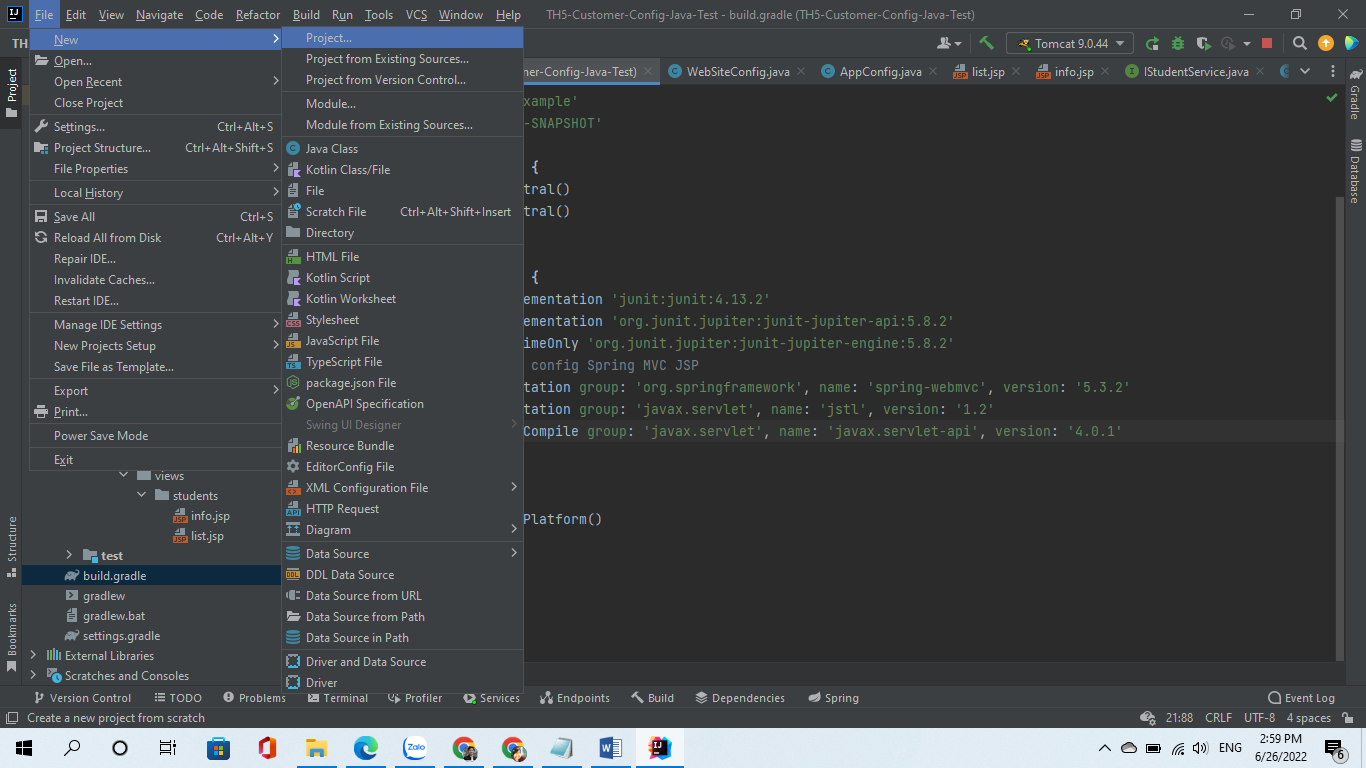
[Thực hành] Ứng dụng quản lý danh sách sinh viên

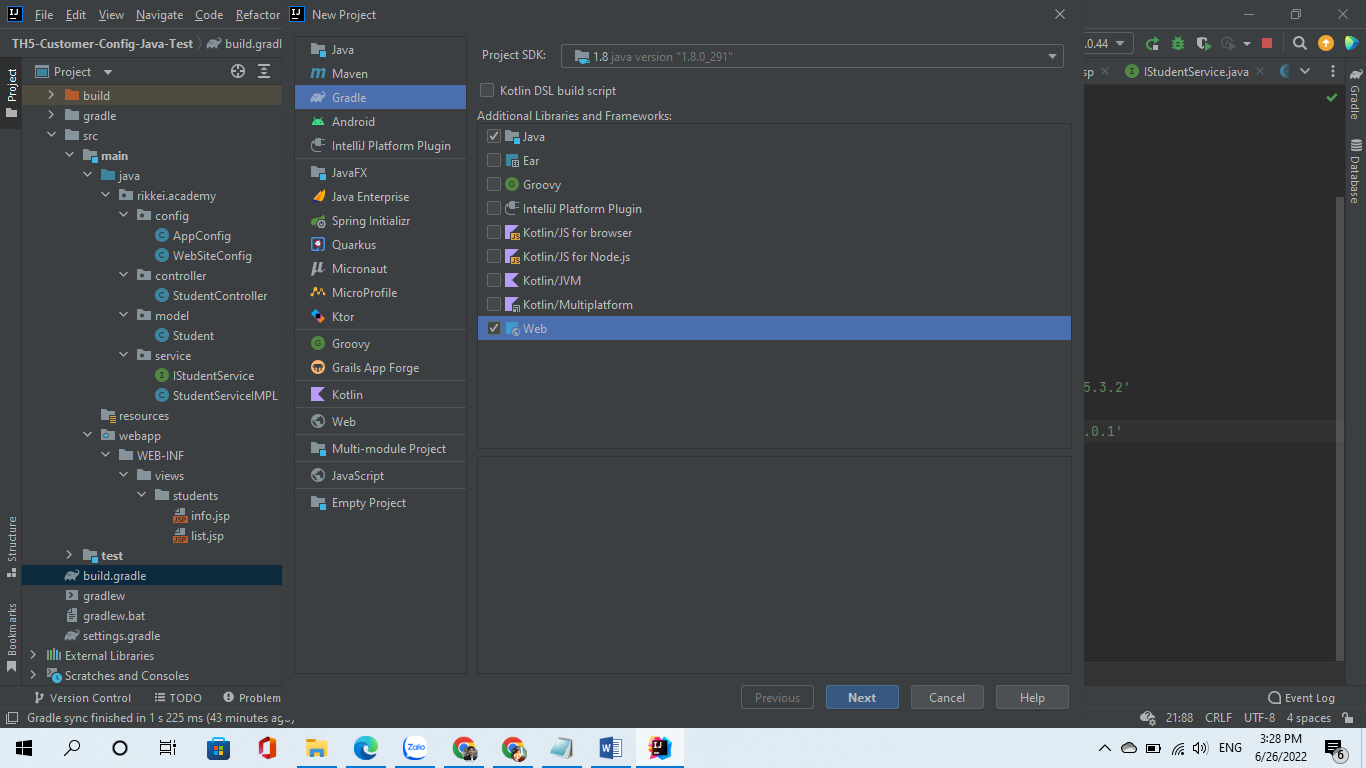
**Mục tiêu:**

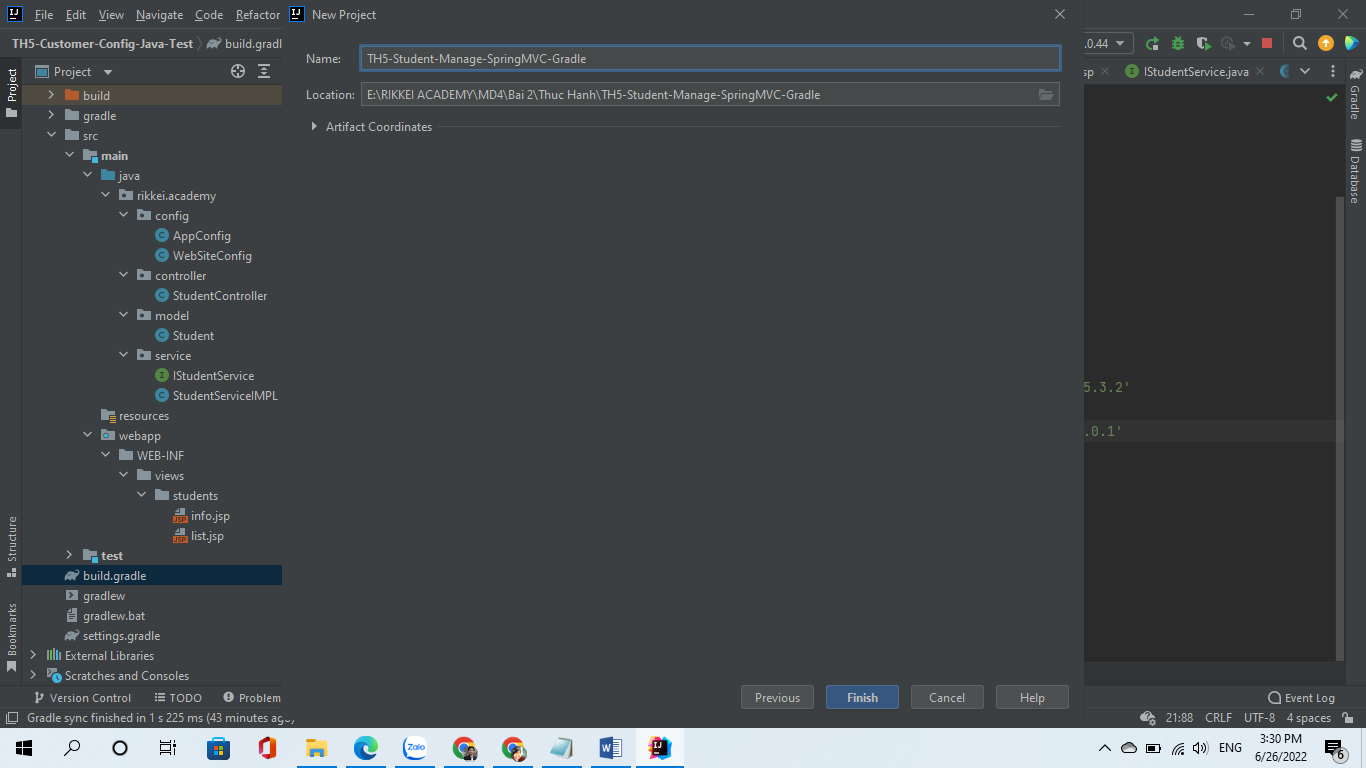
**Trong các bài thực hành trước chúng ta đã thực hiện các cách tạo dự án Spring MVC với kiểu config bằng các file: xml với cả 2 kiểu thư viện là: Maven vs Gradle. Trong bài thực hành này chúng ta sẽ tạo một dự án và config bằng các Class: JAVA với kiểu thư viện là: gradle**

**Hướng dẫn**

* Tạo dự án mới với thư viện: Gradle:



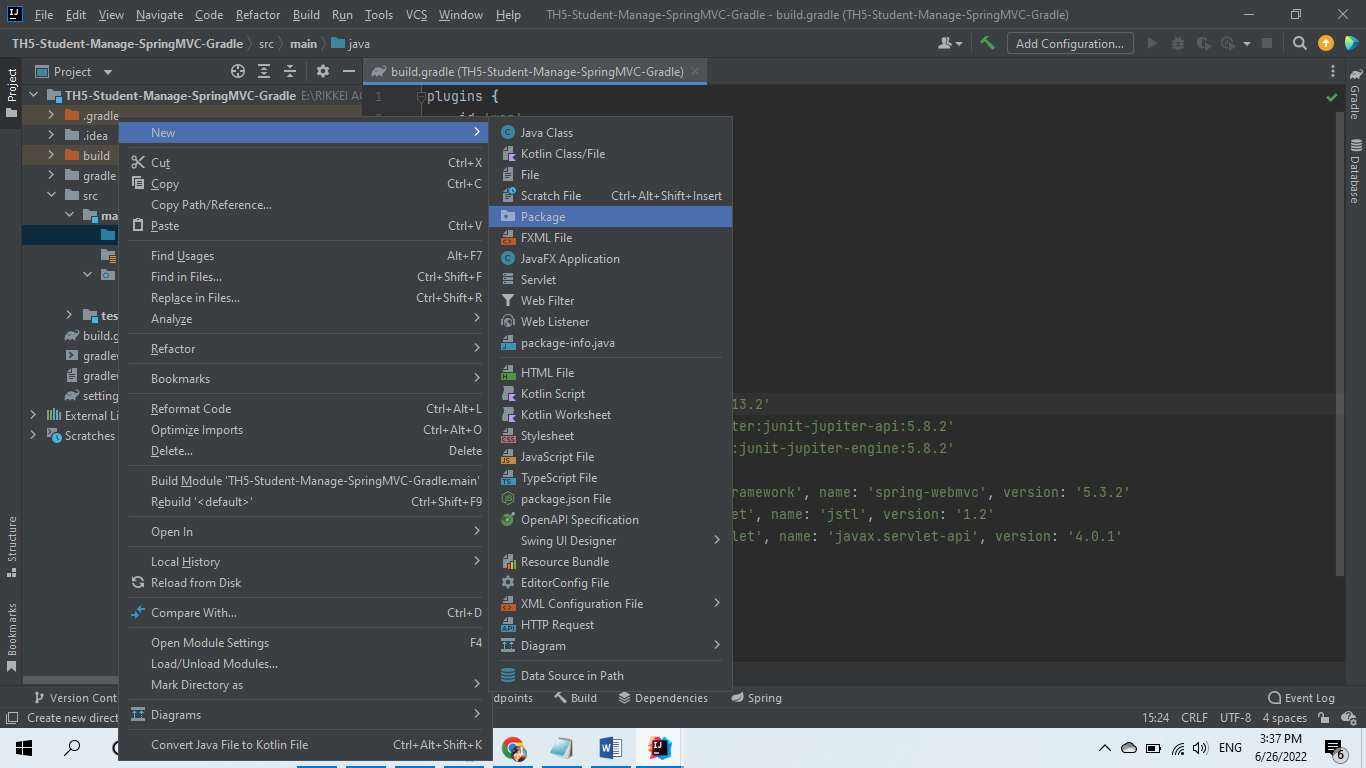


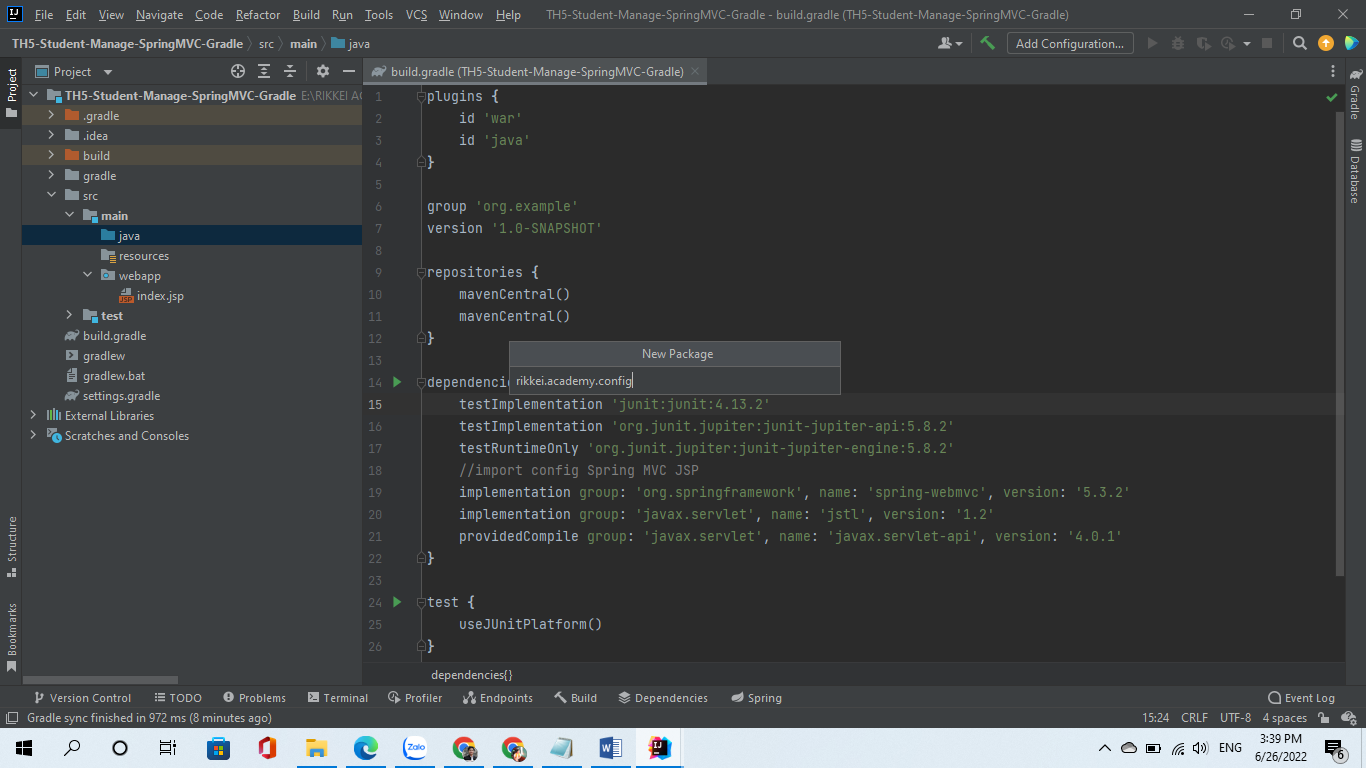


* Thêm thư viện vào **dependencies** trong **build.gradle:**

//import config Spring MVC JSP  
implementation group: 'org.springframework', name: 'spring-webmvc', version: '5.3.2'  
implementation group: 'javax.servlet', name: 'jstl', version: '1.2'  
providedCompile group: 'javax.servlet', name: 'javax.servlet-api', version: '4.0.1'

* Tạo package: rikkei.academy.config:





* Tạo Class: WebConfig (tương ứng với file: web.xml trong config kiểu dạng file xml)

Class này có nhiệm vụ khỏi tạo luồng chạy cho dự án Spring MVC.

* Thêm code vào như sau:

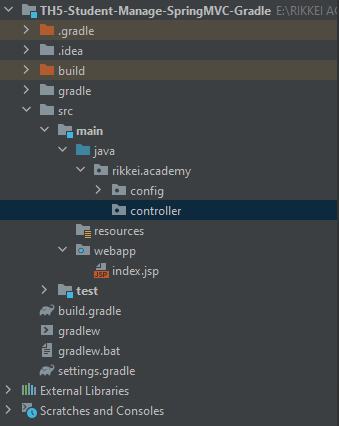
package rikkei.academy.config;  
  
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;  
  
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {  
 @Override  
 protected Class<?>[] getRootConfigClasses() {  
 return new Class[0];  
 }  
  
 @Override  
 protected Class<?>[] getServletConfigClasses() {  
 return new Class[0];  
 }  
  
 @Override  
 protected String[] getServletMappings() {  
 return new String[0];  
 }  
}

Lớp: AbstractAnnotationConfigDispatcherServletInitializer -> Chính là lớp mà Spring Framework đã xây dựng sẵn trong đó có 3 phương thức khởi tạo luồng chạy cho dự án.

* Tạo Class: AppConfig: (Lớp này giống như file: dispatcher-servlet.xml – có nhiệm vụ khởi tạo luồng chạy sang càng tầng controller và view)
* Thêm code vào như sau:
* package rikkei.academy.config;  
    
  import org.springframework.context.annotation.Bean;  
  import org.springframework.context.annotation.ComponentScan;  
  import org.springframework.context.annotation.Configuration;  
  import org.springframework.web.servlet.ViewResolver;  
  import org.springframework.web.servlet.config.annotation.EnableWebMvc;  
  import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;  
  import org.springframework.web.servlet.view.InternalResourceViewResolver;  
    
  @Configuration  
  @EnableWebMvc  
  @ComponentScan("rikkei.academy.controller")  
  public class AppConfig implements WebMvcConfigurer {  
   @Bean  
   public ViewResolver viewResolver() {  
   InternalResourceViewResolver viewResolver = new InternalResourceViewResolver();  
   viewResolver.setPrefix("/WEB-INF/views/");  
   viewResolver.setSuffix(".jsp");  
   return viewResolver;  
   }  
  }
* Sau đó chúng ta sử lại Class: WebConfig như sau:

package rikkei.academy.config;  
  
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;  
  
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {  
 @Override  
 protected Class<?>[] getRootConfigClasses() {  
 return new Class[]{AppConfig.class};  
 }  
  
 @Override  
 protected Class<?>[] getServletConfigClasses() {  
 return new Class[0];  
 }  
  
 @Override  
 protected String[] getServletMappings() {  
 return new String[]{"/"};  
 }  
}

* Chúng ta tạo thêm packeage: controller :



* Ta thấy rằng: @ComponentScan("rikkei.academy.controller") trong Class: AppConfig chính là config luồng chạy sang tầng controller
* Quan sát cách đường dẫn xuống tầng view và định dạng file jsp (Các bạn có thể mở lại dự án config với file: xml để so sách giữa file: dispatcher-servlet và Class: AppConfig
* Tạo package: service để tạo các Class trong tầng service:
* Tạo package: model -> Tạo Class: Student thêm code vào như sau:
* package rikkei.academy.model;  
    
  public class Student {  
   private Long id;  
   private String name;  
   private String email;  
   private String address;  
    
   public Student() {  
   }  
    
   public Student(Long id, String name, String email, String address) {  
   this.id = id;  
   this.name = name;  
   this.email = email;  
   this.address = address;  
   }  
    
   public Long getId() {  
   return id;  
   }  
    
   public void setId(Long id) {  
   this.id = id;  
   }  
    
   public String getName() {  
   return name;  
   }  
    
   public void setName(String name) {  
   this.name = name;  
   }  
    
   public String getEmail() {  
   return email;  
   }  
    
   public void setEmail(String email) {  
   this.email = email;  
   }  
    
   public String getAddress() {  
   return address;  
   }  
    
   public void setAddress(String address) {  
   this.address = address;  
   }  
    
   @Override  
   public String toString() {  
   return "Student{" +  
   "id=" + id +  
   ", name='" + name + '\'' +  
   ", email='" + email + '\'' +  
   ", address='" + address + '\'' +  
   '}';  
   }  
  }
* Tạo **IstudentService** trong package: service -> Thêm code vào như sau:

package rikkei.academy.service;  
  
import rikkei.academy.model.Student;  
  
import java.util.List;  
  
public interface IStudentService {  
 List<Student> findAll();  
  
 Student findById(Long id);  
}

* Tạo **StudentServiceIMPL** trong package: service -> Thêm code vào như sau:
* package rikkei.academy.service;  
    
  import rikkei.academy.model.Student;  
    
  import java.util.ArrayList;  
  import java.util.List;  
    
  public class StudentServiceIMPL implements IStudentService{  
   public static List<Student> *studentList* = new ArrayList<>();  
   static {  
   *studentList*.add(new Student(1L,"Chinh ND","chinhnd2@rikkeisoft.com","Hai Duong"));  
   *studentList*.add(new Student(2L,"DungLV","dunglv@rikkeisoft.com","Phu Tho"));  
   *studentList*.add(new Student(3L,"Toan TV","toan@rikkeisoft.com","Quang Ninh"));  
   }  
   @Override  
   public List<Student> findAll() {  
   return *studentList*;  
   }  
    
   @Override  
   public Student findById(Long id) {  
   return *studentList*.get(Math.*toIntExact*(id));  
   }  
  }
* Tạo classs: **StudentController** thêm code vào như sau:
* package rikkei.academy.controller;  
    
  import org.springframework.beans.factory.annotation.Autowired;  
  import org.springframework.stereotype.Controller;  
  import org.springframework.web.bind.annotation.GetMapping;  
  import org.springframework.web.bind.annotation.PathVariable;  
  import org.springframework.web.servlet.ModelAndView;  
  import rikkei.academy.model.Student;  
  import rikkei.academy.service.StudentServiceIMPL;  
    
  import javax.servlet.http.HttpServletRequest;  
  import java.util.List;  
  @Controller  
  public class StudentController {  
   @Autowired  
   private StudentServiceIMPL studentServiceIMPL;  
    
   @GetMapping(value = {"/","/students"})  
   public ModelAndView showList() {  
   ModelAndView modelAndView = new ModelAndView("students/list");  
   List<Student> studentList = studentServiceIMPL.findAll();  
   modelAndView.addObject("students", studentList);  
   return modelAndView;  
   }  
   @GetMapping("/detail/{id}")  
   public String detail(@PathVariable Long id, HttpServletRequest request) {  
   System.*out*.println("detail");  
   System.*out*.println("id" + id);  
   Long \_id = id-1;  
   Student student = studentServiceIMPL.findById(\_id);  
   request.setAttribute("student", student);  
   return "/students/info";  
   }  
  }
* Ta cần phải @Bean Service vào App-Config như sau: (Để @Autowride được tầng service sang controller)

@Bean  
public StudentServiceIMPL customerService() {  
 return new StudentServiceIMPL();  
}

* Tạo thư mục WEB-INF trong webapp => tạo thư mục views => tạo thư mục students => tạo các file .jsp ở trong đó:
* File: **list.jsp**
* <%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
  <style>  
   table {  
   border: 1px solid #000;  
   }  
   th, td {  
   border: 1px dotted #555;  
   }  
  </style>  
    
  <table>  
   <caption>Students List</caption>  
   <thead>  
   <tr>  
   <th>Id</th>  
   <th>Name</th>  
   <th>Email</th>  
   <th>Address</th>  
   </tr>  
   </thead>  
   <tbody>  
   <**c:forEach** var="st" items="${requestScope.students}">  
   <tr>  
   <td>  
   <**c:out** value="${st.id}"/>  
   </td>  
   <td>  
   <a href="detail/${st.id}">${st.name}</a>  
   </td>  
   <td>  
   <**c:out** value="${st.email}"/>  
   </td>  
   <td>  
   <**c:out** value="${st.address}"/>  
   </td>  
   </tr>  
   </**c:forEach**>  
   </tbody>  
  </table>
* File: **info.jsp**

<%@ **taglib** uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
<%@ **page** contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Detail</title>  
</head>  
<body>  
<h1>Student details</h1>  
<p>  
 <a href="/students">Back to students list</a>  
</p>  
  
<table border="1px" style="color: crimson">  
 <tr>  
 <td>STT</td>  
 <td>${requestScope["student"].getId()}</td>  
 </tr>  
 <tr>  
 <td>Name: </td>  
 <td>${requestScope["student"].getName()}</td>  
 </tr>  
 <tr>  
 <td>Email: </td>  
 <td>${requestScope["student"].getEmail()}</td>  
 </tr>  
 <tr>  
 <td>Address: </td>  
 <td>${requestScope["student"].getAddress()}</td>  
 </tr>  
</table>  
</body>  
</html>

* Xóa file: **index.jsp** mặc định của project, add **Tomcat** vào dự án và chạy quan sát kết quả:

### **Hướng dẫn nộp bài:**

Up code lên github.

Paste link github vào phần nộp bài.