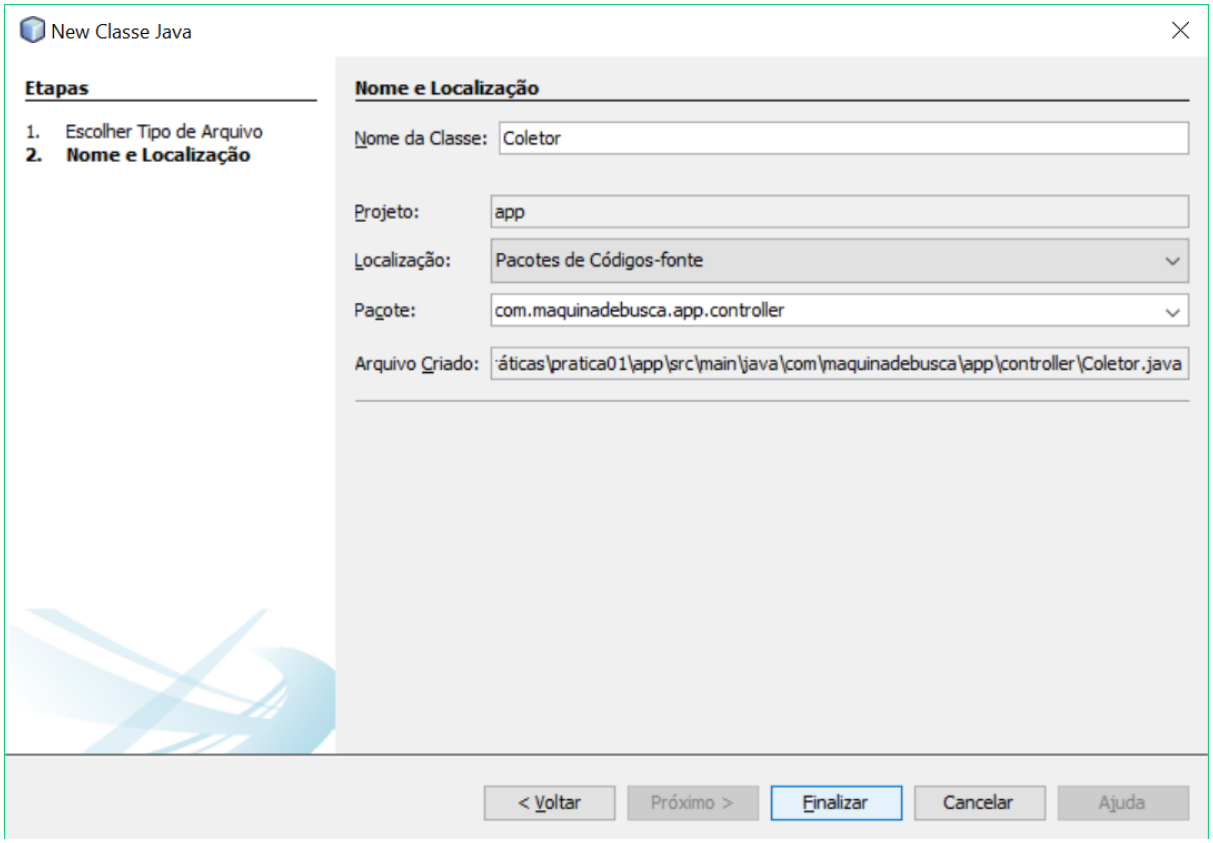
Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02



Clique com o botão direito do mouse sobre o pacote com.maquinadebusca.app.controller. Em seguida, clique em Novo. Depois, clique em Classe Java.



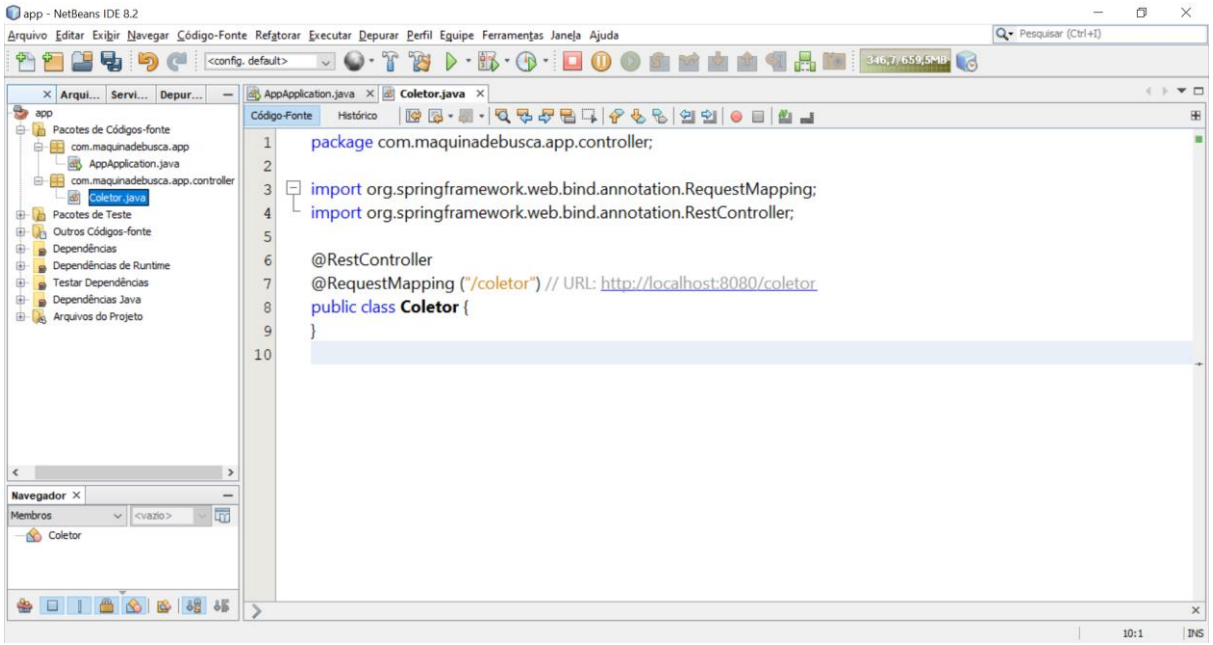
No campo Nome da Classe da janela que se abre, informe o nome Coletor, conforme apresentado na figura abaixo. Em seguida, clique em Finalizar.



Anotar a classe “Coletor", conforme apresentado abaixo:

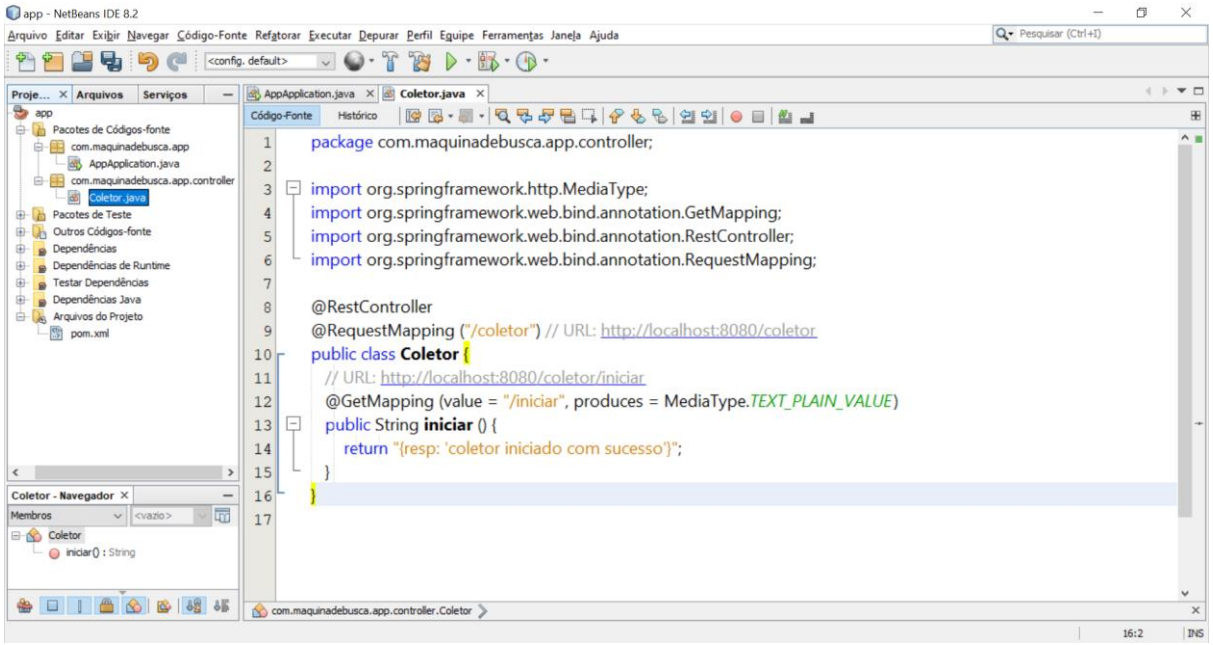
```
package com.maquinadebusca.app.controller;

import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping ("/coletor") // URL: http://localhost:8080/coletor
public class Coletor {
   //...
}
```

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

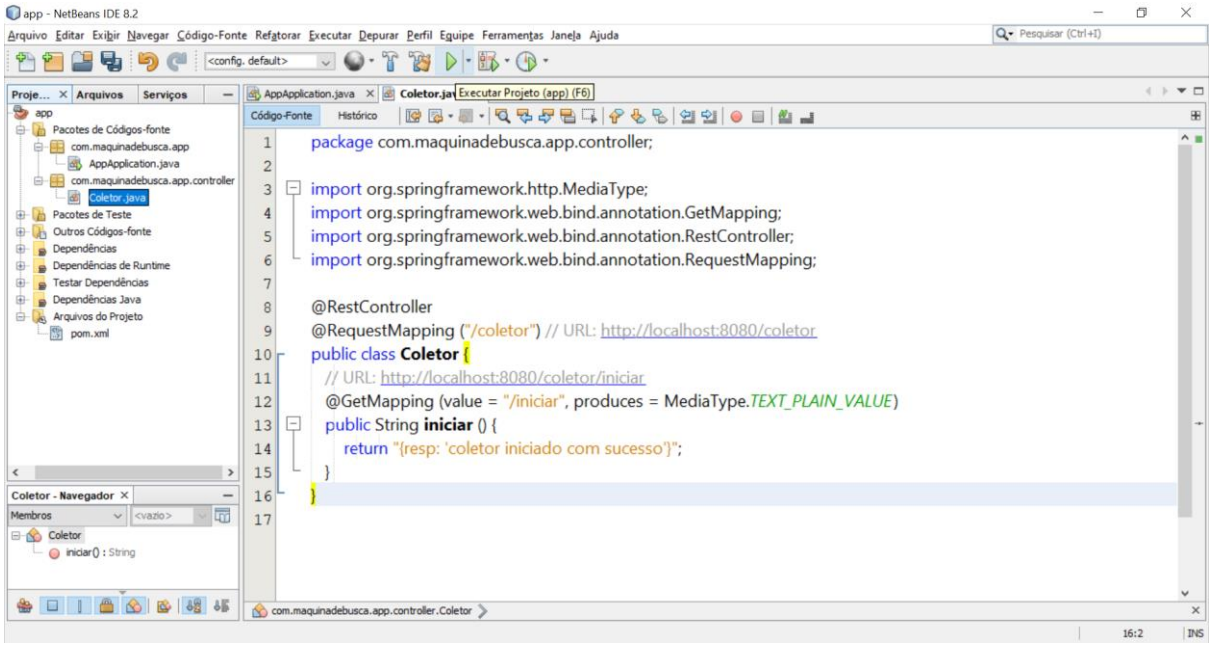


Na classe “Coletor“, criar o método “iniciar ()“, conforme apresentado abaixo.

```
package com.maquinadebusca.app.controller;

import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.bind.annotation.RequestMapping;

@RestController
@RequestMapping ("/coletor") // URL: http://localhost:8080/coletor
public class Coletor {
  // URL: http://localhost:8080/coletor/iniciar
  @GetMapping (value = "/iniciar", produces = MediaType.TEXT_PLAIN_VALUE)
  public String iniciar () {
     return "{resp: 'coletor iniciado com sucesso'}";
  }
}
```

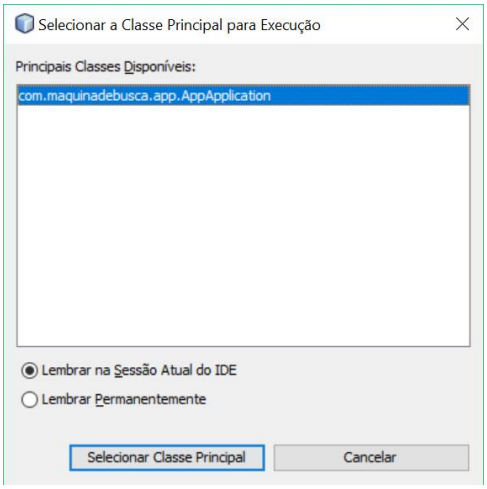


Em seguida, execute o projeto, clicando no ícone para execução ou teclando F6.

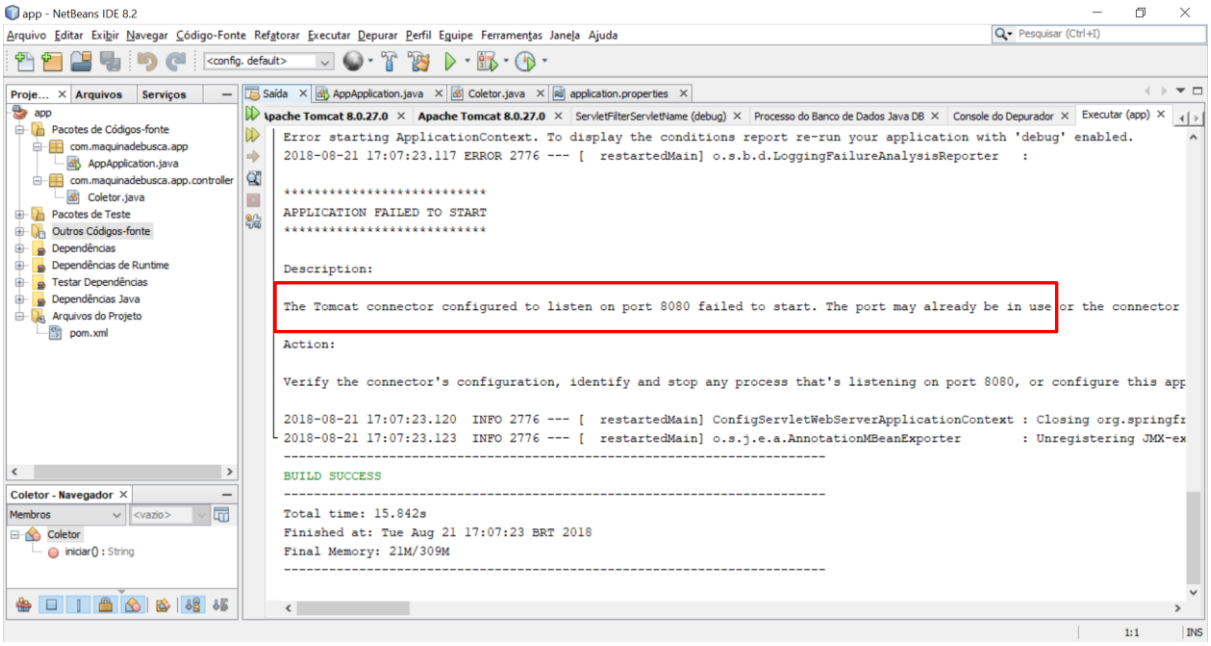


Na janela que se abre, clique na classe principal do projeto (com.maquinadebusca.app.AppApplication). Em seguida, clique em Selecionar Classe Principal.

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

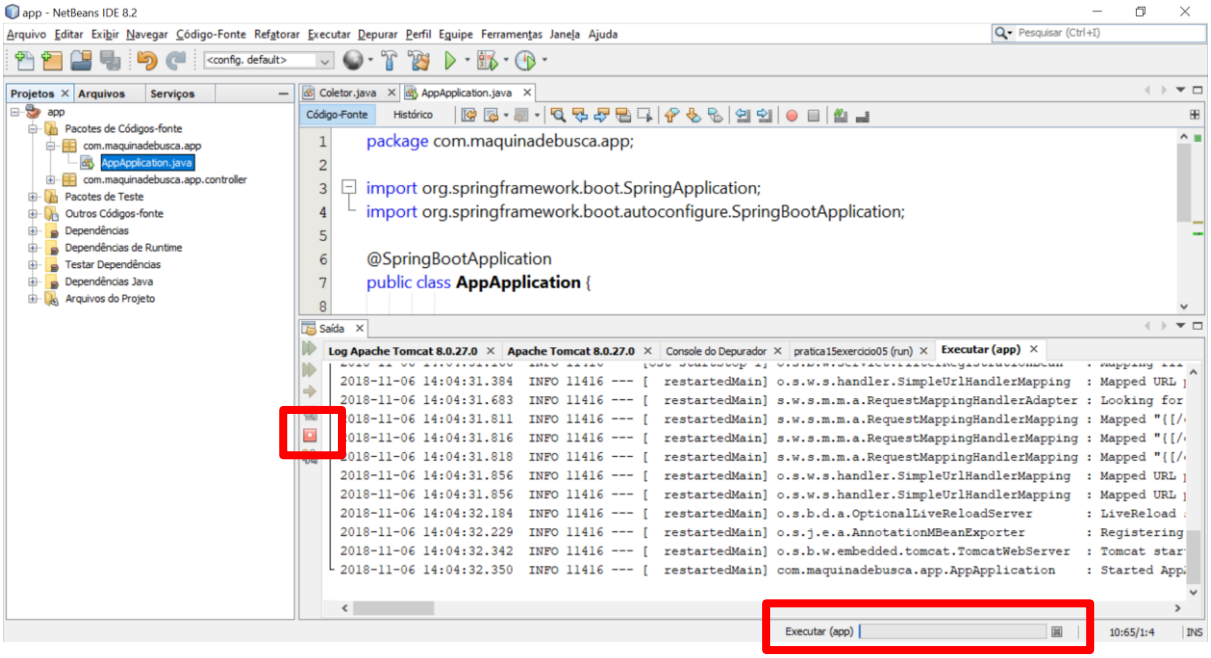


Observe a saída do Spring Boot, verificando as mensagens apresentadas. Caso haja algum conflito de porta, como apresentado na figura abaixo, será necessário finalizar execuções prévias do projeto ou alterar a porta.

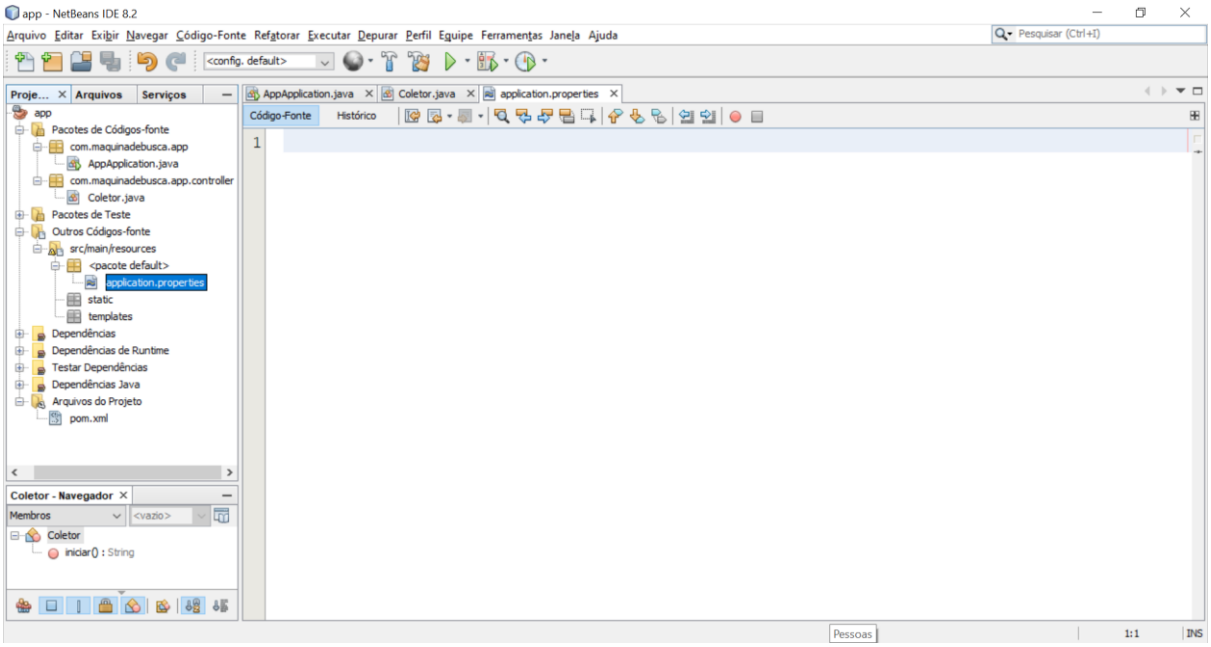


Para finalizar execuções prévias do projeto, basta clicar em um dos botões assinalados na figura abaixo.

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02



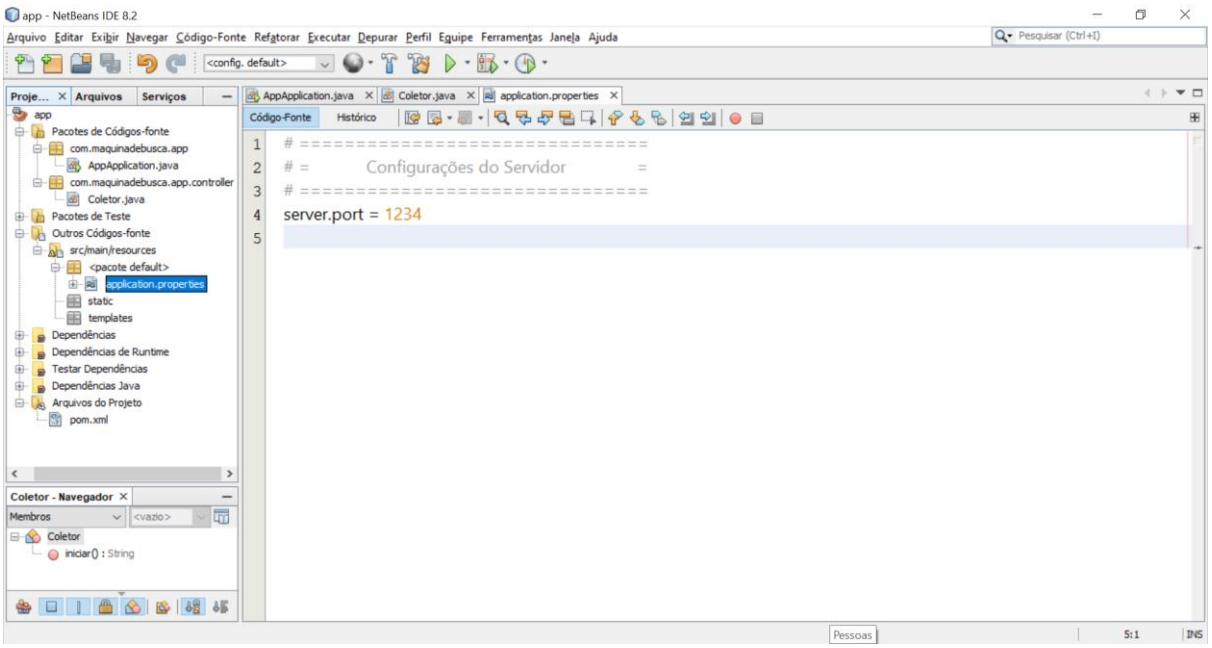
Para alterar a porta do projeto, clique em Outros Códigos-fonte. Depois, clique em src/main/resources. Em seguida, clique em <pacote default>. Por fim, abra o arquivo application.properties.



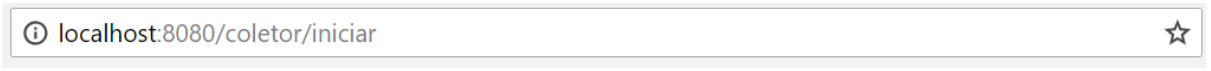
Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

No arquivo application.properties, informe:

```
# ================================
# =        Configurações do Servidor       =
# ================================
server.port = 1234
```



Depois que o servidor for iniciado, abra o navegador Google Chrome. Na barra de endereço, informe http://localhost:8080/coletor/iniciar ou http://localhost:1234/coletor/iniciar, caso tenha sido necessário alterar a porta de acesso à aplicação. Observe a saída no navegador.



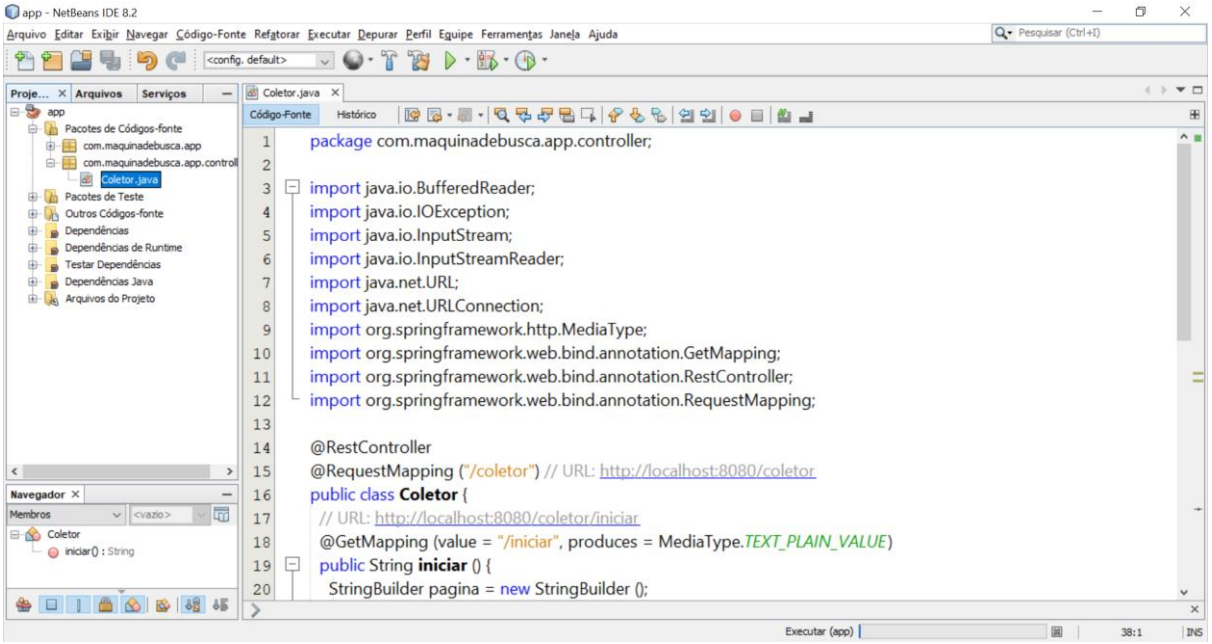

```
{resp: 'coletor iniciado com sucesso'}
```

ou

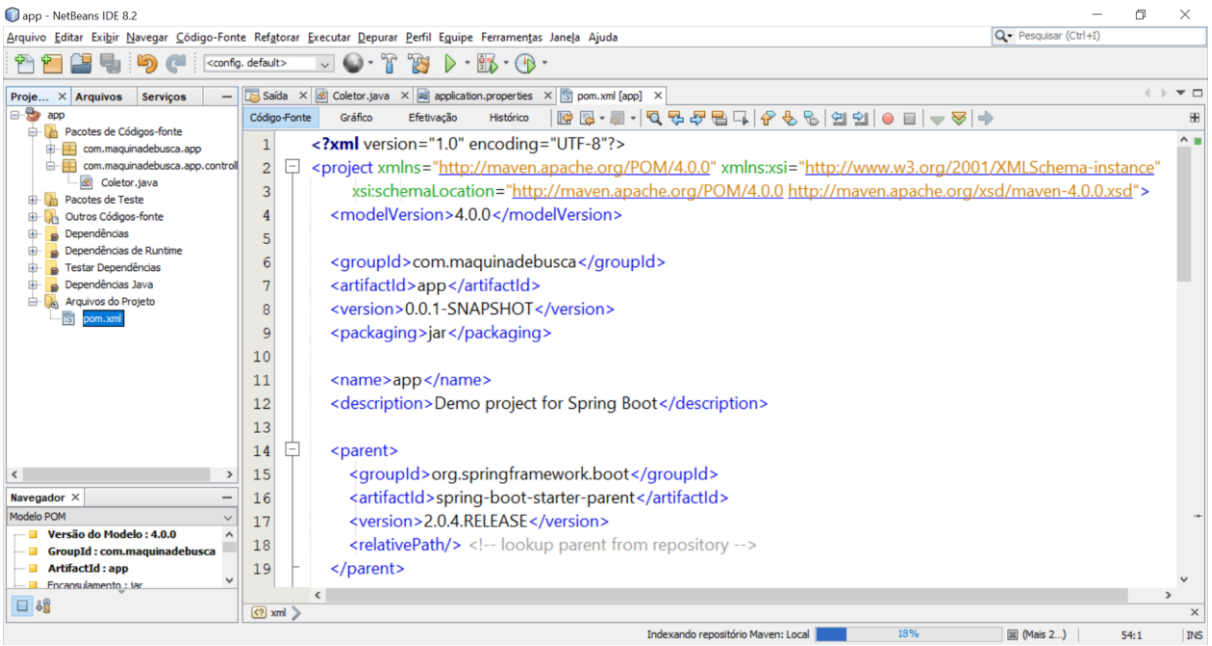

```
{resp: 'coletor iniciado com sucesso'}
```

2) Altere o método iniciar (), conforme indicado abaixo e execute o programa novamente.

```
package com.maquinadebusca.app.controller;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.bind.annotation.RequestMapping;

@RestController
@RequestMapping ("/coletor") // URL: http://localhost:8080/coletor
public class Coletor {
 // URL: http://localhost:8080/coletor/iniciar
 @GetMapping (value = "/iniciar", produces = MediaType.TEXT_PLAIN_VALUE)
 public String iniciar () {
  StringBuilder pagina = new StringBuilder ();
  try {
   URL url = new URL
("http://journals.ecs.soton.ac.uk/java/tutorial/networking/urls/readingWriting.html");
   URLConnection url_connection = url.openConnection ();
   InputStream is = url_connection.getInputStream ();
   InputStreamReader reader = new InputStreamReader (is);
   BufferedReader buffer = new BufferedReader (reader);

   String linha;
   while ((linha = buffer.readLine ()) != null) {
    pagina.append (linha);
   }
  } catch (IOException e) {
   pagina.append ("Erro: não foi possível coletar a página.");
  }
  return pagina.toString ();
 }
}
```



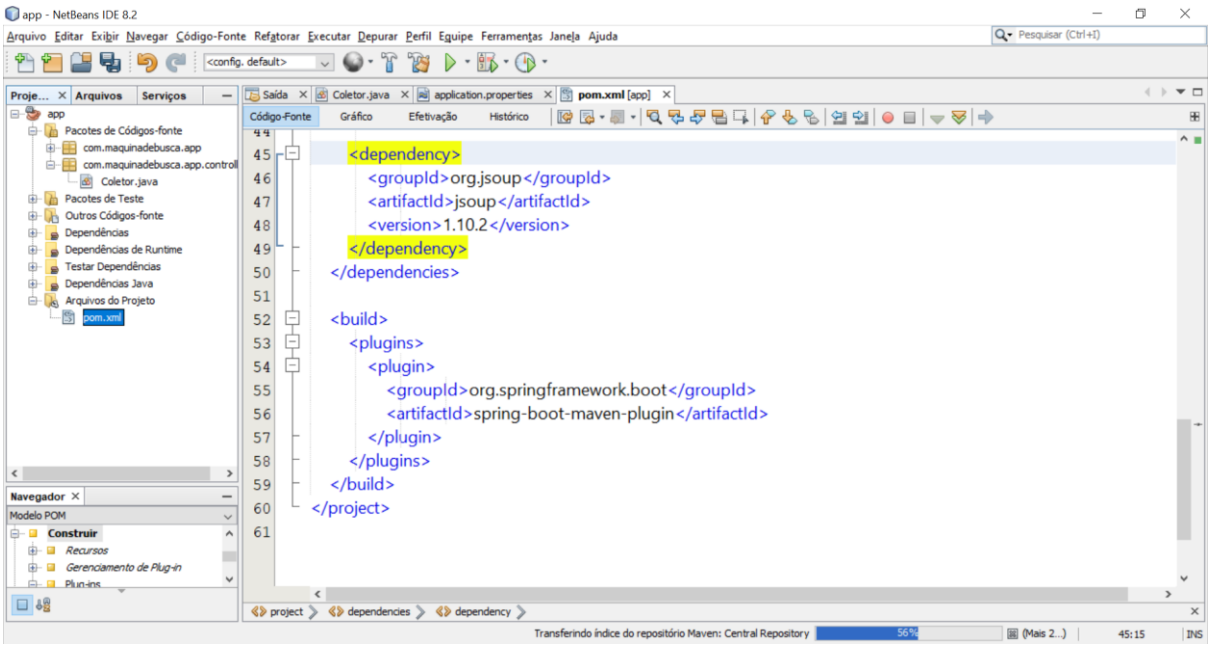
3) Clique na pasta Arquivos do Projeto. Em seguida, abra o arquivo pom.xml. Esse arquivo lista todas as dependências do projeto.



Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

Ao final do elemento <dependencies> do arquivo pom.xml, inclua uma nova dependência para o Jsoup (https://jsoup.org/). O Jsoup é uma biblioteca que auxilia o trabalho com arquivos HTML.

```
<dependency>
   <groupId>org.jsoup</groupId>
   <artifactId>jsoup</artifactId>
   <version>1.10.2</version>
</dependency>
```



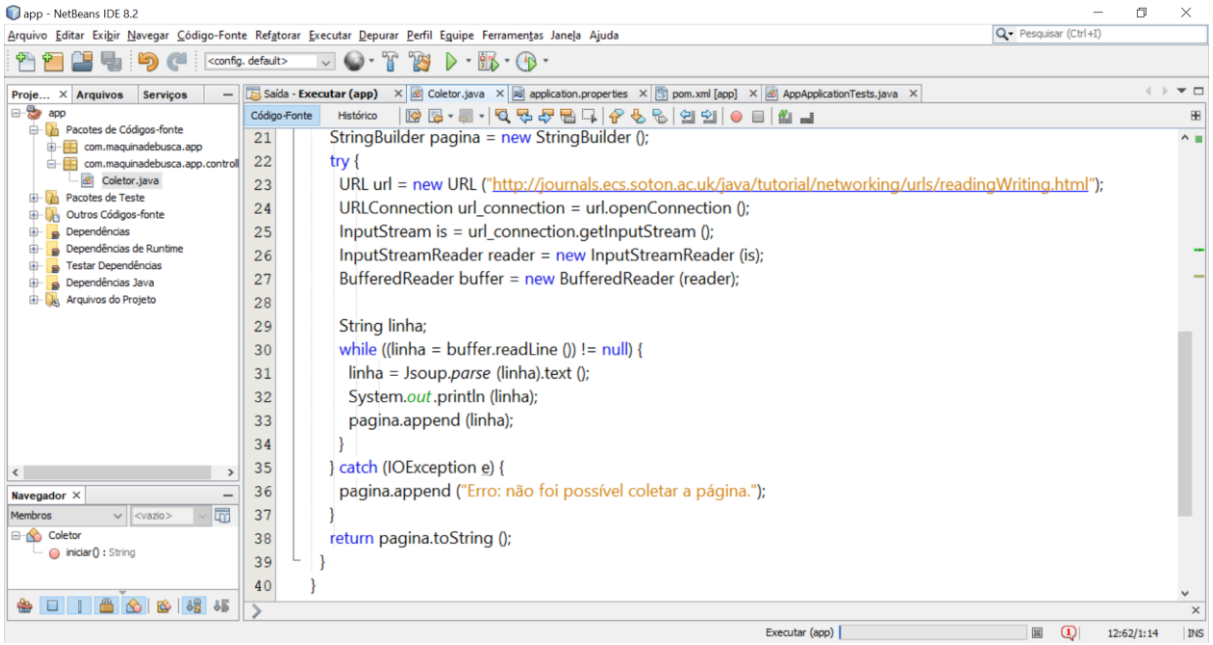
Em seguida, altere a classe Coletor conforme apresentado abaixo, para que ele exclua os elementos HTML, tags, da página coletada.

```
package com.maquinadebusca.app.controller;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import org.jsoup.Jsoup;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.bind.annotation.RequestMapping;
```

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02

```
@RestController
@RequestMapping ("/coletor") // URL: http://localhost:8080/coletor
public class Coletor {
 // URL: http://localhost:8080/coletor/iniciar
 @GetMapping (value = "/iniciar", produces = MediaType.TEXT_PLAIN_VALUE)
 public String iniciar () {
  StringBuilder pagina = new StringBuilder ();
  try {
   URL url = new URL
("http://journals.ecs.soton.ac.uk/java/tutorial/networking/urls/readingWriting.html");
    URLConnection url_connection = url.openConnection ();
    InputStream is = url_connection.getInputStream ();
    InputStreamReader reader = new InputStreamReader (is);
    BufferedReader buffer = new BufferedReader (reader);

    String linha;
    while ((linha = buffer.readLine ()) != null) {
     linha = Jsoup.parse (linha).text ();
     System.out.println (linha);
     pagina.append (linha);
    }
  } catch (IOException e) {
   pagina.append ("Erro: não foi possível coletar a página.");
  }
  return pagina.toString ();
 }
}
```

Centro Universitário UNA
Sistemas de Informação
Tecnologias Emergentes
Prática de Laboratório
Wesley Dias Maciel
2019/02



Execute o projeto novamente e observe a saída no navegador e no console do servidor.


```
Reading from and Writing to a URLConnectionhref="../sockets/index.html">href="../../index.html">href="../index.html">Working with URLsReading from and Writing to a
URLConnection (notes)If you've successfully used openConnection() to initiatecommunications with a URL, then you have a reference to a URLConnection object.The URLConnection
class contains many methods that let you communicatewith the URL over the network. URLConnection is an HTTP-centric class--manyof its methods are useful only when working
with HTTP URLs.However, most URL protocols let you read from and write to the connectionso this page shows you how to both read from and write to a URL througha URLConnection
object.Reading from a URLConnection(notes)The following program performs the same function as the program shown inReading Directly from a URL. However, ratherthan opening a
stream directly from the URL, this program explicitly opensa connection to the URL, gets an input stream on the connection, and readsfrom the input stream:import
java.net.*;import java.io.*;class ConnectionTest {public static void main(String[] args) {try {URL yahoo = new URL("http://www.yahoo.com/");URLConnection yahooConnection =
yahoo.openConnection();DataInputStream dis = new DataInputStream(yahooConnection.getInputStream());String inputLine;while ((inputLine = dis.readLine()) != null)
{System.out.println(inputLine);}dis.close();} catch (MalformedURLException me) {System.out.println("MalformedURLException: " + me);} catch (IOException ioe)
{System.out.println("IOException: " + ioe);}}}The output from this program should be identical to the output from theprogram that opens a stream directly from the URL. You
can use eitherway to read from a URL. However, sometimes reading from a URLConnectioninstead of reading directly from a URL might be more useful to you asyou can use the
URLConnection object for other tasks (like writing to theURL) at the same time.Again if, instead of output from the program, you see the following errormessage:IOException:
```