

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC
BỘ MÔN TOÁN – TIN ỨNG DỤNG



Báo cáo đồ án

Đề tài: Ứng dụng quản lý tài liệu bằng Liferay
Môn học: Lập trình J2EE

(Năm học: 2024 - 2025)

Sinh viên thực hiện:

Hoàng Võ Trí Bảo (22110022)

Đỗ Minh Triết (21110419)

Giảng viên hướng dẫn:

Ths. Hà Văn Thảo

Ngày 9 tháng 7 năm 2025

Mục lục

1. Giới thiệu.....	3
1.1 Mục tiêu dự án	3
1.2 Nhóm phát triển	3
1.3 Mã nguồn	3
2. Nền tảng công nghệ.....	4
2.1 Nền tảng Liferay	4
2.2 Các công nghệ khác	4
2.3 Môi trường thực thi	5
3. Kiến trúc.....	5
3.1 Mô hình MVC và Tính Module	5
3.2 Tính Module	6
4. Phân tích thiết kế.....	6
4.1 Yêu cầu chức năng	6
4.2 Thiết kế cơ sở dữ liệu	6
4.2 Thiết kế giao diện	7
5. Quy trình phát triển.....	8
5.1 Cài đặt môi trường	8
5.2 Phát triển ứng dụng	8
5.2.1 Tạo Liferay Workspace	8
5.2.2 Tạo Module Service Builder	10
5.2.3 Tạo Module Portlet	20
5.2.4 Triển khai công nghệ này lên server Liferay	45
6. Kết quả.....	45
7. Kết luận.....	49
7.1 Đánh giá	49
7.2 Những nâng cấp trong tương lai	49
7.3 Nhận định cuối cùng	50

1. Giới thiệu

1.1 Mục tiêu dự án

Mục tiêu chính của dự án ứng dụng quản lý tài liệu bằng Liferay là:

- Phát triển một hệ thống quản lý tài liệu để theo dõi và lưu trữ dữ liệu theo thời gian tùy theo mục đích sử dụng của cá nhân/tổ chức.
- Cung cấp giao diện người dùng tối giản nhưng trực quan để người dùng dễ dàng quản lý.
- Đảm bảo khả năng mở rộng và nâng cấp trong tương lai.
- Giúp nhóm tìm hiểu, học hỏi và nghiên cứu thêm về việc lập trình ứng dụng cho các mục tiêu tương lai.

1.2 Nhóm phát triển

Nhóm phát triển ứng dụng Quản lý Tài Liệu bao gồm các thành viên sau:

- **Trưởng nhóm phát triển :** Đỗ Minh Triết (21110419) – Dẫn dắt nhóm phát triển và giám sát việc triển khai ứng dụng đồng thời thiết kế giao diện người dùng.
- **Lập trình viên:** Hoàng Võ Trí Bảo (22110022) – Tập trung phát triển Liferay và tương tác cơ sở dữ liệu

1.3 Mã nguồn

Mã nguồn dự án được đăng tải trên github tại đường link này:

<https://github.com/UhtredRegress/J2ee-Final>

2. Nền tảng công nghệ:

2.1 Nền tảng Liferay:

Liferay được chọn làm nền tảng phát triển vì những lý do sau:

- **Khả Năng Tùy Biến Cao:** Liferay cho phép tùy chỉnh giao diện và chức năng dễ dàng, phù hợp với nhu cầu quản lý tồn kho giày.
- **Hỗ Trợ Nhiều Tính Năng:** Nền tảng này tích hợp sẵn nhiều tính năng hữu ích như quản lý người dùng, quản lý nội dung và cộng tác.
- **Khả Năng Mở Rộng:** Liferay có thể mở rộng để đáp ứng nhu cầu phát triển trong tương lai và tăng cường tính năng mà không cần thay đổi cấu trúc hiện có.
- **Bảo Mật Cao:** Nền tảng này cung cấp các tính năng bảo mật mạnh mẽ để bảo vệ dữ liệu người dùng và thông tin nhạy cảm.

2.2 Các công nghệ khác

Ngoài Liferay, dự án còn sử dụng các công nghệ sau:

- **Hệ Quản Trị Cơ Sở Dữ Liệu:** PostgreSQL được sử dụng để quản lý và lưu trữ dữ liệu tồn kho.
- **Ngôn Ngữ Lập Trình:** Java được sử dụng cho phần logic phía máy chủ, trong khi HTML, CSS và JavaScript được sử dụng cho giao diện người dùng.
- **Công Cụ Quản Lý Phiên Bản:** Git và Github được sử dụng để quản lý mã nguồn và phối hợp làm việc nhóm.
- **Máy Chủ Ứng Dụng:** Apache Tomcat được chọn làm máy chủ ứng dụng để triển khai ứng dụng web.

2.2 Môi trường thực thi

Dự án được thực thi trong môi trường sau:

- **Bộ nhớ:** 16.0 GB
- **Vi xử lý:** Intel® Core™ i7-6600U × 4 ~ 2.8GHz
- **Đồ họa liên:** Intel® HD Graphics 520
- **Hệ điều hành:** Windows 10

Môi trường Liferay:

- **Liferay Developer Studio:** Phiên bản 3.9.8.202212271250-ga9
- **Liferay Portal:** Liferay Community Edition Portal 7.4.3.112 CE GA112
- **Java JDK:** Java JDK 11

3. Kiến trúc

3.1 Mô hình MVC và Tính Module

Ứng dụng được xây dựng trên mô hình MVC. Mô hình MVC có ba tầng:

- **Model (Mô Hình):** Tầng mô hình giữ dữ liệu ứng dụng và logic để thao tác với dữ liệu đó.
- **View (Hiển Thị):** Tầng hiển thị chứa logic để hiển thị dữ liệu.
- **Controller (Điều Khiển):** Là trung gian trong mô hình MVC, Controller chứa logic để truyền dữ liệu qua lại giữa các tầng Model và View.

Các thành phần của MVC tương tác với nhau theo cách sau:

- Người dùng tương tác với View.
- View chuyển các yêu cầu của người dùng tới Controller.
- Controller xử lý các yêu cầu và tương tác với Model để thực hiện các thay đổi hoặc lấy dữ liệu.
- Model thông báo cho View về bất kỳ sự thay đổi nào trong dữ liệu.
- View cập nhật hiển thị dựa trên dữ liệu từ Model.

Ưu điểm của kiến trúc này:

- Tách biệt rõ ràng các tầng, dễ bảo trì và mở rộng
- Sử dụng các công nghệ và thư viện phổ biến của Liferay để triển khai và vận hành
- Đảm bảo an ninh, tính bảo mật cao nhờ các tính năng như xác thực, phân quyền

3.2 Tính Module

Các ứng dụng của Liferay được chia thành nhiều module riêng biệt. Với Service Builder, tầng Model được tạo thành một module Service và một module API. Điều này tương ứng với Model trong mô hình MVC. Tầng View và tầng Controller chia sẻ một module, đó là module Web. Trong một ứng dụng lớn, lớp -Portlet có thể trở nên cồng kềnh và khó quản lý nếu nó chứa tất cả logic điều khiển. Liferay cung cấp các lớp lệnh MVC để chia nhỏ chức năng điều khiển:

- **MVCActionCommand:** Sử dụng các lớp -ActionCommand để chứa từng hành động của portlet, được gọi bởi các URL hành động.
- **MVCRenderCommand:** Sử dụng các lớp -RenderCommand để chứa phương thức render mà chuyển hướng đến JSP thích hợp, bằng cách phản hồi các URL render.
- **MVCResourceCommand:** Sử dụng các lớp -ResourceCommand để thực hiện việc phục vụ tài nguyên trong portlet MVC, bằng cách phản hồi các URL tài nguyên.

4. Phân tích thiết kế

4.1 Yêu cầu chức năng

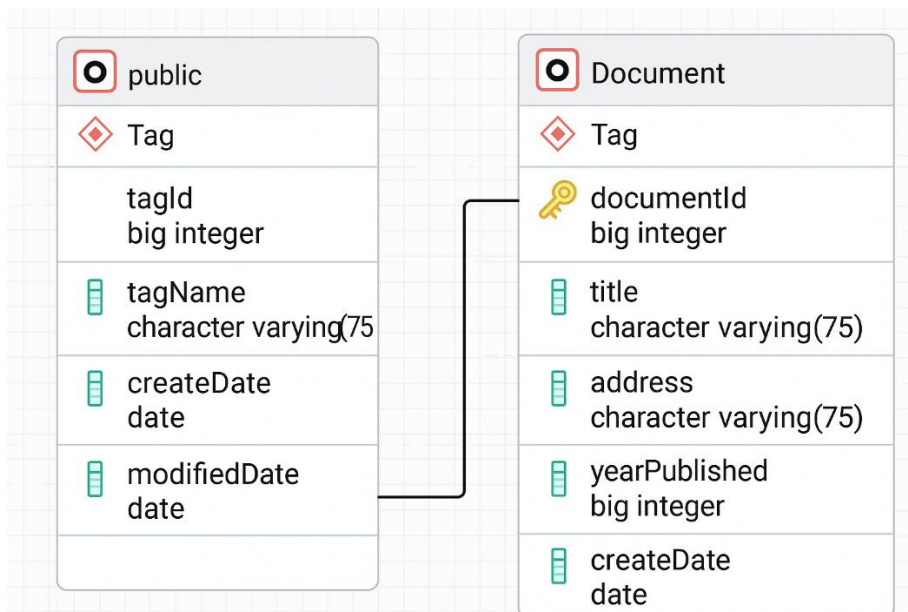
Ứng dụng được thiết kế với phạm vi nhỏ và mục đích sử dụng nội bộ/cá nhân nên yêu cầu chức năng tương đối đơn giản.

Nhóm chức năng thiết yếu

- Tạo, xem, sửa và xóa tài liệu
- Tạo, xem, sửa và xóa tag
- Giao diện người dùng tối giản, dễ sử dụng, đảm bảo tính responsive cho các thiết bị khác nhau

4.2 Thiết kế cơ sở dữ liệu

Lược đồ ERD cho cơ sở dữ liệu:



4.3 Thiết kế giao diện

Ứng dụng được xây dựng với giao diện đơn giản, thân thiện và dễ sử dụng. Dưới đây là các giao diện chính của hệ thống:

- **Trang chủ(/home)**

Hiển thị trang chủ của ứng dụng, bao gồm 2 nút Document và Tag để quản lý tài liệu và thẻ loại

/home/create:

Đây là trang dùng để thêm tài liệu vào

/home/edit/{id}

Đây là trang dùng để chỉnh sửa tài liệu

- **Trang quản lý tài liệu(/document)**

Đây là trang dùng để quản lý tài liệu, gồm:

/document/index:

Hiển thị các tài liệu hiện chứa trong database

/document/delete:

Dùng để xóa tài liệu không cần thiết đi

- **Trang quản lý Tag(/tag)**

Đây là trang dùng để quản lý các thẻ loại(tag), gồm:

/tag/index:

Hiển thị các tag đang lưu trong database

/tag/create:

Tạo các tag mới

/tag/delete:

Xóa các tag không cần thiếu đi

5. Quy trình phát triển:

5.1 Cài đặt môi trường

- **Cài đặt Liferay:** Thực hiện cài đặt Liferay theo hướng dẫn từ trang chính thức của Liferay
- **Cài đặt Liferay Developer Studio:** Thực hiện cài đặt Liferay Developer Studio theo hướng dẫn từ trang chính thức của Liferay[3].
- **Cài đặt Cơ sở dữ liệu SQL (PostgreSQL):** Khi giải nén Portal bằng cách tự động tải khi tạo Workspace hoặc tải thủ công, tìm đến file portal-ext.properties và thêm các dòng sau:

```
jdbc.default.driverClassName=org.postgresql.Driver
```

```
jdbc.default.url=jdbc:postgresql://localhost:5432/lportal
```

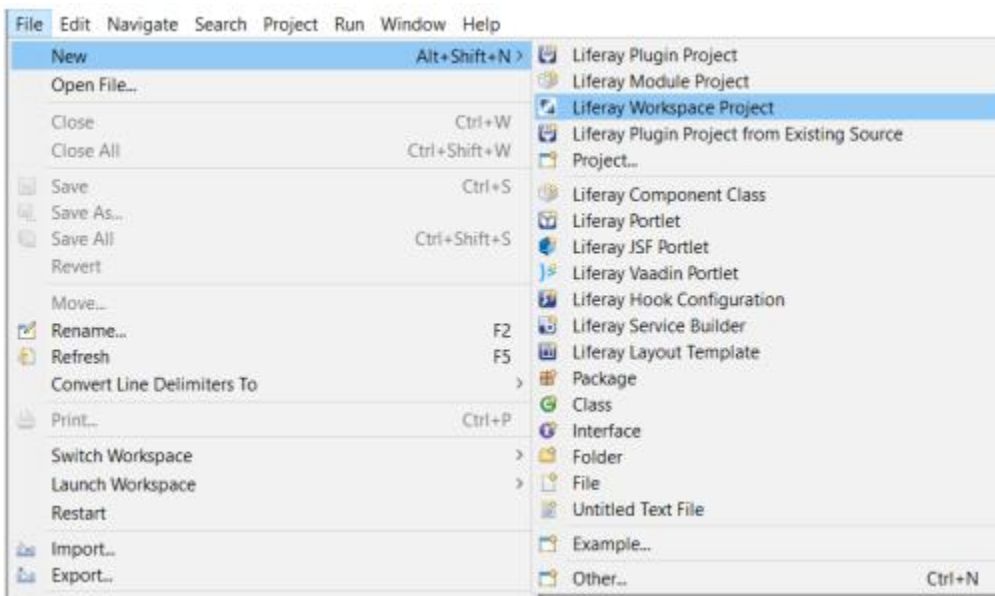
```
jdbc.default.username=<username>
```

```
jdbc.default.password=<password>
```

5.2 Phát triển ứng dụng

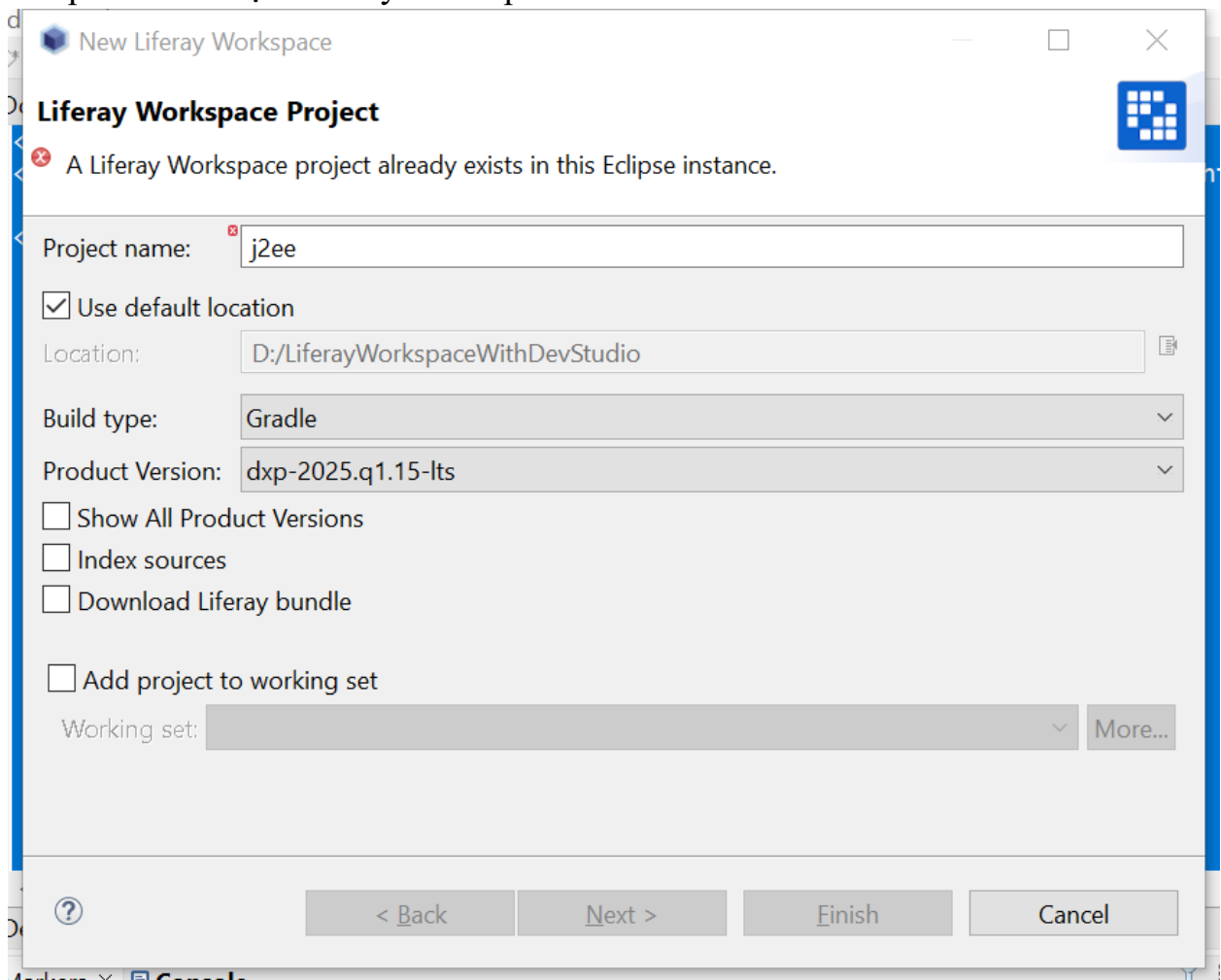
5.2.1 Tạo Liferay Workspace

Để tạo một Liferay Workspace trong Developer Studio, chọn File → New → Liferay Workspace Project.



Một hộp thoại New Liferay Workspace xuất hiện, cung cấp một số tùy chọn cấu hình. Làm theo hướng dẫn dưới đây để tạo workspace của bạn.

- Đặt tên cho workspace.
- Chọn vị trí bạn muốn đặt workspace.
- Đánh dấu tích vào ô Download Liferay bundle nếu muốn tự động tạo một instance Liferay trong workspace của mình. Nếu chọn, ô trống yêu cầu đặt tên cho server hiện ra.
- Lưu ý: Nếu muốn cấu hình một gói Liferay hiện có vào workspace của mình, có thể tạo một thư mục cho gói đó trong workspace và cấu hình nó trong tệp gradle.properties của workspace bằng cách đặt thuộc tính liferay.workspace.home.dir.
- Đánh dấu tích vào ô Add project to working set nếu muốn workspace là một phần của một tập làm việc lớn hơn mà bạn đã tạo trong Developer Studio.
- Nhấp Finish để tạo Liferay Workspace.



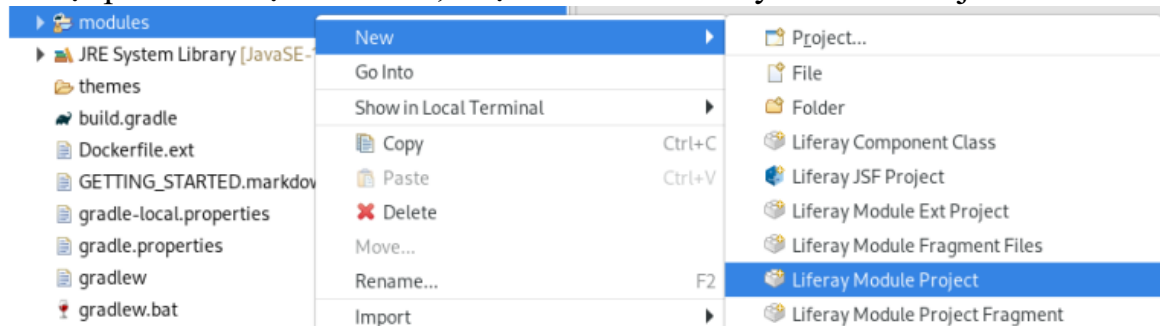
Một hộp thoại xuất hiện yêu cầu mở góc nhìn Liferay Workspace. Nhấp Yes, góc nhìn sẽ chuyển sang Liferay Workspace.

Lưu ý: Cũng có thể tạo một Liferay Workspace trong quá trình khởi động ban đầu của một instance Liferay Developer Studio. Một Liferay Workspace trong Liferay Developer Studio đã được tạo thành công. Nếu sử dụng Liferay Developer Studio, cũng có thể tạo workspace trong quá trình khởi động ban đầu.

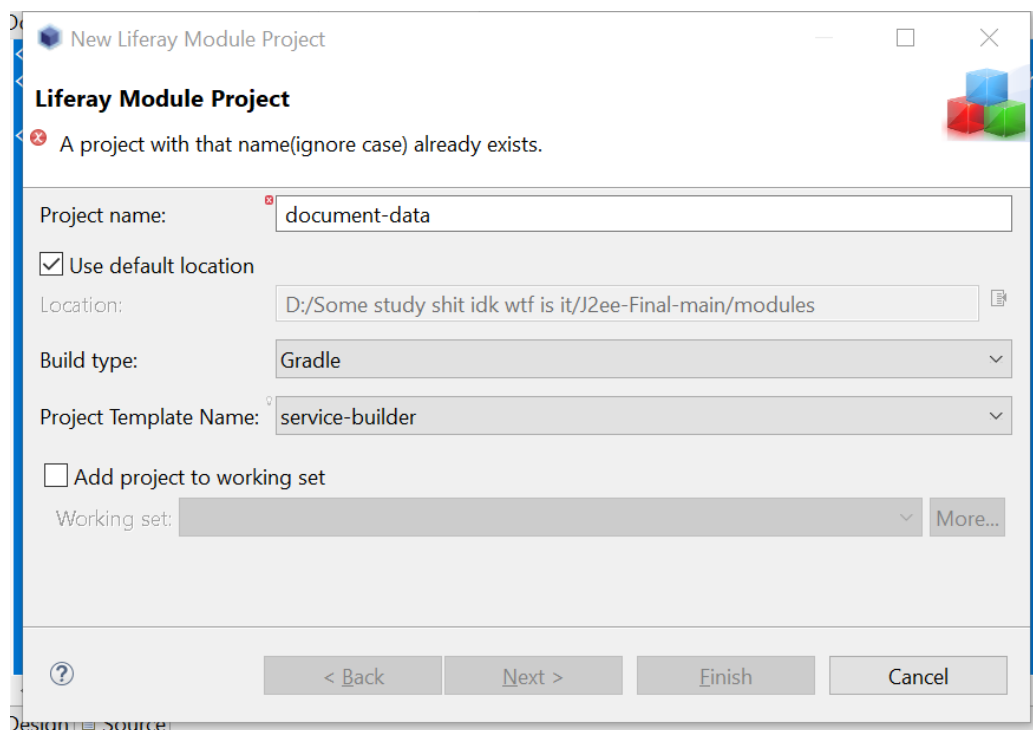
5.2.2 Tạo Module Service Builder

Đây sẽ là module Model như đã nói ở phần kiến trúc. Các bước tạo module:

- Chuột phải thư mục modules, chọn New → Liferay Module Project

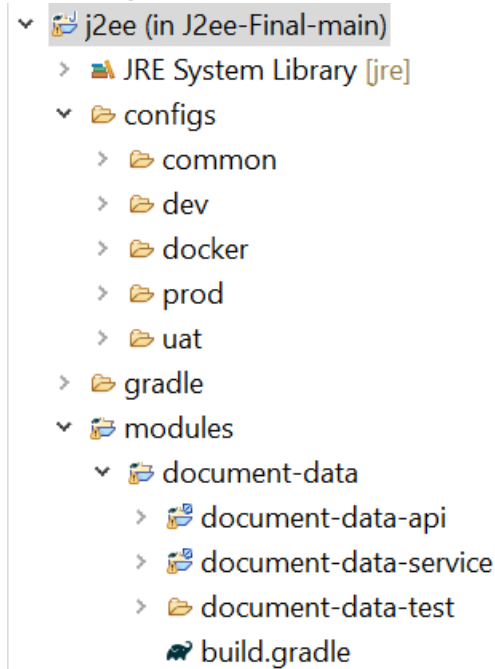


- Hộp thoại hiện ra, đặt tên cho Module và chọn Project Template Name là service-builder. Sau đó nhấn Next.



- Hộp thoại mới hiện ra để đặt tên package. Sau khi đặt tên xong nhấn Finish và chờ Liferay cài đặt module mới này.

- Sau khi chạy xong, nhìn vào thư mục modules sẽ thấy thư mục mới có tên trùng với tên đã đặt và có cấu trúc như sau



- Mở tệp <tên dự án>-service/service.xml lên. Thay đoạn code trong file bằng đoạn code sau

```
<?xml version="1.0"?>
```

```
<!DOCTYPE service-builder PUBLIC "-//Liferay//DTD Service Builder 7.4.0//EN"
"http://www.liferay.com/dtd/liferay-service-builder_7_4_0.dtd">
```

```
<service-builder dependency-injector="ds" package-path="com.document.data">
```

```
  <namespace>DocumentData</namespace>
```

```
  <entity local-service="true" name="Tag" remote-service="true">
```

```
    <column name="tagId" primary="true" type="long" />
```

```
    <column name="tagName" type="String"/>
```

```
    <column name="createDate" type="Date" />
```

```
    <column name="modifiedDate" type="Date" />
```

```
  </entity>
```

```

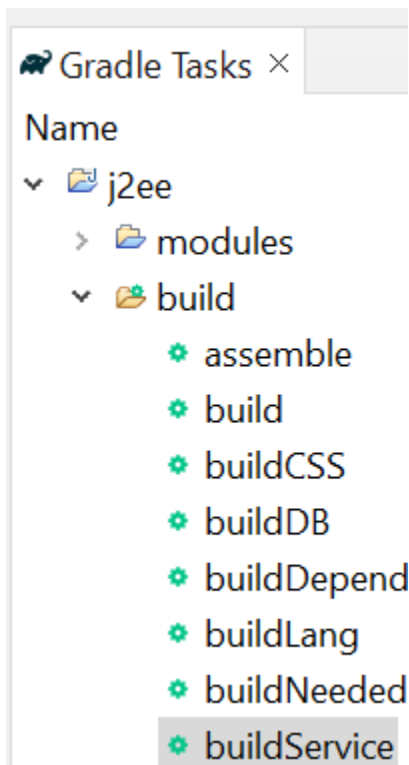
<entity local-service="true" name="Document" remote-service="true">
    <column name="documentId" primary="true" type="long" />
    <column name="tagId" type="long" />
    <column name="title" type="String"/>
    <column name="address" type="String"/>
    <column name="author" type="String"/>
    <column name="yearPublished" type="long"/>
    <column name="createDate" type="Date" />
    <column name="modifiedDate" type="Date" />

    <reference entity="Tag" package-path="com.document.data"/>
</entity>

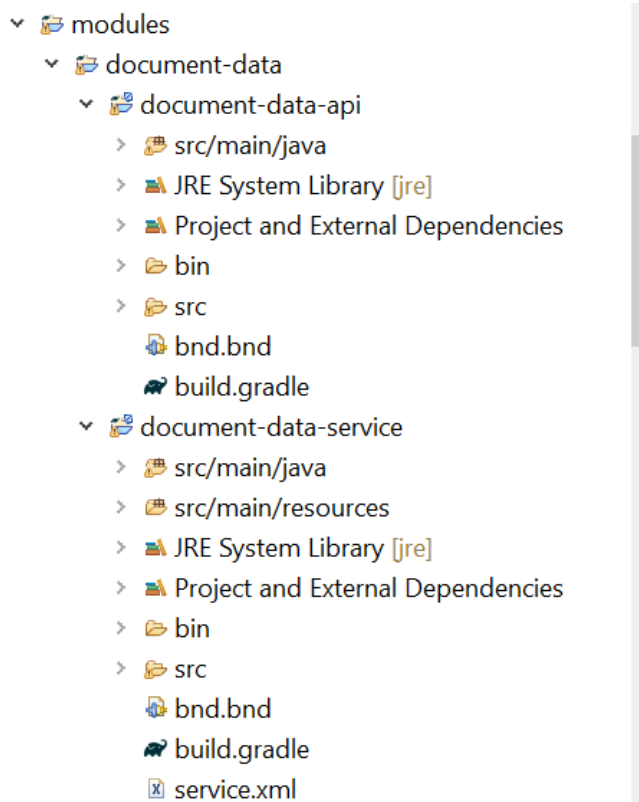
```

</service-builder>

- Chuyển qua View của Gradle Tasks, chọn <tên workspace> → modules → <tên module service> -> buildService để xây dựng Service và chờ Liferay xử lý



- Sau khi xử lý thành công, cấu trúc thư mục của module sẽ có thêm các thư mục src



- Nhấn Ctrl + F5 để Liferay cập nhật trạng thái của Workspace. Các thư mục sẽ được cập nhật các package tương ứng.
- Các bước tiếp theo ta tập trung vào thư mục <tên module>-service. Sau đây là 1 số đoạn code quan trọng trong các class: TagLocalServiceImpl.java, DocumentLocalServiceImpl.java, các class còn lại có thể tham khảo tại source code gốc đã để ở trên

TagLocalServiceImpl.java

```
package com.document.data.service.impl;

import com.document.data.exception.NoSuchTagException;
import com.document.data.model.Tag;
import com.document.data.service.base.TagLocalServiceBaseImpl;
import com.liferay.portal.aop.AopService;
```

```
import com.liferay.portal.kernel.exception.PortalException;
```

```
import java.util.Date;
```

```
import java.util.List;
```

```
import org.osgi.service.component.annotations.Component;
```

```
@Component(
```

```
    property
```

```
=
```

```
    "model.class.name=com.document.data.model.Tag",
```

```
    service = AopService.class
```

```
)
```

```
public class TagLocalServiceImpl extends TagLocalServiceBaseImpl {
```

```
    public Tag addTag(String tagName) {
```

```
        long
```

```
        tagId
```

```
=
```

```
        counterLocalService.increment(Tag.class.getName());
```

```
        Tag tag = super.createTag(tagId);
```

```
        tag.setTagId(tagId);
```

```
        tag.setTagName(tagName);
```

```
        Date now = new Date();
```

```
        tag.setCreateDate(now);
```

```
        tag.setModifiedDate(now);
```

```
        return super.addTag(tag);
```

```
    }
```

```
    public Tag editTag(long tagId, String tagName) throws
```

```
PortalException {
```

```

        Tag foundTag = super.getTag(tagId);
        foundTag.setTagName(tagName);
        foundTag.setModifiedDate(new Date());

        return super.updateTag(foundTag);
    }

    public List<Tag> getAllTag(){
        return tagPersistence.findAll();
    }
}

```

DocumentLocalServiceImpl.java

```

package com.document.data.service.impl;

import com.document.data.model.Document;
import com.document.data.model.Tag;
import com.document.data.service.TagLocalService;
import com.document.data.service.base.DocumentLocalServiceBaseImpl;

import com.liferay.portal.aop.AopService;
import com.liferay.portal.kernel.exception.PortalException;
import com.document.data.viewmodel.DocumentIndexViewModel;

import java.io.File;
import java.io.IOException;

```

```

import java.nio.file.Files;
import java.nio.file.StandardCopyOption;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
import java.util.List;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    property = "model.class.name=com.document.data.model.Document",
    service = AopService.class
)
public class DocumentLocalServiceImpl extends DocumentLocalServiceBaseImpl {
    @Reference
    private TagLocalService _tagLocalService;

    private String fileHandle(File file, String title, Date now) throws IOException {
        try {
            if (file == null || !file.exists()) {
                throw new IllegalArgumentException("Source file is null or doesn't exist: " +
file);
            }
            String basePath = new File("").getAbsolutePath();
            String uploadDir = basePath + File.separator + "upload";

            File targetDir = new File(uploadDir);

```



```

        if (!targetDir.exists()) {
            targetDir.mkdirs();
        }

        String originalFileName = file.getName();
        String extension = "";

        int i = originalFileName.lastIndexOf('.');
        if (i > 0) {
            extension = originalFileName.substring(i);
        }

        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd_HH-mm-ss");

        String formattedDate = formatter.format(now);
        String safeTitle = title.replaceAll("[^a-zA-Z0-9-_.]", "_");
        String fileName = safeTitle + "_" + formattedDate + extension;

        File destinationFile = new File(targetDir, fileName);

        System.out.println("Uploading to: " + destinationFile.getAbsolutePath());

        Files.copy(file.toPath(),
                    destinationFile.toPath(),
                    StandardCopyOption.REPLACE_EXISTING);

        System.out.println("File successfully copied.");

```

```

        return "upload/" + fileName;

    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}

public Document addDocument(String title, File file, long tagId, String author, long
yearPublished) throws IOException {
    long documentId =
counterLocalService.increment(Document.class.getName());

    Document doc = super.createDocument(documentId);
    doc.setTitle(title);
    doc.setAuthor(author);
    doc.setTagId(tagId);
    doc.setYearPublished(yearPublished);
    Date now = new Date();
    doc.setCreateDate(now);
    doc.setModifiedDate(now);
    String address = fileHandle(file, title, now);
    doc.setAddress(address);
    return super.addDocument(doc);
}

private boolean deleteFile(String filePath) {
    File file = new File(filePath);
    return file.delete();
}

```

```
public Document editDocument(long documentId, String title, File file, long tagId,
String author, long yearPublished) throws PortalException, IOException {
```

```
    Document doc = super.getDocument(documentId);
```

```
    doc.setTitle(title);
```

```
    doc.setAuthor(author);
```

```
    doc.setTagId(tagId);
```

```
    doc.setYearPublished(yearPublished);
```

```
    Date now = new Date();
```

```
    doc.setModifiedDate(now);
```

```
    if (file != null && file.exists()) {
```

```
        if (deleteFile(doc.getAddress()) == false) {
```

```
            throw new PortalException();
```

```
        }
```

```
        String address = fileHandle(file, title, now);
```

```
        doc.setAddress(address);
```

```
    }
```

```
    return super.updateDocument(doc);
```

```
}
```

```
public List<DocumentIndexViewModel> getDocumentViewModel(int start, int end)
throws PortalException {
```

```
    List<Document> listDoc = super.getDocuments(start, end);
```

```
    HashMap<Long, String> tagDiction = new HashMap<Long, String>();
```

```
    List<DocumentIndexViewModel> result = new
ArrayList<DocumentIndexViewModel>();
```

```

        for (Document doc: listDoc) {
            DocumentIndexViewModel docView = new
DocumentIndexViewModel();

            docView.setDocumentId(doc.getDocumentId());
            docView.setTagId(doc.getTagId());
            docView.setAddress(doc.getAddress());

            if (!tagDiction.containsKey(doc.getTagId())) {
                String tagName =
_tagLocalService.getTag(docView.getTagId()).getTagName();
                tagDiction.put(docView.getTagId(), tagName);
            }
            docView.setTagName(tagDiction.get(docView.getTagId()));
            docView.setAuthor(doc.getAuthor());
            docView.setCreateDate(doc.getCreateDate());
            docView.setModifiedDate(doc.getModifiedDate());
            docView.setYearPublished(doc.getYearPublished());
            docView.setTitle(doc.getTitle());
            result.add(docView);
        }
        return result;
    }
}

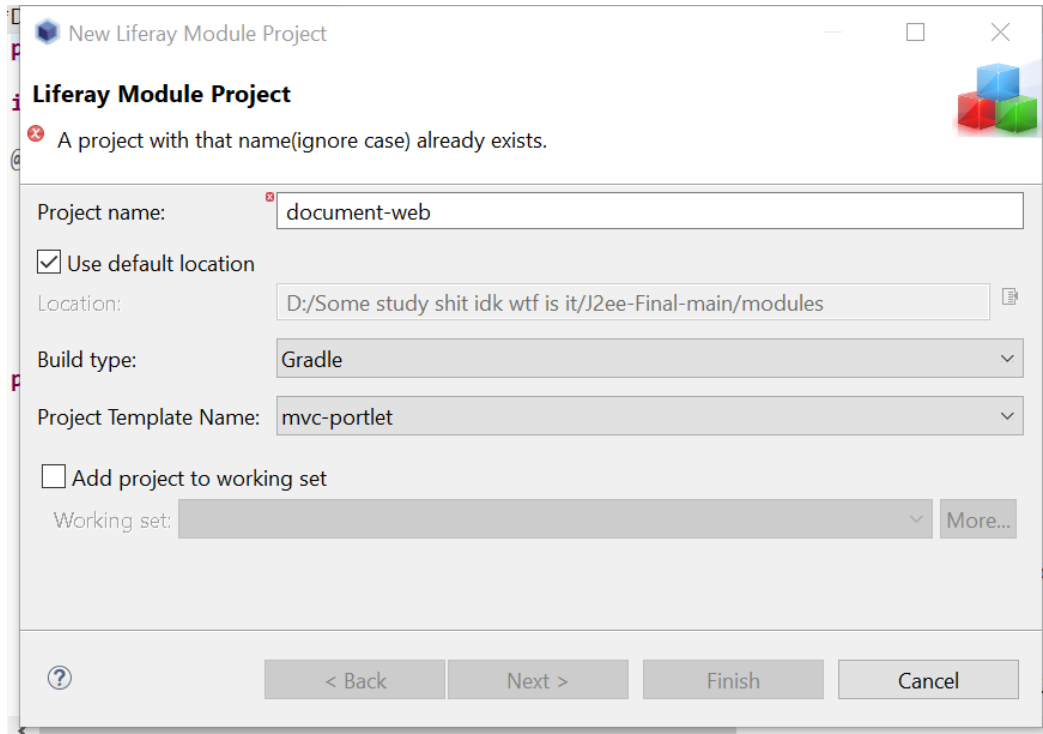
```

- Lại thực hiện cái bước xây dựng Service và cập nhật Workspace tương tự trước đó. Như vậy đã hoàn thành xây dựng Service cho ứng dụng.

5.2.3 Tạo Module Portlet

Các bước tạo Module:

- Tạo module mới với template là mvc-portlet tương tự như module Service ở trên. Sau đó nhấn Next



- Điền thông tin về tên Class và tên package, sau đó nhấn Finish
- Trong package <tên package>.constants, cập nhật class bằng đoạn code sau

package *com.document.web.constants;*

*/***

** @author Vivobook*

**/*

public class *DocumentWebPortletKeys* {

public static final String ***DOCUMENTWEB*** =

"com_document_web_DocumentWebPortlet";

}

- Các bước tiếp theo ta tập trung ở 2 directory là com.document.web.portlet.action và com.document.web.portlet.render:
- Tạo các class sau ở com.document.web.portlet.action và thêm vào các code như sau:

- **DocumentCreateActionCommand.java**

```
package com.document.web.portlet.action;
```

```
import com.document.data.service.DocumentLocalService;
```

```
import com.document.web.constants.DocumentWebPortletKeys;
```

```
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;
```

```
import com.liferay.portal.kernel.upload.UploadPortletRequest;
```

```
import com.liferay.portal.kernel.util.ParamUtil;
```

```
import com.liferay.portal.kernel.util.PortalUtil;
```

```
import java.io.File;
```

```
import javax.portlet.ActionRequest;
```

```
import javax.portlet.ActionResponse;
```

```
import javax.portlet.PortletException;
```

```
import org.osgi.service.component.annotations.Component;
```

```
import org.osgi.service.component.annotations.Reference;
```

```
@Component(
```

```
    immediate = true,
```

```
    property = {
```

```
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
```

```
        "mvc.command.name=/document/create"
```

```

    },
    service = MVCActionCommand.class
)

public class DocumentCreateActionCommand implements MVCActionCommand {

    @Reference
    private DocumentLocalService _documentLocalService;

    @Override
    public boolean processAction(ActionRequest actionRequest, ActionResponse actionResponse)
    throws PortletException {

        String title = ParamUtil.getString(actionRequest, "title");
        String author = ParamUtil.getString(actionRequest, "author");
        long yearPublished = ParamUtil.getLong(actionRequest, "yearPublished");
        long tagId = ParamUtil.getLong(actionRequest, "tagId");

        UploadPortletRequest uploadRequest = PortalUtil.getUploadPortletRequest(actionRequest);

        File uploadedFile = uploadRequest.getFile("documentFile");

        try {
            _documentLocalService.addDocument(title, uploadedFile, tagId, author, yearPublished);
        } catch (Exception e) {
            e.printStackTrace();
        }

        actionResponse.setRenderParameter("mvcRenderCommandName", "/document/index");
    }
}

```

```

        return true;
    }
}

```

- DocumentDeleteActionCommand.java

```

package com.document.web.portlet.action;

import com.document.data.service.DocumentLocalService;
import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;
import com.liferay.portal.kernel.util.ParamUtil;

import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.PortletException;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    immediate = true,
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/document/delete"
    },
    service = MVCActionCommand.class
)

```



```

public class DocumentDeleteActionCommand implements MVCActionCommand{

    @Reference

    private DocumentLocalService _documentLocalService;

    @Override

    public boolean processAction(ActionRequest actionRequest, ActionResponse
actionResponse)

        throws PortletException {

        long documentId = ParamUtil.getLong(actionRequest, "documentId");

        try {

            _documentLocalService.deleteDocument(documentId);

        } catch (Exception e)

        {

            e.printStackTrace();

        }

        actionResponse.setRenderParameter("mvcRenderCommandName",
"/document/index");

        return true;

    }

}

```

- DocumentEditActionCommand.java

```

package com.document.web.portlet.action;

import com.document.data.service.DocumentLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;
import com.liferay.portal.kernel.upload.UploadPortletRequest;

```

```

import com.liferay.portal.kernel.util.ParamUtil;
import com.liferay.portal.kernel.util.PortalUtil;

import java.io.File;

import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.PortletException;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    immediate = true,
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/document/edit"
    },
    service = MVCActionCommand.class
)
public class DocumentEditActionCommand implements MVCActionCommand {

    @Reference
    private DocumentLocalService _documentLocalService;

    @Override

```

```

    public boolean processAction(ActionRequest actionRequest, ActionResponse actionResponse)
    throws PortletException {

        long documentId = ParamUtil.getLong(actionRequest, "documentId");

        String title = ParamUtil.getString(actionRequest, "title");

        String author = ParamUtil.getString(actionRequest, "author");

        long yearPublished = ParamUtil.getLong(actionRequest, "yearPublished");

        long tagId = ParamUtil.getLong(actionRequest, "tagId");

        UploadPortletRequest uploadRequest = PortalUtil.getUploadPortletRequest(actionRequest);

        File uploadedFile = uploadRequest.getFile("documentFile");

        try {

            _documentLocalService.editDocument(documentId , title, uploadedFile, tagId, author,
            yearPublished);

        } catch (Exception e) {

            e.printStackTrace();

        }

        actionResponse.setRenderParameter("mvcRenderCommandName", "/document/index");

        return true;

    }
}

```

- TagCreateActionCommand.java

```

package com.document.web.portlet.action;

import com.document.data.service.TagLocalService;

import com.document.web.constants.DocumentWebPortletKeys;

import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;

```

```

import com.liferay.portal.kernel.util.ParamUtil;

import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.PortletException;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    immediate = true,
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/tag/create"
    },
    service = MVCActionCommand.class
)

public class TagCreateActionCommand implements MVCActionCommand{

    @Reference
    private TagLocalService _tagLocalService;

    @Override
    public boolean processAction(ActionRequest actionRequest, ActionResponse actionResponse)
        throws PortletException {
        String tagName = ParamUtil.getString(actionRequest, "tagName");
        try {
            _tagLocalService.addTag(tagName);
        } catch (Exception e)

```

```

    {
        e.printStackTrace();
    }

    actionResponse.setRenderParameter("mvcRenderCommandName", "/tag/index");
    return true;
}

}

```

- TagDeleteActionCommand.java

```

package com.document.web.portlet.action;

import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;
import com.liferay.portal.kernel.util.ParamUtil;

import javax.portlet.ActionRequest;
import javax.portlet.ActionResponse;
import javax.portlet.PortletException;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    immediate = true,
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/tag/delete"
    }
)

```

```

    },
    service = MVCActionCommand.class
)

public class TagDeleteActionCommand implements MVCActionCommand{

    @Reference
    private TagLocalService _tagLocalService;

    @Override
    public boolean processAction(ActionRequest actionRequest, ActionResponse actionResponse)
        throws PortletException {
        long tagId = ParamUtil.getLong(actionRequest, "tagId");

        try {
            _tagLocalService.deleteTag(tagId);
        } catch (Exception e)
        {
            e.printStackTrace();
        }

        actionResponse.setRenderParameter("mvcRenderCommandName", "/tag/index");
        return true;
    }
}

```

- TagEditActionCommand.java

```

package com.document.web.portlet.action;

import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;

```

```
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;  
import com.liferay.portal.kernel.util.ParamUtil;
```

```
import javax.portlet.ActionRequest;  
import javax.portlet.ActionResponse;  
import javax.portlet.PortletException;
```

```
import org.osgi.service.component.annotations.Component;  
import org.osgi.service.component.annotations.Reference;
```

```
@Component(  
    immediate = true,  
    property = {  
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,  
        "mvc.command.name=/tag/edit"  
    },  
    service = MVCActionCommand.class  
)
```

```
public class TagEditActionCommand implements MVCActionCommand{
```

```
    @Reference
```

```
    private TagLocalService _tagLocalService;
```

```
    @Override
```

```
    public boolean processAction(ActionRequest actionRequest, ActionResponse actionResponse)
```

```
        throws PortletException {
```

```
        String tagName = ParamUtil.getString(actionRequest, "tagName");
```

```
        long tagId = ParamUtil.getLong(actionRequest, "tagId");
```

```
        try {
```

```

        _tagLocalService.editTag(tagId, tagName);
    } catch (Exception e)
    {
        e.printStackTrace();
    }
    actionResponse.setRenderParameter("mvcRenderCommandName", "/tag/index");
    return true;
}
}

```

- Tạo các class sau ở com.document.web.portlet.render và thêm vào các code như sau:

- **DocumentCreateRenderCommand.java**

```

package com.document.web.portlet.render;

import com.document.data.model.Tag;
import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;

import java.util.List;

import javax.portlet.PortletException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

```



```

@Component(
    property = {
        "javax.portlet.name="+ DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/document/create"
    },
    service = MVCRenderCommand.class
)

public class DocumentCreateRenderCommand implements MVCRenderCommand {

    @Reference
    private TagLocalService _tagLocalService;

    @Override
    public String render(RenderRequest renderRequest, RenderResponse renderResponse)
    throws PortletException {
        List<Tag> tags = _tagLocalService.getAllTag();

        renderRequest.setAttribute("tagList", tags);

        return "/document/create.jsp";
    }
}

```

- **DocumentEditRenderCommand.java**

```

package com.document.web.portlet.render;

import com.document.data.model.Document;
import com.document.data.model.Tag;
import com.document.data.service.DocumentLocalService;

```

```

import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.exception.PortalException;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;
import com.liferay.portal.kernel.util.ParamUtil;

import java.util.List;

import javax.portlet.PortletException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/document/edit"
    },
    service = MVCRenderCommand.class
)
public class DocumentEditRenderCommand implements MVCRenderCommand {

    @Reference
    private TagLocalService _tagLocalService;

    @Reference

```

```

private DocumentLocalService _documentLocalService;

@Override
public String render(RenderRequest renderRequest, RenderResponse renderResponse)
throws PortletException {

    long documentId = ParamUtil.getLong(renderRequest, "documentId");

    try {

        Document doc = _documentLocalService.getDocument(documentId);
        List<Tag> tags = _tagLocalService.getAllTag();

        renderRequest.setAttribute("tagList", tags);
        renderRequest.setAttribute("document", doc);
    } catch (PortalException e) {

        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return "/document/edit.jsp";
}
}

```

- DocumentIndexRenderCommand.java

```

package com.document.web.portlet.render;

import com.document.data.service.DocumentLocalService;

```

```

import com.liferay.portal.kernel.dao.search.SearchContainer;
import com.liferay.portal.kernel.exception.PortalException;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;
import com.liferay.portal.kernel.util.ParamUtil;

import java.util.ArrayList;
import java.util.List;

import javax.portlet.ActionRequest;
import javax.portlet.PortletException;
import javax.portlet.PortletURL;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;
import com.document.data.viewmodel.DocumentIndexViewModel;
import com.document.web.constants.DocumentWebPortletKeys;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
    property = {
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
        "mvc.command.name=/document/index"
    },
    service = MVCRenderCommand.class
)

public class DocumentIndexRenderCommand implements MVCRenderCommand {

    @Reference
    private DocumentLocalService _documentLocalService;

```

```

@Override

    public String render(RenderRequest renderRequest, RenderResponse renderResponse)
        throws PortletException {

        int cur = ParamUtil.getInteger(renderRequest, SearchContainer.DEFAULT_CUR_PARAM,
1);

        int          delta          =          ParamUtil.getInteger(renderRequest,
SearchContainer.DEFAULT_DELTA_PARAM, 5);

        int start = (cur - 1) * delta;

        int end = start + delta;


        List<DocumentIndexViewModel>          documents          =          new
ArrayList<DocumentIndexViewModel>();

        try {

            documents = _documentLocalService.getDocumentViewModel(start, end);

        } catch (PortalException e) {

            // TODO Auto-generated catch block

            e.printStackTrace();

        }

        int total = _documentLocalService.getDocumentsCount();

        renderRequest.setAttribute("documentList", documents);

        renderRequest.setAttribute("totalCount", total);


        PortletURL iteratorURL = renderResponse.createRenderURL();

        iteratorURL.setParameter("mvcRenderCommandName", "/document/index");

        renderRequest.setAttribute("iteratorURL", iteratorURL);

```

```

        PortletURL editURL = renderResponse.createRenderURL();
        editURL.setParameter("mvcRenderCommandName", "/document/edit");
        renderRequest.setAttribute("editURL", editURL.toString());

        PortletURL deleteURL = renderResponse.createActionURL();
        deleteURL.setParameter(ActionRequest.ACTION_NAME, "/document/delete");
        renderRequest.setAttribute("deleteActionURL", deleteURL.toString());

        return "/document/index.jsp";
    }
}

```

- TagCreateRenderCommand.java

```

package com.document.web.portlet.render;

import com.document.data.model.Tag;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;

import java.util.List;

import javax.portlet.PortletException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;

import org.osgi.service.component.annotations.Component;

@Component(

```

```

        immediate = true,
        property = {
            "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
            "mvc.command.name=/tag/create"
        },
        service = MVCRenderCommand.class
    )
}

public class TagCreateRenderCommand implements MVCRenderCommand{
    @Override
    public String render(RenderRequest renderRequest, RenderResponse renderResponse) throws
    PortletException {
        return "/tag/create.jsp";
    }
}

```

- TagEditRenderCommand.java

```

package com.document.web.portlet.render;

import com.document.data.model.Tag;
import com.document.data.service.TagLocalService;
import com.document.web.constants.DocumentWebPortletKeys;
import com.liferay.portal.kernel.exception.PortalException;
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;
import com.liferay.portal.kernel.util.ParamUtil;

import java.util.List;

import javax.portlet.PortletException;
import javax.portlet.RenderRequest;

```

```
import javax.portlet.RenderResponse;
```

```
import org.osgi.service.component.annotations.Component;
```

```
import org.osgi.service.component.annotations.Reference;
```

```
@Component(
```

```
    immediate = true,
```

```
    property = {
```

```
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
```

```
        "mvc.command.name=/tag/edit"
```

```
    },
```

```
    service = MVCRenderCommand.class
```

```
)
```

```
public class TagEditRenderCommand implements MVCRenderCommand{
```

```
    @Reference
```

```
    private TagLocalService _tagLocalService;
```

```
    @Override
```

```
    public String render(RenderRequest renderRequest, RenderResponse renderResponse) throws  
PortletException {
```

```
        long tagId = ParamUtil.getLong(renderRequest, "tagId");
```

```
        try {
```

```
            Tag foundTag = _tagLocalService.getTag(tagId);
```

```
            renderRequest.setAttribute("tag", foundTag);
```

```
        } catch (PortalException e) {
```

```
            // TODO Auto-generated catch block
```

```
            e.printStackTrace();
```

```
        }
```

```
        return "/tag/edit.jsp";
```



```
}  
}
```

- TagIndexRenderCommand.java

```
package com.document.web.portlet.render;
```

```
import com.liferay.portal.kernel.portlet.bridges.mvc.MVCRenderCommand;
```

```
import java.util.List;
```

```
import javax.portlet.PortletException;
```

```
import javax.portlet.RenderRequest;
```

```
import javax.portlet.RenderResponse;
```

```
import com.document.web.constants.DocumentWebPortletKeys;
```

```
import org.osgi.service.component.annotations.Component;
```

```
import org.osgi.service.component.annotations.Reference;
```

```
import com.document.data.model.Tag;
```

```
import com.document.data.service.TagLocalService;
```

```
@Component(
```

```
    immediate = true,
```

```
    property = {
```

```
        "javax.portlet.name=" + DocumentWebPortletKeys.DOCUMENTWEB,
```

```
        "mvc.command.name=/tag/index"
```

```
    },
```

```
    service = MVCRenderCommand.class
```

```
)
```

```
public class TagIndexRenderCommand implements MVCRenderCommand{
```

@Reference

```
private TagLocalService _tagLocalService;
```

@Override

```
public String render(RenderRequest renderRequest, RenderResponse renderResponse) throws  
PortletException {
```

```
    List<Tag> tagList = _tagLocalService.getAllTag();
```

```
    renderRequest.setAttribute("tagList", tagList);
```

```
    return "/tag/index.jsp";
```

```
}
```

```
}
```

- Còn 1 số code có thể tham khảo tại file gốc, tiếp theo chúng ta sẽ tập trung vào các JSP trong src/main/resources/META-INF/resources
- Trong mục document tạo các file jsp sau và chép code vào:

- **Create.jsp**

```
• <%@ page contentType="text/html; charset=UTF-8" %>  
•  
• <%@ include file="../init.jsp" %>  
•  
• <portlet:actionURL name="/document/create" var="documentCreateURL" />  
•  
• <div class="m-4">  
•     <au:form action="${documentCreateURL}" method="post"  
•     enctype="multipart/form-data">  
•         <au:input name="title" label="Title" type="text" required="true" />  
•         <au:input name="author" label="Author" type="text" required="true" />  
•         <au:input name="yearPublished" label="Year Published" type="number"  
•         required="true" />  
•  
•         <au:select name="tagId" label="Select Tag" required="true">  
•             <au:option value="" label="-- Select Tag --" />  
•             <c:forEach var="tag" items="${tagList}">  
•                 <au:option value="${tag.tagId}" label="${tag.tagName}" />  
•             </c:forEach>  
•         </au:select>  
•  
•         <au:input name="documentFile" label="Upload File" type="file"  
•         required="true" />  
•  
•         <au:button type="submit" value="Add Document" />  
•     </au:form>
```

- `</div>`
-
- **Edit.jsp**
- `<%@ page contentType="text/html; charset=UTF-8" %>`
-
- `<%@ include file="../../init.jsp" %>`
-
- `<portlet:actionURL name="/document/edit" var="documentEditURL" />`
-
- `<div class="m-4">`
- `<auiform action="${documentEditURL}" method="post"`
- `enctype="multipart/form-data">`
- `<auiformput name="documentId" type="hidden" value`
- `= "${document.documentId}" />`
- `<auiformput name="title" label="Title" value="${document.title}"`
- `type="text" required="true" />`
- `<auiformput name="author" label="Author" value="${document.author}"`
- `type="text" required="true" />`
- `<auiformput name="yearPublished" label="Year Published"`
- `value="${document.yearPublished}" type="number" required="true" />`
-
- `<auiformselect name="tagId" label="Select Tag" required="true"`
- `value="${document.tagId}">`
- `<auiformoption value="" label="-- Select Tag --" />`
- `<c:forEach var="tag" items="${tagList}">`
- `<auiformoption value="${tag.tagId}" label="${tag.tagName}" />`
- `</c:forEach>`
- `</auiformselect>`
-
- `<auiforminput name="documentFile" label="Upload File" type="file"`
- `required="false" />`
-
- `<auiformbutton type="submit" value="Add Document" />`
- `</auiform>`
- `</div>`
-
- **Index.jsp**
- `<%@ page contentType="text/html; charset=UTF-8" %>`
-
- `<%@ include file="../../init.jsp" %>`
- `<%@ page import="com.document.data.viewmodel.DocumentIndexViewModel" %>`
- `<liferay-portlet:renderURL var="documentCreateURL">`
- `<portlet:param name="mvcRenderCommandName" value="/document/create" />`
- `</liferay-portlet:renderURL>`
- `<div class="m-4">`
- `Add new`
- `Document`
- ---

- `<liferay-ui:search-container`
- `total="${totalCount}"`
- `delta="5"`

```

• emptyResultsMessage="No documents found"
• iteratorURL="{iteratorURL}">
•
• <liferay-ui:search-container-results results="{documentList}" />
•
• <liferay-ui:search-container-row
•   className="com.document.data.viewmodel.DocumentIndexViewModel"
•   modelVar="document">
•
•   <liferay-ui:search-container-column-text
•     name="Title"
•     value="{document.title}" />
•
•   <liferay-ui:search-container-column-text
•     name="Tag"
•     value="{document.tagName}" />
•
•   <liferay-ui:search-container-column-text
•     name="Author"
•     value="{document.author}" />
•
•   <liferay-ui:search-container-column-text
•     name="Year"
•     value="{document.yearPublished}" />
•
•   <liferay-ui:search-container-column-text name="Actions">
•     <div class="d-flex align-items-center gap-
2">
•
•       <portlet:renderURL
var="documentEditURL">
•
•       <portlet:param
name="mvcRenderCommandName" value="/document/edit" />
•
•       <portlet:param
name="documentId" value="{document.documentId}" />
•
•       </portlet:renderURL>
•       <portlet:actionURL
var="deleteDocumentURL" name="/document/delete">
•
•       <portlet:param
name="documentId" value="{document.documentId}" />
•
•       </portlet:actionURL>
•
•
•       <a href="{documentEditURL}"
class="btn btn-sm btn-warning" style="margin-right:0.25rem">Edit</a>
•
•
•       <form method="post"
action="{deleteDocumentURL}" onsubmit="return confirm('Are you sure to
delete?')" class="m-0 p-0 d-inline">
•
•       <button type="submit"
class="btn btn-sm btn-danger">Delete</button>
•
•       </form>
•     </div>

```

- `</liferay-ui:search-container-column-text>`
- `</liferay-ui:search-container-row>`
-
- `<liferay-ui:search-iterator />`
- `</liferay-ui:search-container>`
- `</div>`

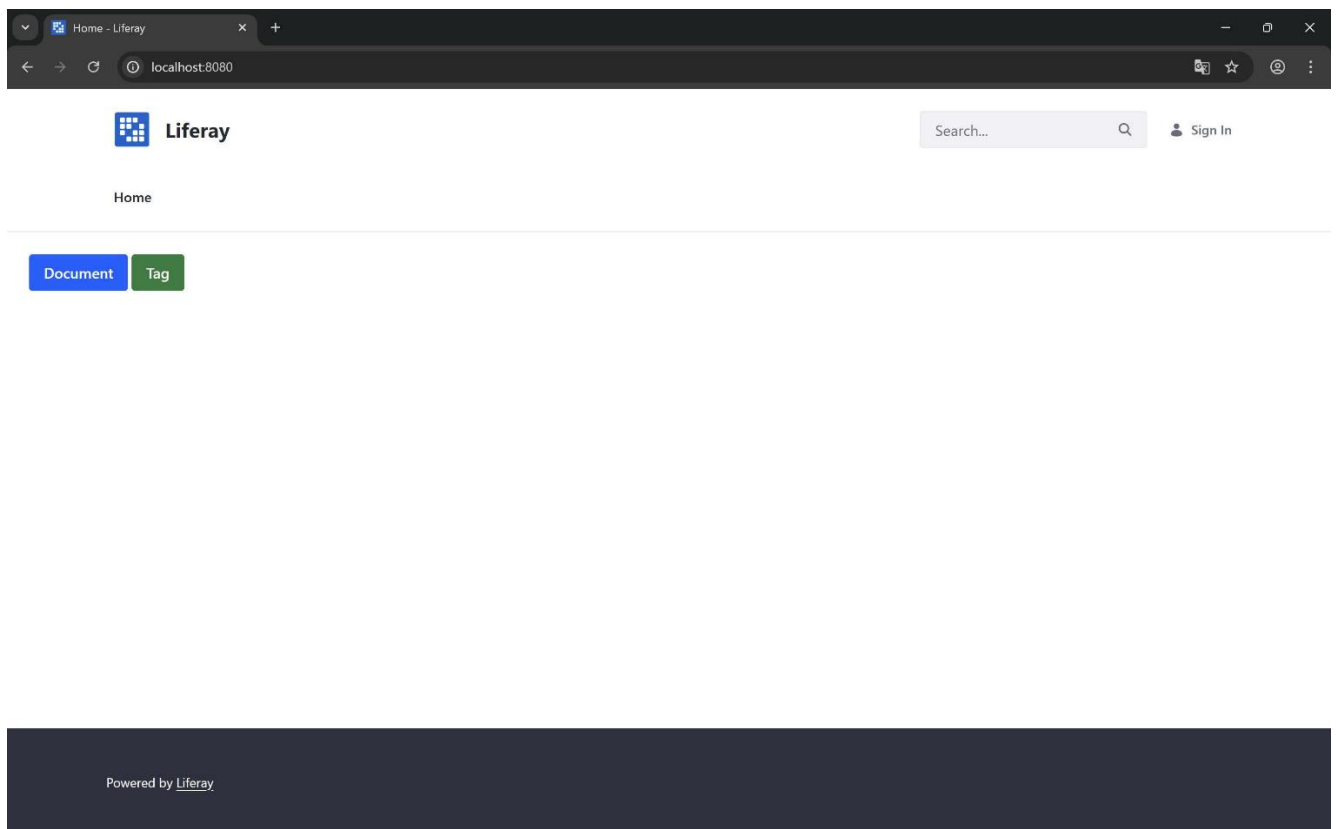
5.2.4 Triển khai công nghệ này lên server Liferay

- Kéo thả các module `<module portlet>`, `<module service>service` và `<module service>-api` vào server ở trong tab Server.
- Khởi chạy Server. Kiểm tra các modules đã được triển khai thành công bằng cách đăng nhập quản trị viên và kiểm tra trong Edit > Fragments and Widgets > Widgets > Sample
- Đến trang Pages, lần lượt tạo 2 trang nội dung trống mới là Document, Tag, sau đó thêm widget này vào trong các trang.

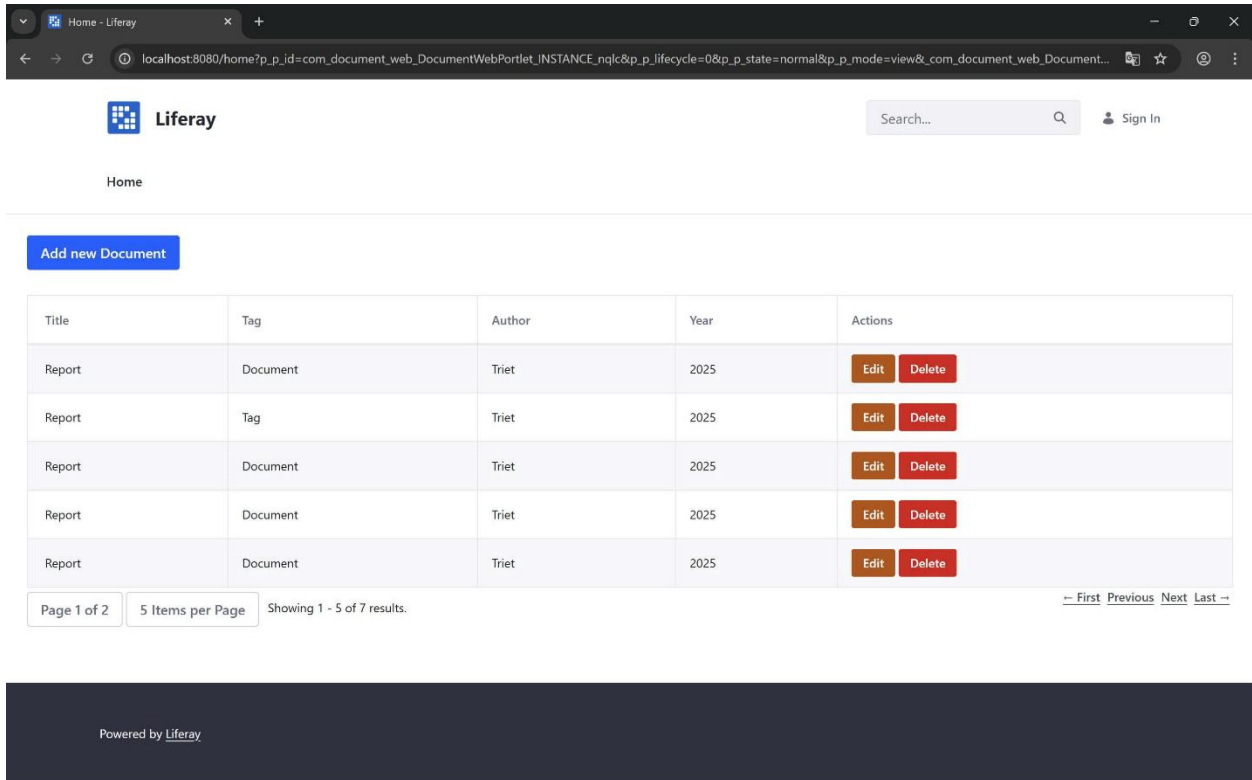
6. Kết quả:

Giao diện ứng dụng:

- Giao diện /home:



Giao diện /document/index:



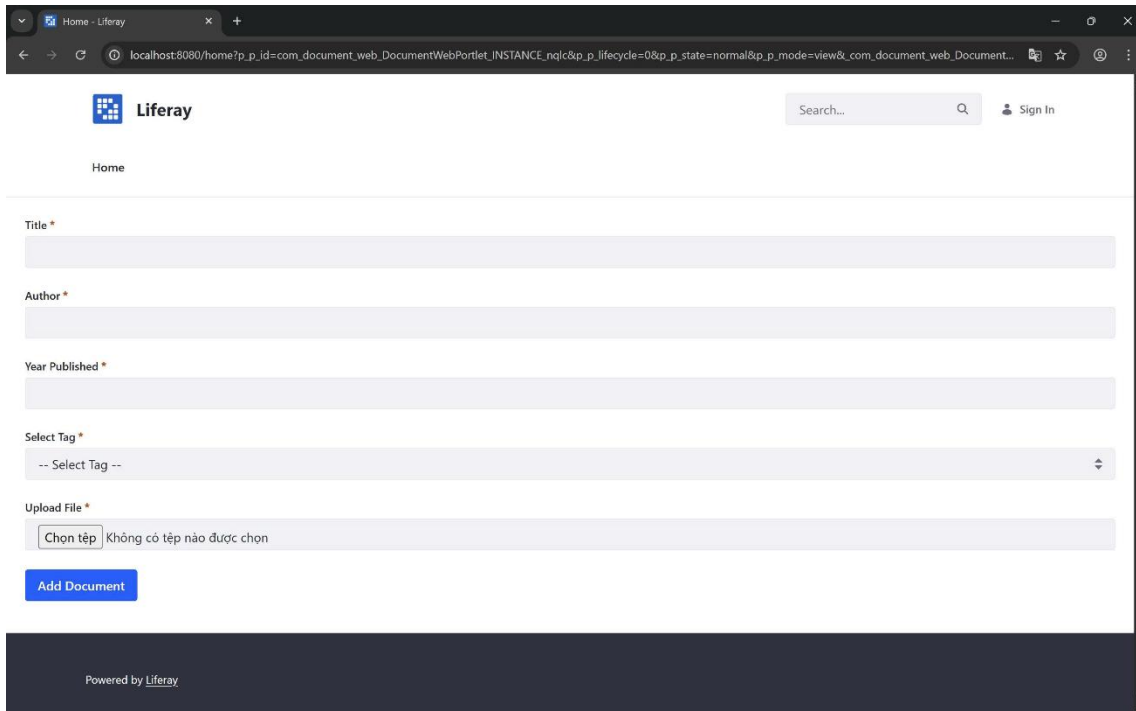
The screenshot shows the Liferay web interface for the /document/index page. The browser address bar displays the URL: localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqlc&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document... The page header includes the Liferay logo, a search bar, and a "Sign In" link. Below the header, there is a "Home" link and a blue button labeled "Add new Document". The main content area features a table with the following columns: Title, Tag, Author, Year, and Actions. The table contains five rows of data, each with a "Report" title, a tag (Document or Tag), the author "Triet", and the year "2025". Each row has "Edit" and "Delete" buttons in the Actions column. Below the table, there is a pagination bar showing "Page 1 of 2", "5 Items per Page", and "Showing 1 - 5 of 7 results". At the bottom of the page, there is a dark footer with the text "Powered by Liferay".

Title	Tag	Author	Year	Actions
Report	Document	Triet	2025	Edit Delete
Report	Tag	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete

Page 1 of 2 5 Items per Page Showing 1 - 5 of 7 results [First](#) [Previous](#) [Next](#) [Last](#)

Powered by Liferay

Giao diện /document/create:



The screenshot shows the Liferay web interface for the /document/create page. The browser address bar displays the URL: localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqlc&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document... The page header includes the Liferay logo, a search bar, and a "Sign In" link. Below the header, there is a "Home" link. The main content area contains a form with the following fields: "Title *" (text input), "Author *" (text input), "Year Published *" (text input), "Select Tag *" (dropdown menu with "-- Select Tag --"), and "Upload File *" (file upload button labeled "Chọn tệp" and a message "Không có tệp nào được chọn"). Below the form, there is a blue button labeled "Add Document". At the bottom of the page, there is a dark footer with the text "Powered by Liferay".

Title *

Author *

Year Published *

Select Tag *

-- Select Tag --

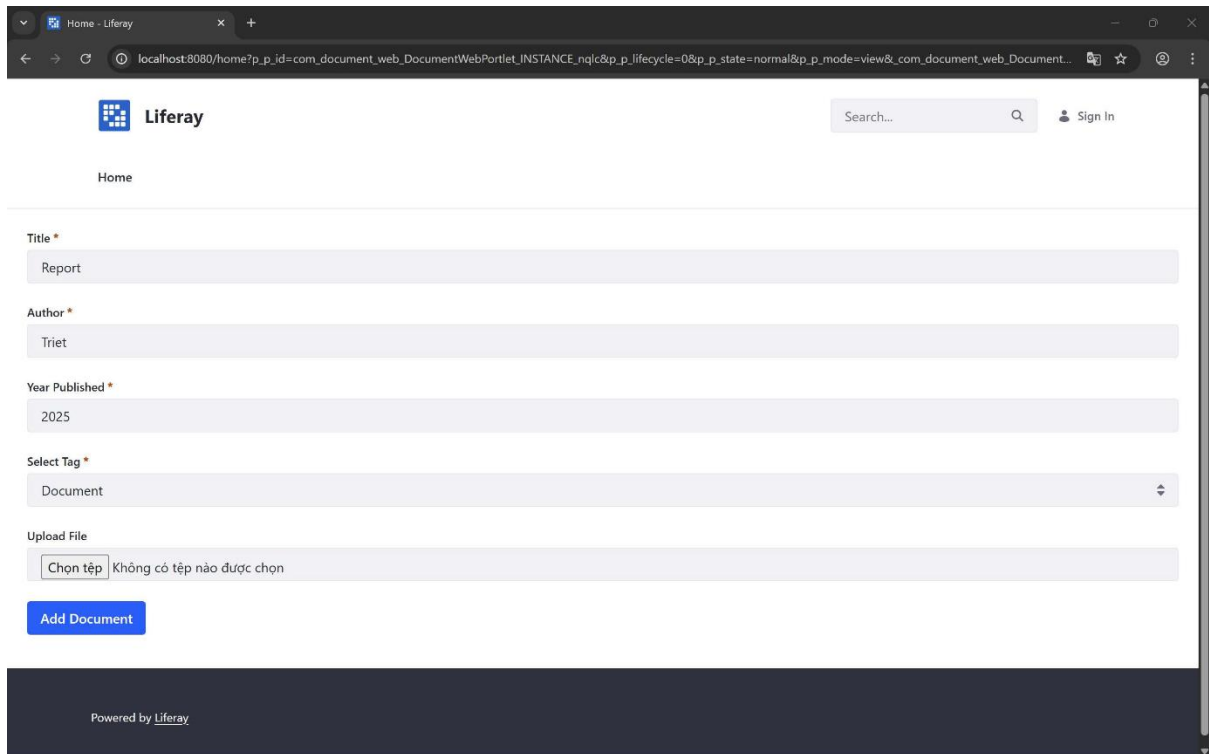
Upload File *

Chọn tệp Không có tệp nào được chọn

Add Document

Powered by Liferay

Giao diện /document/edit/



The screenshot shows the Liferay web portal interface for editing a document. The browser address bar displays the URL: localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqic&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document... The page header includes the Liferay logo, a search bar, and a 'Sign In' link. Below the header, the page is titled 'Home'. The main content area contains several form fields: 'Title *' with the value 'Report', 'Author *' with the value 'Triet', 'Year Published *' with the value '2025', and 'Select Tag *' with the value 'Document'. There is also an 'Upload File' section with a 'Chọn tệp' button and a message 'Không có tệp nào được chọn'. At the bottom of the form is a blue 'Add Document' button. The footer of the page states 'Powered by Liferay'.

Home

Title *

Report

Author *

Triet

Year Published *

2025

Select Tag *

Document

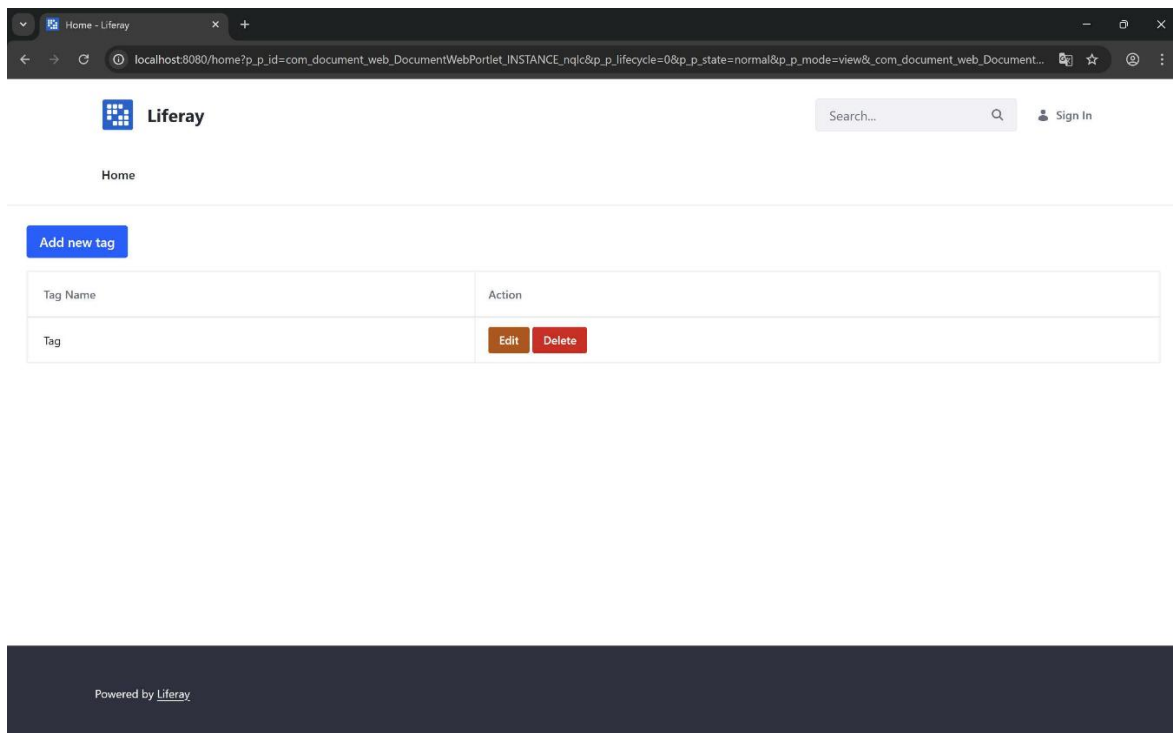
Upload File

Chọn tệp Không có tệp nào được chọn

Add Document

Powered by Liferay

Giao diện /tag/index:



The screenshot shows the Liferay web portal interface for managing tags. The browser address bar displays the URL: localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqic&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document... The page header includes the Liferay logo, a search bar, and a 'Sign In' link. Below the header, the page is titled 'Home'. The main content area features a blue 'Add new tag' button and a table with two columns: 'Tag Name' and 'Action'. The table contains one row with the value 'Tag' in the 'Tag Name' column and 'Edit' and 'Delete' buttons in the 'Action' column. The footer of the page states 'Powered by Liferay'.

Home

Add new tag

Tag Name	Action
Tag	Edit Delete

Powered by Liferay

Giao diện /tag/create:

The screenshot shows a web browser window with the Liferay portal. The address bar shows a URL with parameters. The page header includes the Liferay logo, a search bar, and a 'Sign In' link. Below the header, there is a 'Home' link. The main content area is titled 'Add new tag' and contains a form with a 'Tag Name' label and a text input field. Below the input field is a blue 'Add Tag' button. At the bottom of the page, there is a dark footer with the text 'Powered by Liferay'.

Home - Liferay

localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqlc&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document...

Liferay

Search...

Sign In

Home

Add new tag

Tag Name *

Add Tag

Powered by Liferay

Giao diện /document/delete:

The screenshot shows a web browser window with the Liferay portal. A confirmation dialog is displayed in the center, asking 'Are you sure to delete?' with 'OK' and 'Hủy' buttons. The background is dimmed, showing the 'Add new Document' button and a table of documents. The table has columns for Title, Tag, Author, Year, and Actions. The footer shows 'Powered by Liferay'.

Home - Liferay

localhost:8080/home?p_p_id=com_document_web_DocumentWebPortlet_INSTANCE_nqlc&p_p_lifecycle=0&p_p_state=normal&p_p_mode=view&com_document_web_Document...

Liferay

Search...

Sign In

Home

Add new Document

localhost:8080 cho biết
Are you sure to delete?

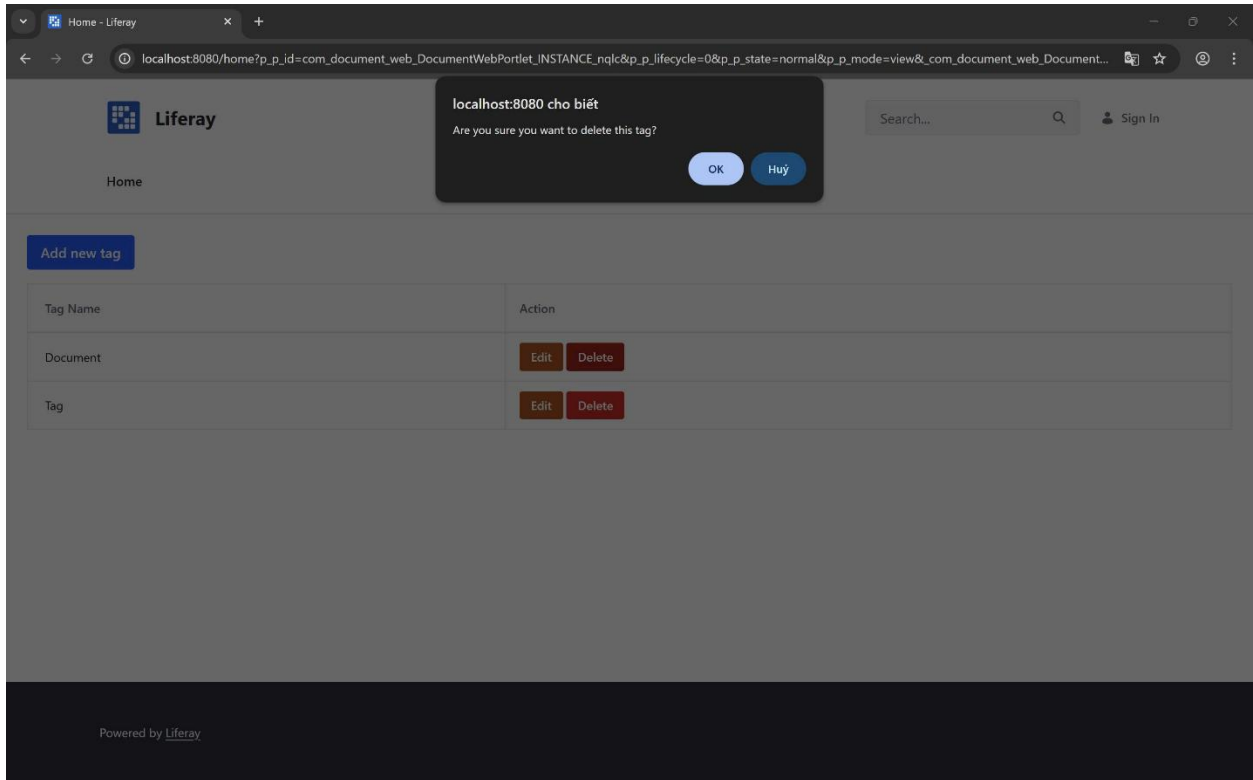
OK Hủy

Title	Tag	Author	Year	Actions
Report	Tag	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete
Report	Document	Triet	2025	Edit Delete

Page 1 of 2 5 Items per Page Showing 1 - 5 of 6 results. — First Previous Next Last —

Powered by Liferay

Giao diện /tag/delete:



7. Kết luận

7.1 Đánh giá:

- Dự án Ứng Dụng Quản Lý Tài liệu bằng Liferay đã thành công trong việc phát triển một hệ thống quản lý tài liệu hiệu quả và tương đối dễ sử dụng. Bằng cách sử dụng nền tảng Liferay, nhóm đã có thể xây dựng một ứng dụng với khả năng tùy biến cao, khả năng mở rộng tốt, giao diện người dùng tối giản, trực quan.
- Ứng dụng này đồng thời cũng đã đáp ứng được các mục tiêu đã đề ra từ ban đầu.

7.2 Những nâng cấp trong tương lai

Mặc dù ứng dụng đã đáp ứng được các nhu cầu cơ bản trong việc quản lý tài liệu, nhưng vẫn còn có những chức năng có thể nâng cấp trong tương lai như:

- Thêm các thông tin chi tiết.
- Phân quyền cho từng vai trò khác nhau.
- Thêm vào chức năng upload tài liệu.
- Tùy chỉnh giao diện.

7.3 Nhận định cuối cùng

Dự án đã đạt được những kết quả đáng khích lệ, mang lại nhiều kiến thức cũng như kinh nghiệm cho nhóm.