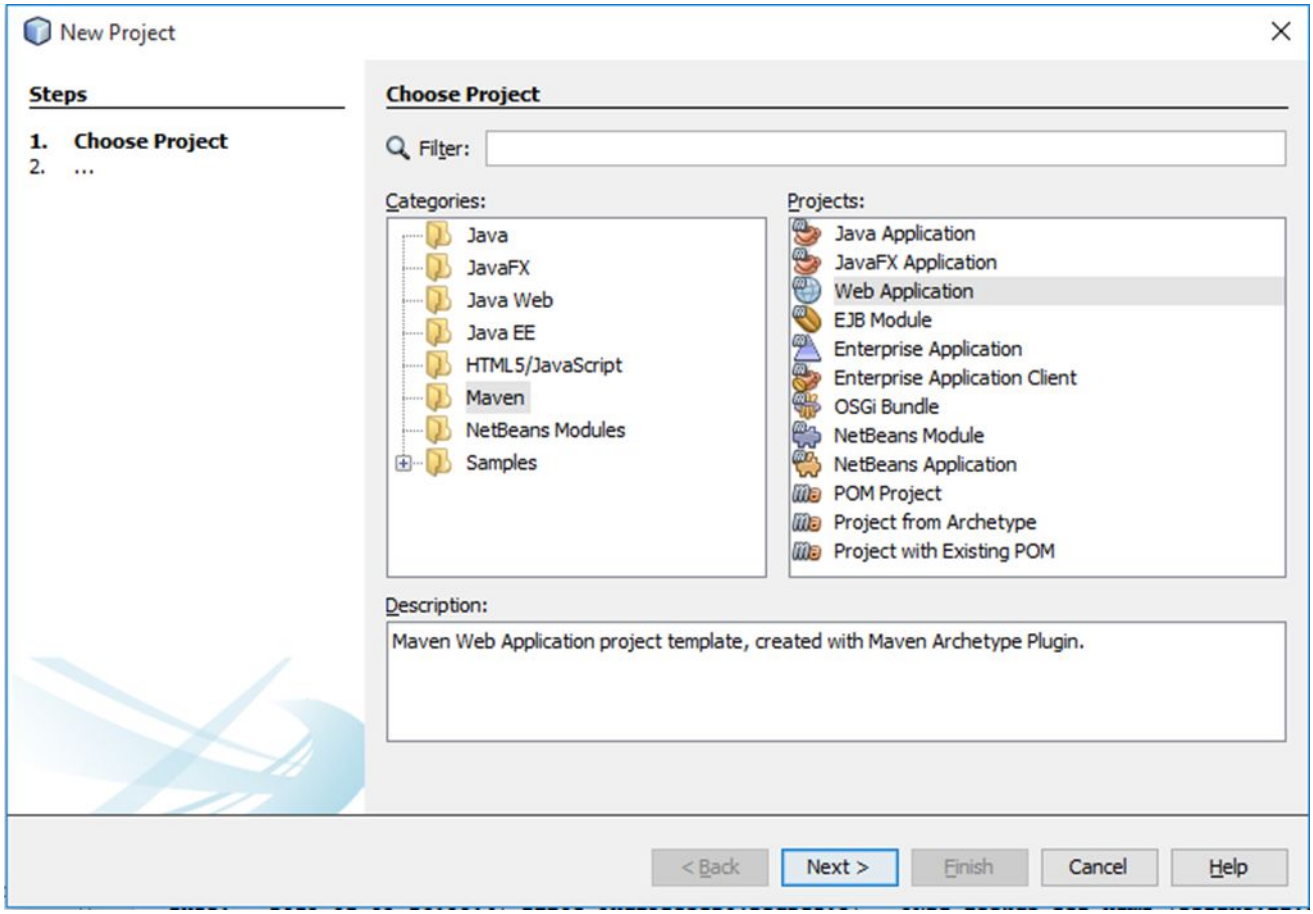
Bắt đầu với java spring mvc thật dễ làm cho người ta chán nản mà bỏ ngang, tìm trên mạng thì mỗi chỗ dậy một tị, lung tung xòe mãi mới chạy được

1. Một số khái niệm

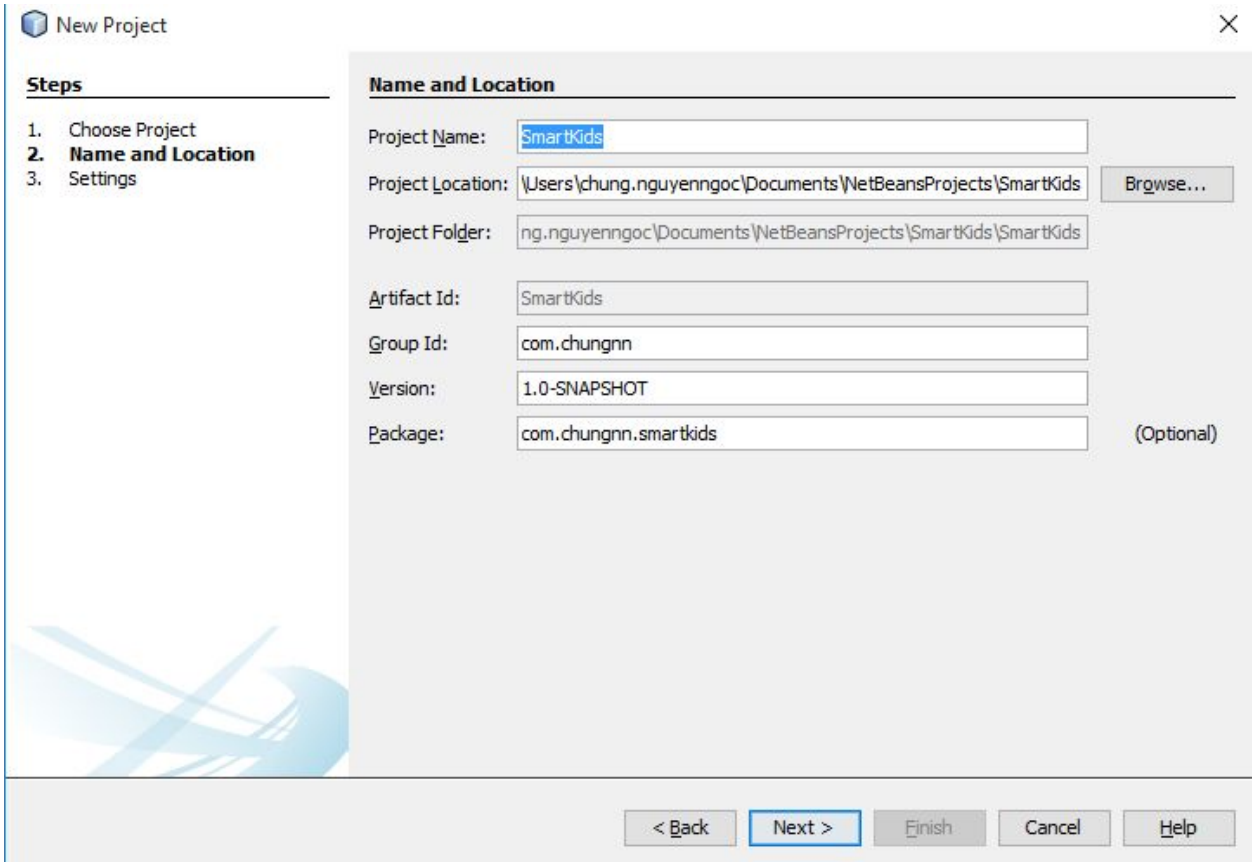
Spring MVC	Là một framework để giúp cho việc lập trình website với Java
Maven	Đây là một công cụ quản lý thư viện, giúp dev đỡ phải tự quản lý, chỉ cần config, nó sẽ tự gắn file thư viện vào
	Do đó, cần phải cài đặt maven trước khi làm các công việc khác. File để định nghĩa các thư viện có tên là pom.xml
Netbeans	IDE dùng cho việc phát triển, mình thấy netbeans dễ sử dụng hơn eclipse (ko biết có đúng ko)
GlassFish / Tomcat	Web server để deploy code sau khi code xong, thường listen port 8080
	Mình thấy GlassFish dễ sử dụng hơn Tomcat))

2. Thực hiện tạo Maven Project Spring MVC

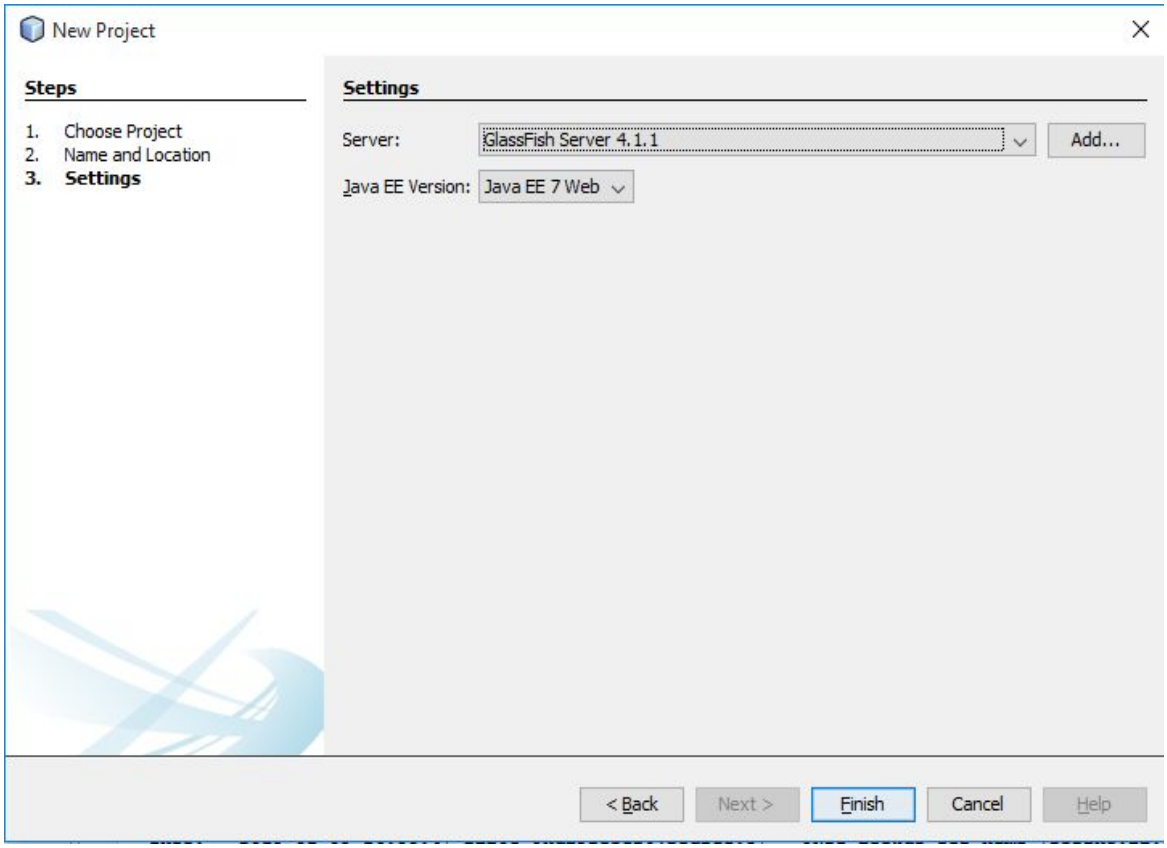
- Mở Netbeans
- Chọn New Project > Maven > Web application Rồi nhấn Next



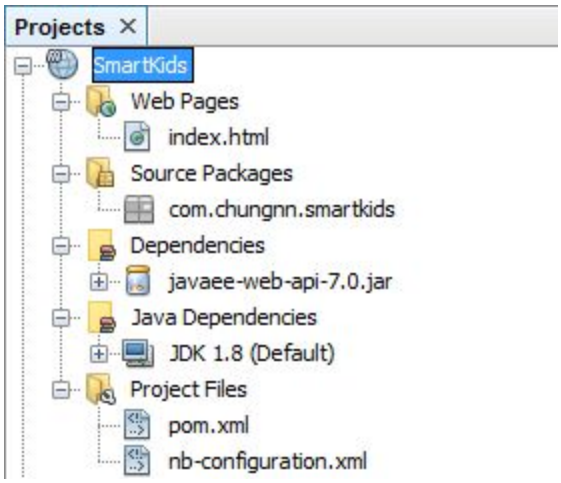
Điền tên Project bạn muốn vào, tham khảo thông tin mẫu dưới đây



- Nhấn next, tại màn hình tiếp theo chọn Server tương ứng GlassFish hoặc Tomcat
GlassFish đi cùng cặp được netbeans giới thiệu nên tích hợp vào ngon hơn, Tomcat hơi bị ghẻ lạnh
Tuy nhiên theo kinh nghiệm khi bắt đầu học cái này thì cứ chơi với GlassFish, còn Tomcat thì do trên server dùng để test mình cài Tomcat
Nên là sau khi chạy được với glassFish thì thử chạy với tomcat coi sao))



- Sau khi Nhấn Finish, ta sẽ được thư mục như sau



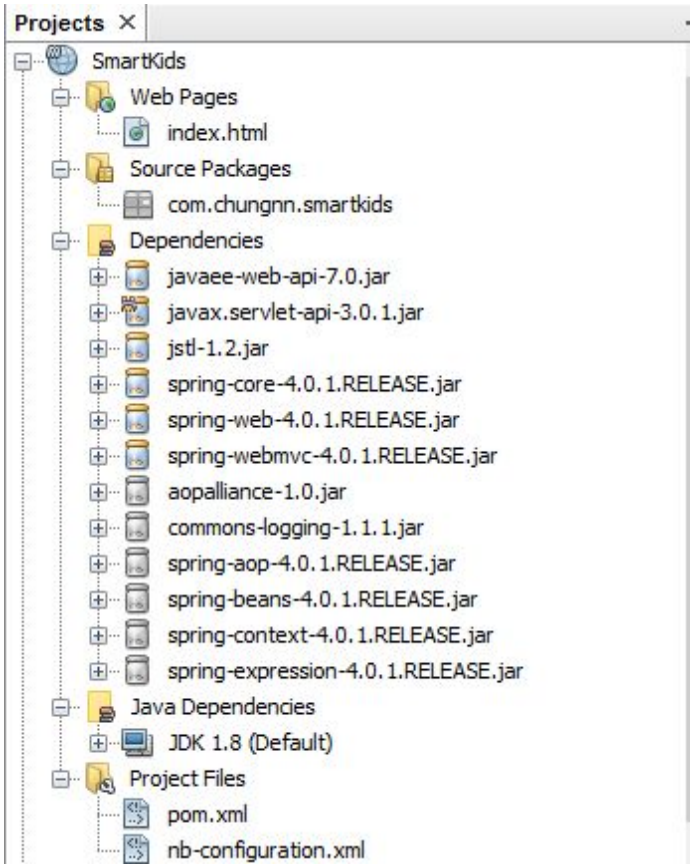
- File pom.xml được tạo sẵn, và chứa các nội dung mặc định, ta sẽ thêm các thư viện cần thiết bằng cách config thêm vào file này
- Tiếp theo, ta sẽ phải config project của mình sẽ sử dụng các thư viện spring, jstl trong file pom.xml
- Trong phần properties cho thêm khai báo

```
<spring.version>4.0.1.RELEASE</spring.version>
<jstl.version>1.2</jstl.version>
<javax.servlet.version>3.0.1</javax.servlet.version>
```

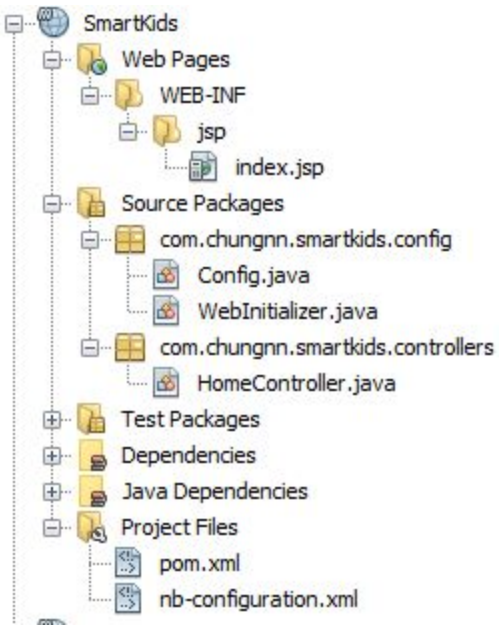
- Trong phần dependencies cho thêm các khai báo

```
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-core</artifactId>
            <version>${spring.version}</version>
        </dependency>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-web</artifactId>
            <version>${spring.version}</version>
        </dependency>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>${spring.version}</version>
        </dependency>

        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>javax.servlet-api</artifactId>
            <version>${javax.servlet.version}</version>
            <scope>provided</scope>
        </dependency>

        <dependency>
            <groupId>jstl</groupId>
            <artifactId>jstl</artifactId>
            <version>${jstl.version}</version>
        </dependency>
```

- Kích chuột phải vào Project và chọn Reload POM Lúc này trong cửa số Project ta sẽ thấy xuất hiện các dependency vừa được khai báo



- Công việc tiếp theo chúng ta sẽ tạo ra cấu trúc project như sau: chú ý: tất cả các thao tác phải tạo thủ công, không copy để tránh bị mấy lỗi linh tinh



Trong package com.chungnn.smartkids.config chứa 2 file Config.java và WebInitializer.java

Đây là 2 file cực kì quan trọng để giúp ứng dụng web của ta có thể chạy được

File Config.java

```
@Configuration
@ComponentScan("com.chungnn.smartkids")
@EnableWebMvc
public class Config extends WebMvcConfigurerAdapter {

    @Bean
    public UrlBasedViewResolver setupViewResolver() {
        UrlBasedViewResolver resolver = new UrlBasedViewResolver();
        resolver.setPrefix("/WEB-INF/jsp/");
        resolver.setSuffix(".jsp");
        resolver.setViewClass(JstlView.class);
        return resolver;
    }

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("/resources/**").addResourceLocations("/WEB-INF/resources/*");
    }
}
```

Với mỗi dòng bị IDE báo đỏ chỉ cần chọn import package , class tương ứng của nó là được
Ở đây có thể thấy việc thiết lập folder chỉ định các file jsp là /WEB-INF/jsp

File WebInitializer.java

```
public class WebInitializer implements WebApplicationInitializer{

    @Override
    public void onStartup(ServletContext servletContext) throws ServletException {
        AnnotationConfigWebApplicationContext ctx = new
AnnotationConfigWebApplicationContext();
        ctx.register(Config.class);
        ctx.setServletContext(servletContext);
        ServletRegistration.Dynamic servlet = servletContext.addServlet("dispatcher", new
DispatcherServlet(ctx));
        servlet.addMapping("/");
        servlet.setLoadOnStartup(1);
    }

}
```

(File này thực hiện thiết lập các tham số servlet cần thiết)

- Tạo file HomeController.java

```
@Controller

public class HomeController {

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public ModelAndView index(ModelAndView model) {
        //map.put("msg", "Hello Spring 4 Web MVC!");
        model.setViewName("index");
        return model;
    }

}
```

Có nghĩa là đăng ký đường dẫn mặc định khi thực hiện HTTP GET thì sẽ thực thi function index này, sau đó render file index.jsp

- Thực hiện chạy project (bấm vào nút run trên IDE)

À quên mất, nếu tạo file index.jsp trong Netbeans thì nội dung mặc định là như sau nên mới được kết quả như hình))

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Hello World!</h1>
    </body>
</html>
```

Đến đây chúng ta đã tạo xong được một project Java Spring MVC đơn giản có thể chạy thành công
Phần tiếp theo, chúng ta sẽ chế tạo các class phục vụ cho việc kết nối csdl, thực hiện CRUD sử dụng jdbc, jdbc template

3. Thực hiện CRUD với JDBC, JDBCTemplate

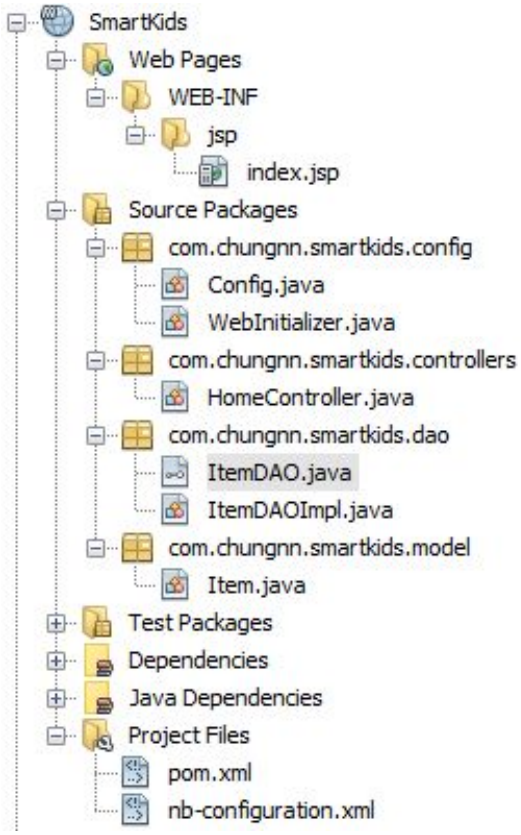
Tạo csdl mydb, bảng items như sau:

```
CREATE TABLE `items` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) DEFAULT NULL,
  `image_url` varchar(255) DEFAULT NULL,
  `audio_url` varchar(255) DEFAULT NULL,
  `cat` varchar(20) DEFAULT NULL,
  `cdate` datetime DEFAULT NULL,
  `mdate` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8;
```

- Khai báo thêm các thư viện dependency & reload POM

```
        <!-- Spring JDBC Support -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-jdbc</artifactId>
            <version>${spring.version}</version>
        </dependency>

        <!-- MySQL Driver -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.0.5</version>
        </dependency>
```

- Cấu trúc project mới sẽ như sau



- File Item.java

```
public class Item {

    private int id;
    private String title;
    private String image_url;
    private String audio_url;
    private String cat;
    private String cdate;
    private String mdate;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

    public String getImageUrl() {
        return image_url;
    }

    public String getAudioUrl() {
        return audio_url;
    }
```

```
    public String getCat() {
        return cat;
    }

    public String getCdate() {
        return cdate;
    }

    public String getMdate() {
        return mdate;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public void setImageUrl(String image_url) {
        this.image_url = image_url;
    }

    public void setAudioUrl(String audio_url) {
        this.audio_url = audio_url;
    }

    public void setCat(String cat) {
        this.cat = cat;
    }

    public void setCdate(String cdate) {
        this.cdate = cdate;
    }

    public void setMdate(String mdate) {
        this.mdate = mdate;
    }

    @Override
    public String toString() {
        return "Item{" + "id=" + id + ", title=" + title + ", image_url=" + image_url + ",
audio_url=" + audio_url + ", cat=" + cat + ", cdate=" + cdate + ", mdate=" + mdate + '}';
    }


}
```

Ta có thể khai báo các thuộc tính, sau đó dùng tính năng generate code của IDE, rồi chỉnh sửa lại 1 chút

- File ItemDAO.java

```
public interface ItemDAO {

    //Create
    public void save(Item item);
    //Read

    public Item getById(int id);
    //Update

    public void update(Item item);
    //Delete

    public void deleteById(int id);
    //Get All

    public List<Item> getAll();
}
```

- File ItemDAOImpl.java

```
public class ItemDAOImpl implements ItemDAO{


    @Override
    public void save(Item item) {

    }

    @Override
    public Item getById(int id) {
        return new Item();
    }

    @Override
    public void update(Item item) {

    }

    @Override
    public void deleteById(int id) {

    }

    @Override
    public List<Item> getAll() {

    }
}
```

- Tiếp theo ta tạo file cấu hình kết nối csdl

File /src/main/resources/application.properties

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://10.1.7.11:3306/mydb?characterEncoding=UTF-8&characterSetResults=UTF-8&
amps;useUnicode=yes
jdbc.username=root
jdbc.password=vietis@12345
```

- Tạo Bean thực hiện getDataSource trong Config.java

```
@Bean
    public DataSource getDataSource() {
        DriverManagerDataSource dataSource = new DriverManagerDataSource();
        dataSource.setDriverClassName(env.getRequiredProperty("jdbc.driverClassName"));
        dataSource.setUrl(env.getRequiredProperty("jdbc.url"));
        dataSource.setUsername(env.getRequiredProperty("jdbc.username"));
        dataSource.setPassword(env.getRequiredProperty("jdbc.password"));
        return dataSource;
    }
```

Chú ý: lúc này phải khai báo thêm Annotation:
    @PropertySource("classpath:application.properties")
và
   @Autowired
   private Environment env;

Tóm lại, file Config.java sẽ trở thành như sau:

```
@Configuration
@PropertySource("classpath:application.properties")
@ComponentScan("com.chungnn.smartkids")
@EnableWebMvc
public class Config extends WebMvcConfigurerAdapter {

    @Autowired
    private Environment env;

    @Bean
    public UrlBasedViewResolver setupViewResolver() {
        UrlBasedViewResolver resolver = new UrlBasedViewResolver();
        resolver.setPrefix("/WEB-INF/jsp/");
        resolver.setSuffix(".jsp");
        resolver.setViewClass(JstlView.class);
        return resolver;
    }

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("/resources/**").addResourceLocations("/WEB-INF/resources/*");
    }

    @Bean
    public DataSource getDataSource() {
        DriverManagerDataSource dataSource = new DriverManagerDataSource();
```

```
        dataSource.setDriverClassName(env.getRequiredProperty("jdbc.driverClassName"));
        dataSource.setUrl(env.getRequiredProperty("jdbc.url"));
        dataSource.setUsername(env.getRequiredProperty("jdbc.username"));
        dataSource.setPassword(env.getRequiredProperty("jdbc.password"));
        return dataSource;
    }
}
```

- Kế đó, ta tạo Bean JdbcTemplate (Cũng trong Config.java)

```
@Bean
    public JdbcTemplate getJdbcTemplate() {
        JdbcTemplate jdbc = new JdbcTemplate(getDataSource());
        return jdbc;
    }
```

và tạo Bean ItemDAO

```
@Bean
    public ItemDAO getItemDAO() {
        return new ItemDAOImpl();
    }
```

Sau khi khai báo các Bean này thì ta có thể dùng Annotation @Autowired

- Với việc chỉnh sửa như sau, ta có thể thu được kết quả liệt kê danh sách các item
+ File HomeController

```
@Autowired
    private ItemDAO itemDao;

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public ModelAndView index(ModelAndView model) {
        //map.put("msg", "Hello Spring 4 Web MVC!");
        List<Item> items = itemDao.getAll();
        model.addObject("items", items);
        model.setViewName("index");
        return model;
    }
```

+ File ItemDAOImpl
Thêm khai báo thuộc tính jdbcTemplate
@Autowired
   private JdbcTemplate jdbcTemplate;

Chỉnh sửa hàm getAll()

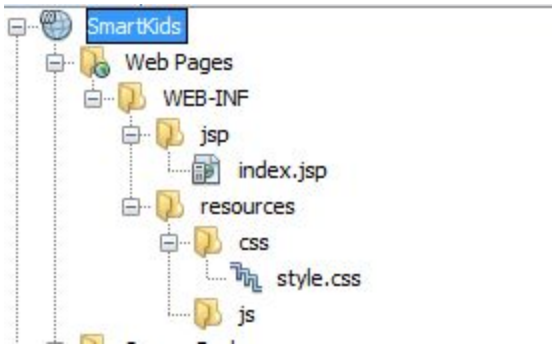
```
@Override
    public List<Item> getAll() {
        String sql = "SELECT * FROM items";
        List<Item> listItem = jdbcTemplate.query(sql, new RowMapper<Item>() {
            @Override
            public Item mapRow(ResultSet rs, int rowNum) throws SQLException {
                Item aItem = new Item();
                aItem.setId(rs.getInt("id"));
                aItem.setTitle(rs.getString("title"));
                aItem.setImageUrl(rs.getString("image_url"));
                aItem.setAudioUrl(rs.getString("audio_url"));
                aItem.setCat(rs.getString("cat"));
                aItem.setCdate(rs.getString("cdate"));
                aItem.setMdate(rs.getString("mdate"));
                return aItem;
            }
        });
        return listItem;
    }
```

+ File index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Danh sách Item</h1>
        <table>
            <tr>
                <td>STT</td> <td>Tiêu đề</td>
            </tr>
            <c:forEach var="item" items="${items}" varStatus="status">
                <tr>
                    <td>${status.index + 1}</td>
                    <td>${item.title}</td>
                </tr>
            </c:forEach>
        </table>
    </body>
</html>
```

Kết quả ta thu được trang có nội dung

- Thực hiện thêm css, js vào trang
+ Trước tiên tạo thêm folder & file cần thiết



+ Nội dung file style.css

```
table, tr, td {
    border: 1px solid black;
    border-collapse: collapse;
}
```

+ Đưa css vào trang

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="cp" value="${pageContext.request.servletContext.contextPath}" scope="request" />
<%-- Chú ý dòng ngày --%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
        <link rel="stylesheet" type="text/css" href="${cp}/resources/css/style.css" /> <%--
Sử dụng biến ${cp} --%>
    </head>
    <body>
        <h1>Danh sách Item</h1>
        <table>
            <tr>
                <td>STT</td> <td>Tiêu đề</td>
```

```
            </tr>
            <c:forEach var="item" items="${items}" varStatus="status">
                <tr>
                    <td>${status.index + 1}</td>
                    <td>${item.title}</td>
                </tr>
            </c:forEach>
        </table>
    </body>
</html>
```

Trong ngữ cảnh này thì ${cp} sẽ được http://localhost:8080/SmartKids/

Khi làm như vậy ta sẽ được các url tuyệt đối luôn đúng, đỡ dẫn tới load sai css, js sau này, nhất là ở các trang con

Hoặc ta sử dụng thẻ base như cách sau, cũng được kết quả tương tự

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="cp" value="${pageContext.request.servletContext.contextPath}" scope="request" />
<%-- Chú ý dòng ngày --%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
        <base url="${cp}"/> <%-- Sử dụng biến ${cp} --%>
        <link rel="stylesheet" type="text/css" href="resources/css/style.css" />
    </head>
    <body>
        <h1>Danh sách Item</h1>
        <table>
            <tr>
                <td>STT</td> <td>Tiêu đề</td>
            </tr>
            <c:forEach var="item" items="${items}" varStatus="status">
                <tr>
                    <td>${status.index + 1}</td>
                    <td>${item.title}</td>
                </tr>
            </c:forEach>
        </table>
    </body>
</html>
```

- Thực hiện thêm mới item
+ Bổ sung thêm link thêm mới Item

```
<h1>Danh sách Item</h1>
 <h3><a href="<c:url value ="/newEmployee"/>">New Item</a></h3>
```

+ Tạo các method để hứng request

```
@RequestMapping(value = "/newItem", method = RequestMethod.GET)
    public ModelAndView newItem(ModelAndView model) {
        Item newItem = new Item();
        model.addObject("item", newItem);
        model.setViewName("itemForm");
        return model;
    }

    @RequestMapping(value = "/saveItem", method = RequestMethod.POST)
    public ModelAndView saveItem(@ModelAttribute Item item) {
        itemDao.save(item);
        return new ModelAndView("redirect:/");
    }
```

+ Chỉnh lại trong ItemDAOImpl

```
@Override
    public void save(Item item) {
        String query = "INSERT INTO items (title, image_url, audio_url, cat, cdate, mdate)
VALUES (?,?,?,?,?,?)";
        jdbcTemplate.update(query, item.getTitle(), item.getImageUrl(), item.getAudioUrl(),
item.getCat(), item.getCdate(), item.getMdate());
    }
```

+ Tạo file itemForm.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="cp" value="${pageContext.request.servletContext.contextPath}" scope="request" />
<%-- Chú ý dòng ngày --%>
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
        <base url="${cp}"/> <%-- Sử dụng biến ${cp} --%>
        <link rel="stylesheet" type="text/css" href="resources/css/style.css" />
    </head>
    <body>
        <h1>Tạo mới Item</h1>
        <form:form action="saveItem" method="post" modelAttribute="item">
            <table>
                <form:hidden path="id"/>
                <tr>
                    <td>Title:</td>
                    <td><form:input path="title" /></td>
                </tr>
                <tr>
                    <td>ImageURL:</td>
                    <td><form:input path="imageUrl" /></td>
                </tr>
                <tr>
                    <td>AudioURL:</td>
                    <td><form:input path="audioUrl" /></td>
                </tr>
                <tr>
                    <td>Cat:</td>
                    <td><form:input path="cat" /></td>
```

```
                </tr>
                <tr>
                    <td>CDate: </td>
                    <td><form:input path="cdate" /></td>
                </tr>
                <tr>
                    <td>MDate: </td>
                    <td><form:input path="mdate" /></td>
                </tr>
                <tr>
                    <td colspan="2" align="center"><input type="submit" value="Save"></td>
                </tr>
            </table>
        </form:form>
    </body>
</html>
```

+ Chạy thử và xem kết quả

Danh sách Item

New Item

STT	Tiêu đề
1	Ô tô
2	Thuyền buồm
3	Xe tải
4	Xe đua
5	A
6	BBB

- Thực hiện update , delete item
+ Sửa file Controller

```
@RequestMapping(value = "/deleteItem", method = RequestMethod.GET)
    public ModelAndView deleteItem(HttpServletRequest request) {
        int itemId = Integer.parseInt(request.getParameter("id"));
        itemDao.deleteById(itemId);
        return new ModelAndView("redirect:/");
    }

    @RequestMapping(value = "/editItem", method = RequestMethod.GET)
    public ModelAndView editItem(HttpServletRequest request) {
        int itemId = Integer.parseInt(request.getParameter("id"));
        Item item = itemDao.getById(itemId);
        ModelAndView model = new ModelAndView("editItemForm");
        model.addObject("item", item);
        return model;
    }

    @RequestMapping(value = "/updateItem", method = RequestMethod.POST)
    public ModelAndView updateItem(@ModelAttribute Item item) {
        itemDao.update(item);
        return new ModelAndView("redirect:/");
    }
```

+ Sửa file index.jsp

```
<h1>Danh sách Item</h1>
        <h3><a href="<c:url value ="/newItem"/>">New Item</a></h3>
        <table>
            <tr>
                <td>STT</td> <td>Tiêu đề</td> <td></td>
            </tr>
            <c:forEach var="item" items="${items}" varStatus="status">
```

```
                <tr>
                    <td>${status.index + 1}</td>
                    <td>${item.title}</td>
                    <td>
                        <a href="<c:url value ="/editItem?id=${item.id}"/>">Edit</a>
                            
                        <a href="<c:url value ="/deleteItem?id=${item.id}"/>">Delete</a>
                    </td>
                </tr>
            </c:forEach>
        </table>
```

+ Tạo file editItemForm.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="cp" value="${pageContext.request.servletContext.contextPath}" scope="request" />
<%-- Chú ý dòng ngày --%>
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
        <base url="${cp}"/> <%-- Sử dụng biến ${cp} --%>
        <link rel="stylesheet" type="text/css" href="resources/css/style.css" />
    </head>
    <body>
        <h1>Cập nhật Item</h1>
        <form:form action="updateItem" method="post" modelAttribute="item">
            <table>
                <form:hidden path="id"/>
                <tr>
                    <td>Title:</td>
                    <td><form:input path="title" /></td>
                </tr>
                <tr>
                    <td>ImageURL:</td>
                    <td><form:input path="imageUrl" /></td>
                </tr>
                <tr>
                    <td>AudioURL:</td>
                    <td><form:input path="audioUrl" /></td>
                </tr>
                <tr>
                    <td>Cat:</td>
                    <td><form:input path="cat" /></td>
                </tr>
                <tr>
                    <td>CDate: </td>
                    <td><form:input path="cdate" /></td>
                </tr>
                <tr>
                    <td>MDate: </td>
                    <td><form:input path="mdate" /></td>
                </tr>
                <tr>
                    <td colspan="2" align="center"><input type="submit" value="Save"></td>
                </tr>
            </table>
        </form:form>
    </body>
</html>
```

```

```

+ Sửa lại method getById trong ItemDAOImpl.java

```
@Override
    public Item getById(int id) {
        String sql = "SELECT * FROM items WHERE id=" + id;
        return jdbcTemplate.query(sql, new ResultSetExtractor<Item>() {
            @Override
            public Item extractData(ResultSet rs) throws SQLException,
                    DataAccessException {
                if (rs.next()) {
                    Item aItem = new Item();
                    aItem.setId(rs.getInt("id"));
                    aItem.setTitle(rs.getString("title"));
                    aItem.setImageUrl(rs.getString("image_url"));
                    aItem.setAudioUrl(rs.getString("audio_url"));
                    aItem.setCat(rs.getString("cat"));
                    aItem.setCdate(rs.getString("cdate"));
                    aItem.setMdate(rs.getString("mdate"));
                    return aItem;
                }
                return null;
            }
        });
    }
```

+ Sửa lại method update trong ItemDAOImpl.java

```
@Override
    public void update(Item item) {
        String sql = "UPDATE items SET title=?, image_url=?, audio_url=?, cat=?, cdate=?,
mdate=? "
                + " WHERE id=?";
        jdbcTemplate.update(sql, item.getTitle(), item.getImageUrl(), item.getAudioUrl(),
item.getCat(), item.getCdate(), item.getMdate(), item.getId());
    }
```

+ Sửa lại method deleteById trong ItemDAOImpl.java

```
@Override
   public void deleteById(int id) {
        String sql = "DELETE FROM items WHERE id=?";
        jdbcTemplate.update(sql, id);
    }
```

Như vậy chúng ta đã hoàn thành xong việc CRUD, tuy nhiên còn 1 vấn đề đó là font chữ tiếng Việt không insert , update thành công

Do đó ta phải thực hiện tiếp phần sau

4. Setup tiếng Việt cho dự án, cho server

- Tạo file WEB-INF/glassfish-web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org//DTD GlassFish Application Server 3.1
Servlet 3.0//EN" "http://glassfish.org/dtds/glassfish-web-app_3_0-1.dtd">
<glassfish-web-app error-url="">
  <class-loader delegate="true"/>
  <jsp-config>
    <property name="keepgenerated" value="true">
      <description>Keep a copy of the generated servlet class' java code.</description>
    </property>
  </jsp-config>
  <parameter-encoding default-charset="UTF-8" />
</glassfish-web-app>
```

- Tạo file WEB-INF/web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <jsp-config>
        <jsp-property-group>
            <url-pattern>*.jsp</url-pattern>
            <page-encoding>UTF-8</page-encoding>
        </jsp-property-group>
    </jsp-config>
    <filter>
        <filter-name>encoding-filter</filter-name>
        <filter-class>
            org.springframework.web.filter.CharacterEncodingFilter
        </filter-class>
        <init-param>
            <param-name>encoding</param-name>
            <param-value>UTF-8</param-value>
        </init-param>
    </filter>
    <filter-mapping>
        <filter-name>encoding-filter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

- Đối với tomcat server, cần thực hiện chỉnh sửa tại TOMCAT_INSTALL_FOLDER/conf/server.xml

```
<Connector URIEncoding="UTF-8" port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />

<Connector URIEncoding="UTF-8" port="8009" protocol="AJP/1.3" redirectPort="8443" />
```

Đến đây chúng ta đã thực sự hoàn thành project CRUD với Spring MVC4 và hỗ trợ UTF8 đầy đủ