

React-based Lifeary Web Application and Upgrade of Liferay Project

Project report submitted in partial fulfillment of requirements

For the degree of

Master of Computer Application

of

Computer Science and Engineering Department

of

Jadavpur University

by

Md Shakir Shamim

Regn. No. - 154226 of 2020-2021

Exam Roll No. - MCA2360026

under the supervision of

Dr. Sarmistha Neogy Lahiri

Professor

Department of Computer Science and Engineering

JADAVPUR UNIVERSITY

Kolkata, West Bengal, India

2023

Certificate from the Supervisor

This is to certify that the work embodied in this report entitled "**React-based Lifeary Web Application and Upgrade of Liferay Project**" has been satisfactorily completed by **Md Shakir Shamim** (Registration Number 154226 of 2020 – 21; Class Roll No. 002010503018; Examination Roll No. *MCA2360026*). It is a bona-fide piece of work carried out under my supervision and guidance at Jadavpur University, Kolkata for partial fulfilment of the requirements for the awarding of the **Master of Computer Application** degree of the Department of Computer Science and Engineering, Faculty of Engineering and Technology, Jadavpur University, during the academic year 2022 – 23.

Dr. Sarmistha Neogy Lahiri,
Professor,
Department of Computer Science and Engineering,
Jadavpur University.
(Supervisor)

Forwarded By:

Prof. Nandini Mukherjee,
Head,
Department of Computer Science and Engineering,
Jadavpur University.

Prof. Ardhendu Ghoshal,
Dean,
Faculty of Engineering & Technology,
Jadavpur University.

Department of Computer Science and Engineering
Faculty of Engineering And Technology
Jadavpur University, Kolkata - 700 032

Certificate of Approval

This is to certify that the thesis entitled "**React-based Lifeary Web Application and Upgrade of Liferay Project**" is a bona-fide record of work carried out by **Md Shakir Shamim** (Registration Number 154226 of 2020 – 21; Class Roll No. 002010503018; Examination Roll No. *MCA2360026*) in partial fulfilment of the requirements for the award of the degree of **Master of Computer Application** in the **Department of Computer Science and Engineering, Jadavpur University**, during the period of January 2023 to May 2023. It is understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose of which it has been submitted.

Examiners:

(Signature of The Examiner)

(Signature of The Examiner)

ACKNOWLEDGEMENT

I am pleased to express my gratitude and regards towards my Project Guide **Dr.Sarmistha Neogy Lahiri**, Professor, Department of Computer Science and Engineering, Jadavpur University, without whose valuable guidance, inspiration and attention towards me, pursuing my project would have been impossible.

Last but not the least, I express my regards towards my friends and family for bearing with me and for being a source of constant motivation during the entire term of the work.

Md Shakir Shamim

MCA Final Year

Exam Roll No. - MCA2360026

Regn. No. - 154226 of 2020 – 21

Department of Computer Science and Engineering,
Jadavpur University.

Contents

1	Introduction	1
1.1	Why we need web application ?	1
1.2	What is liferay?	1
1.3	Web development using Liferay	2
2	Problem definition and some Liferay features	3
2.1	Problem definition	3
2.1.1	React based web Application performing CRUD operation on Blog Posting, Documents and Articles	3
2.1.2	Upgrade of Project	4
2.2	Some Out of The Box (OOTB) features of Liferay used	5
2.2.1	Service Builder	5
2.2.2	MVC Portlet Web Module	5
2.2.3	Liferay Headless REST API for Content Management System	5
3	Present Work on React-based Web Application	6
3.1	Introduction	6
3.2	Developing a React Application using Liferay	7
3.3	Setting Up Dependencies	7
3.3.1	Using the Liferay JS Generator	7
3.3.2	How to Build and Deploy	8
3.4	Creating a React Component	8
3.5	Liferay JavaScript Object	9
3.6	Calling APIs	9
3.6.1	Consuming REST Services	9
3.6.2	Calling APIs in react component	11
3.7	Using Clay Components	11
3.8	Experiment and Results	12
3.8.1	Navigation bar	12
3.8.2	Create/Add Entries	12
3.8.3	Read/Fetch, Update and Delete entries	14

4	Present Work on Upgrade project	17
4.1	Introduction	17
4.2	Why we need to Upgrade?	17
4.3	Approach	18
4.3.1	Database Upgrade	18
4.3.2	Code upgrade	20
4.4	Experiment and results	20
4.4.1	Web Application that is going to be updated	20
4.4.2	DataBase Upgrade	21
4.4.3	DataBase Upgrade	22
5	Conclusion	24

List of Figures

3.1	BlogPosting headless API	10
3.2	Document headless API	10
3.3	Knowledge Base Article headless API	11
3.4	Navigation bar	12
3.5	Adding Blog Posting	13
3.6	Adding Documents	13
3.7	Adding Knowledge Base Article	14
3.8	Fetching Blog Posting	15
3.9	Fetching Document	15
3.10	Fetching Knowledge Base Article	16
4.1	Basic Form a web application which collects user details and it is deployed on liferay	
6.2	20
4.2	Clicking on Add Entry redirect you to the Form page	21
4.3	By clicking saves after entering details it saves data in Database	21
4.4	During database upgrade	22
4.5	After code upgrade and deployment to 7.4 DXP	22

Abstract

This report explores the development of a React-based web application using Liferay as the backend platform and the process of upgrading a Liferay project from version 6.2 to 7.4. It discusses the advantages of using React for frontend development and how Liferay facilitates the integration of React components. The report highlights the utilization of Liferay's Headless APIs for managing CMS functionalities like blogs, documents, and knowledge base articles, enabling the creation of dynamic and interactive web applications. Additionally, it outlines the steps involved in the upgrade process, emphasizing the benefits of upgrading to Liferay 7.4, including enhanced features, improved performance, and advanced security measures.

Chapter 1

Introduction

This report provides a concise overview of web applications built using Liferay, a leading open-source enterprise portal platform. Web applications are online platforms that businesses and individuals use to provide services, engage with users, and deliver a dynamic online experience. Liferay offers a robust framework for developing and managing web applications, making it a popular choice for businesses seeking an all-in-one solution.

This report focuses on web applications built using Liferay and highlights the advantages of utilizing this platform. It explores the benefits of Liferay's scalability, flexibility, content management capabilities, security features, collaboration tools, integration capabilities, and support for mobile-friendly design.

By examining the benefits of building web applications with Liferay, this report aims to showcase the value of this platform in empowering businesses to create feature-rich and user-friendly online experiences. It serves as a valuable resource for businesses and developers interested in harnessing the capabilities of Liferay to build robust and engaging web applications.

1.1 Why we need web application ?

Web applications are crucial in today's digital world for several reasons. They allow businesses and individuals to reach a global audience and provide services conveniently. Web applications streamline processes, save time and resources, and can handle growth.

Web applications offer a multitude of benefits that make them invaluable in today's digital landscape. One of the key advantages is accessibility, as web applications can be accessed from any device with an internet connection, providing users with the flexibility to engage with services and information at their convenience. Additionally, web applications have a global reach, enabling businesses to connect with a vast audience across different geographic locations, expanding customer reach and unlocking potential market opportunities.

1.2 What is liferay?

Liferay is a versatile and flexible platform that enables businesses to build and manage their online presence effectively. It serves as a comprehensive solution for creating websites, portals, and other

web applications. With Liferay, organizations can easily customize and tailor their digital experiences to meet specific needs and engage with their target audience more effectively. The platform provides a user-friendly interface, robust features, and integration capabilities, allowing businesses to streamline internal processes, collaborate with teams, and deliver personalized content and services to users. Liferay simplifies the development and management of digital experiences, making it an ideal choice for businesses seeking to enhance their online presence.[8]

1.3 Web development using Liferay

Web development using Liferay involves utilizing its platform and tools to create interactive and dynamic web applications. Liferay offers a drag-and-drop interface, pre-built components, and customizable themes for easy and visually appealing web page creation. Its built-in content management system simplifies content organization, version control, and publication.

Liferay also provides extensive integration capabilities to connect with external systems and services. User authentication, permission management, and personalization features enhance security and create personalized user experiences. Overall, Liferay empowers developers to build feature-rich, visually appealing, and secure web applications.

Chapter 2

Problem definition and some Liferay features

2.1 Problem definition

2.1.1 React based web Application performing CRUD operation on Blog Posting, Documents and Articles

This React-based web applications within the Liferay platform. React is a popular JavaScript library used for building user interfaces. With the Liferay React Module, developers can leverage the power of React while taking advantage of Liferay's features, such as user authentication, content management, and workflow management.

In Liferay, blog postings, documents, and knowledge base articles are content types that can be managed and published using Liferay's content management system (CMS) capabilities

Headless APIs used in this application

Liferay also provides headless APIs (Application Programming Interfaces) that allow developers to access and interact with blog postings, documents, and knowledge base articles programmatically. These APIs enable developers to retrieve, create, update, and delete content from Liferay's CMS without directly accessing the administrative interface

By using Liferay's headless APIs, developers can build custom applications, integrations, or front-end interfaces that interact with Liferay's content management system. These APIs follow industry-standard protocols such as REST (Representational State Transfer) or GraphQL, making it easier to communicate with Liferay's CMS from external systems or applications.

User interface (UI) framework used in this Application

Clay is a user interface (UI) framework developed by Liferay that provides a library of reusable and customizable UI components. It offers a consistent and modern design system that helps developers quickly build visually appealing and responsive user interfaces for web applications

With Clay, developers can achieve a cohesive and consistent look across their web application, ensuring a seamless user experience. The framework is built using modern web technologies like HTML, CSS, and JavaScript, making it compatible with various browsers and devices

Deployment

This react application deployed in Liferay DXP (Digital Experience Platform). Liferay DXP 7.4 provides a robust and scalable environment for hosting and managing web applications, including those built using React.

NOTE : Details of this project is on Chapter 3

2.1.2 Upgrade of Project

This project more focused on upgrade of web application. Web application that made and deployed on liferay 6.2 that takes user details and saves in MYSQL database. This is going to upgrade into liferay 7.4.

Details of web application going to upgrade

Front-end or view section of this web application made by using MVC Portlet Web Module. And back-end service made using service builder of liferay. Using these service its going to communicate with database.

In Liferay 6.2, the concept of a Gradle workspace did not exist. Liferay 6.2 predominantly used Ant build scripts for managing and building projects.

A Gradle workspace is a concept introduced in later versions of Liferay, starting from Liferay 7 and onwards. It provides a unified build environment for managing multiple Liferay projects within a single workspace

The upgrade from Liferay 6.2, a portlet-based framework, to Liferay 7.4 DXP (Digital Experience Platform) marks a significant transition towards a more modern and enhanced web development environment. Liferay 7.4 DXP introduces a range of improvements, including a modular architecture based on OSGi (Open Service Gateway Initiative) and a more streamlined development experience.

Major steps of upgrade

- Database upgrade
- Code upgrade

Database is upgrade via docker image and custom development is upgrade manually.

Note : Details of this project is in Chapter 4

2.2 Some Out of The Box (OOTB) features of Liferay used

2.2.1 Service Builder

Liferay Service Builder is a powerful tool that simplifies the development of database-driven applications within the Liferay platform. It provides a code generation mechanism that automates the creation of persistence and service layers, reducing the manual effort required for database integration.

This generated code provides a convenient and standardized way to interact with the database, perform CRUD operations (Create, Read, Update, Delete), and access and manipulate data within Liferay's framework. It abstracts away the complexities of database management, allowing developers to focus on application logic and functionality.[6]

2.2.2 MVC Portlet Web Module

In Liferay, portlets are the components that generate the user interface of required web application. Liferay's MVC portlets are portlets that follow the Model View Controller (MVC) architectural pattern. It consists of three layers.

- **Model:** The model layer represents the application's data and business logic. It contains Java classes that interact with the database and perform operations on the data.
- **View:** The view layer represents the application's user interface and includes JSP files, HTML templates, and CSS style sheets.
- **Controller:** The controller layer handles user requests and coordinates interactions between the model and view layers. It contains Java classes that process user input and generate appropriate responses.

Liferay's MVC portlets are based on the Spring MVC framework and use the Liferay Service Builder to generate the model layer. They provide a standardized way to develop portlets and promote code reuse and modularity.[5]

2.2.3 Liferay Headless REST API for Content Management System

Liferay's headless APIs and CMS (Content Management System) provide a powerful and flexible solution for managing and delivering content across different digital channels. With Liferay's headless API for blogs, developers can programmatically create, update, and retrieve blog posts. This API enables seamless integration with external systems and applications, ensuring engaging and informative blog content for users.

The headless API for documents allows efficient management of various file types, such as PDFs and images. Developers can upload, organize, and retrieve documents, ensuring easy access to relevant resources.

Liferay's headless API for knowledge base articles enables the creation and organization of informative articles. These articles serve as a valuable knowledge base, providing users with self-service support and comprehensive information.

The CMS component of Liferay ensures efficient content organization, version control, and access control for blogs, documents, and knowledge base articles. It offers a unified platform for managing and delivering content, enhancing user experiences across digital channels.[2]

Chapter 3

Present Work on React-based Web Application

3.1 Introduction

This Chapter focuses on the development of a React portlet and its deployment on Liferay DXP using Clay, a UI component library. The React portlet is designed to perform CRUD (Create, Read, Update, Delete) operations using the API provided by Liferay for various content types, including blog posts, documents, and knowledge base articles.

Liferay DXP is an enterprise portal platform that enables organizations to build and manage their digital experiences. It provides a robust set of features, including content management, collaboration tools, and integration capabilities. React, on the other hand, is a popular JavaScript library for building user interfaces, known for its flexibility, performance, and reusability.

The objective of this project is to develop a highly interactive and responsive user interface using React components and integrate it with Liferay DXP. By leveraging the API provided by Liferay, we can create, retrieve, update, and delete content within the Liferay platform. Specifically, we will focus on three content types: blog posts, documents, and knowledge base articles.

The use of Clay, a UI component library developed by Liferay, allows us to quickly and efficiently build the user interface by leveraging pre-designed components. Clay provides a set of reusable and customizable components that adhere to Liferay's design guidelines, ensuring a consistent and visually appealing user experience.

The CRUD operations performed through the React portlet will enable users to create new blog posts, documents, and knowledge base articles, retrieve existing content, update content, and delete content. These operations will be seamlessly integrated with Liferay DXP, providing a unified and user-friendly interface for managing different types of content.

In summary, this project aims to demonstrate the development of a React portlet deployed on Liferay DXP using Clay. The portlet will leverage Liferay's API to perform CRUD operations on various content types, including blog posts, documents, and knowledge base articles. By combining the power of React, Clay, and Liferay DXP, we can create a modern and efficient content management solution.

3.2 Developing a React Application using Liferay

The Liferay React Module is a module that allows developers to create React-based web applications within the Liferay platform. React is a popular JavaScript library used for building user interfaces. With the Liferay React Module, developers can leverage the power of React while taking advantage of Liferay's features, such as user authentication, content management, and workflow management.

The Liferay React Module provides a set of tools and components that developers can use to build their applications. It includes a React component library, which provides pre-built UI components that can be used to create web pages and interfaces. It also includes a development environment that allows developers to create, test, and debug their React applications within Liferay.[3]

3.3 Setting Up Dependencies

Use the Yeoman generator-liferay-js generator. This is the path to take when you are not already using a Liferay Gradle Workspace. After you install it, use the `yo liferay-js` command to start the generator. If this is the route you want to take, first install `yo` and the generator using the command `npm install -g yo generator-liferay-js` and then you can use the `yo liferay-js` command.

1. Install node and npm
2. Install yo `npm install -g yo`
3. Install liferay-js `npm install -g generator-liferay-js`

3.3.1 Using the Liferay JS Generator

Run the Liferay JS Generator: `yo liferay-js` and follow the prompts:

- What type of project do you want to create? React Widget
- What name shall I give to the folder hosting your project? `my-react-app`
- What is the human readable description of your project? `My React App`
- Do you want to add localization support? `Y`
- Do you want to add configuration support? `Y`
- Under which category should your widget be listed? `category.sample`
- Do you have a local installation of Liferay for development? `N` // If you
- are using Docker, answer no.
- Do you want to generate sample code? `N`

3.3.2 How to Build and Deploy

The steps to deploy your widget depend on where you are deploying it to. Here I am deploying to a Docker container and to a local Liferay bundle.

- Before we deploy to our Docker container we need to build our widget. We can build the widget using the following command: **npm run build**
- In order to deploy to docker you need to know the Container ID for your Liferay Docker container. You can find this by running the command: **docker ps**
- With our Container ID we can deploy our newly built module using the docker cp command: **docker cp dist/[my-app.jar] [container]:/opt/liferay/deploy**

3.4 Creating a React Component

- In src create a new file: App.js with the following contents:

```
import React from 'react';

function App() {
  return (
    <h1>Hello World</h1>
  );
}

export default App;
```

We can include this component in our entry file index.js

- At the top of index.js add the following imports:

```
import React from 'react';
import ReactDOM from 'react-dom';

import App from './App';
```

The first two import statements are importing dependencies that have been included from our package.json file. The last import statement is importing the React component we just created. Notice that we import it using a relative path.

- Then replace the export statement in index.js with the following:

```
export default function main({portletNamespace, contextPath, portletElementId, configuration}) {
  ReactDOM.render(
    <App />,
    document.getElementById(portletElementId)
  );
}
```


3.5 Liferay JavaScript Object

Liferay has a suite of Javascript objects and services at your disposal, even if you are using React.

The Liferay JavaScript object exposes methods, objects, and properties that you can use to access Liferay DXP-specific information. This section contains a comprehensive list of some of the most useful utilities you can find inside the Liferay object.

```
_getEntries(callback) {  
  var instance = this;  
  
  Liferay.Service(  
    '/assettag/get-groups-tags',  
    {  
      groupIds: instance.get('groupIds'),  
    },  
    callback  
  );  
}
```

You can still do this in your React code, too!

Liferay has lots of other JS available such as `fire()` and `on()` for InterPortlet Communication (IPC), that works even with React code.

3.6 Calling APIs

Liferay's headless RESTful APIs provide a way to interact with Liferay's content and services in a decoupled manner, allowing you to retrieve data and perform operations programmatically. These APIs follow REST principles and use standard HTTP methods for various operations[1]

3.6.1 Consuming REST Services

Sign in to Liferay at <http://localhost:8080>. Use the email address `test@liferay.com` and the password `test`. When prompted, change the password to `learn`.

- Liferay DXP's REST services are published at this URL:

```
http[s]://[hostname]:[port]/o/api
```

- On your Docker instance, you can find them here:

```
http://localhost:8080/o/api
```

APIs are divided into several categories. This example uses the BlogPosting service to retrieve blog posts from the Blogs widget, but you can use this procedure with any of the published services.

1. Select the Headless Delivery category. This category contains the BlogPosting service. You can use the filter to search for services.

2. Click the Show Schemas button, and on the right side of the screen a list of all the schemas in this category appear
3. Keep a browser tab open to the schema browser; when you want to PUT a BlogPosting, you'll need its schema

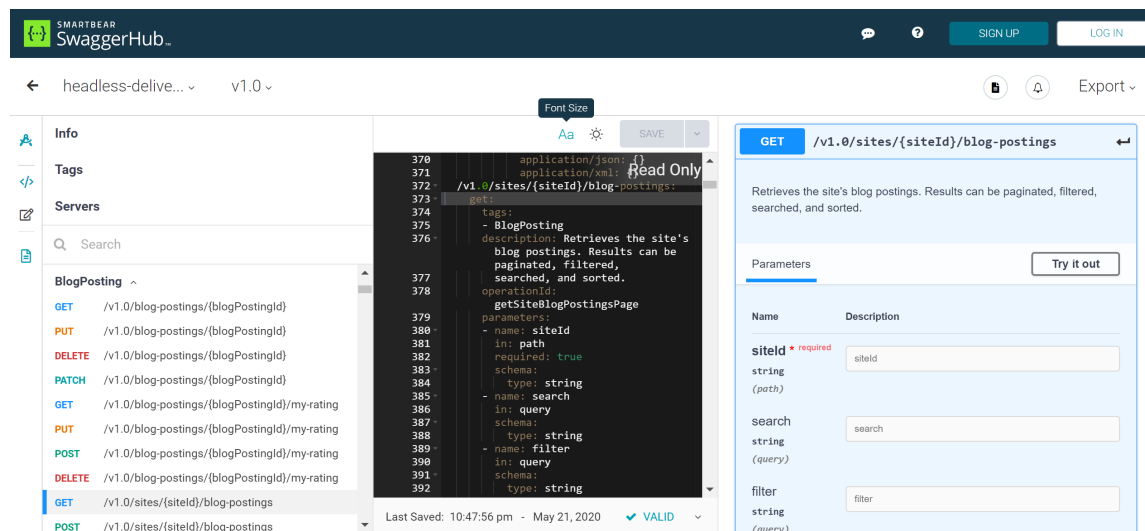


Figure 3.1: BlogPosting headless API

Document			
DELETE	/v1.0/documents/{documentId}	deleteDocument	▼
GET	/v1.0/documents/{documentId}	getDocument	▼
PATCH	/v1.0/documents/{documentId}	patchDocument	▼
PUT	/v1.0/documents/{documentId}	putDocument	▼
DELETE	/v1.0/asset-libraries/{assetLibraryId}/documents/by-external-reference-code/{externalReferenceCode}	deleteAssetLibraryDocumentByExternalReferenceCode	▼
GET	/v1.0/asset-libraries/{assetLibraryId}/documents/by-external-reference-code/{externalReferenceCode}	getAssetLibraryDocumentByExternalReferenceCode	▼
PUT	/v1.0/asset-libraries/{assetLibraryId}/documents/by-external-reference-code/{externalReferenceCode}	putAssetLibraryDocumentByExternalReferenceCode	▼
DELETE	/v1.0/documents/{documentId}/my-rating	deleteDocumentMyRating	▼
GET	/v1.0/documents/{documentId}/my-rating	getDocumentMyRating	▼
POST	/v1.0/documents/{documentId}/my-rating	postDocumentMyRating	▼
PUT	/v1.0/documents/{documentId}/my-rating	putDocumentMyRating	▼

Figure 3.2: Document headless API

KnowledgeBaseArticle ^		
DELETE	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}	deleteKnowledgeBaseArticle ✓
GET	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}	getKnowledgeBaseArticle ✓
PATCH	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}	patchKnowledgeBaseArticle ✓
PUT	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}	putKnowledgeBaseArticle ✓
DELETE	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}/my-rating	deleteKnowledgeBaseArticleMyRating ✓
GET	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}/my-rating	getKnowledgeBaseArticleMyRating ✓
POST	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}/my-rating	postKnowledgeBaseArticleMyRating ✓
PUT	/v1.0/knowledge-base-articles/{knowledgeBaseArticleId}/my-rating	putKnowledgeBaseArticleMyRating ✓
DELETE	/v1.0/sites/{siteId}/knowledge-base-articles/by-external-reference-code/{externalReferenceCode}	deleteSiteKnowledgeBaseArticleByExternalReferenceCode ✓
GET	/v1.0/sites/{siteId}/knowledge-base-articles/by-external-reference-code/{externalReferenceCode}	getSiteKnowledgeBaseArticleByExternalReferenceCode ✓

Figure 3.3: Knowledge Base Article headless API

3.6.2 Calling APIs in react component

```
const getBlog = async () => {
  const response = await Liferay.Util.fetch('/o/headless-delivery/v1.0/sites/${Liferay.ThemeDisplay.getSiteGroupId()}', {
    method: 'GET',
    headers: {
      'content-Type': 'application/json',
    }
  })

  const json = await response.json();
  setBlog(json.items);
}
```

by using `Liferay.Util.fetch` liferay java script object does not required authentication and `Liferay.ThemeDisplay.getSiteGroupId()` to get group id needed for hitting the above APIs

3.7 Using Clay Components

Clay is a UI framework developed by Liferay that provides a collection of reusable and customizable components for building modern and consistent user interfaces in Liferay DXP. It offers a set

of design patterns, styles, and pre-built components that can be easily integrated into Liferay applications, ensuring a consistent look and feel across different modules and page

- To use a Clay component we first have to install it's npm module. In this case, we want to use the Clay form component, so we can install it using the following command:

```
npm install @clayui/form
```

- After installing we can use it in our component by importing it in the top of our file like this:

```
import ClayForm, {ClayInput} from '@clayui/form';
```

3.8 Experiment and Results

Complete Functioning App after deployment in Liferay DXP

3.8.1 Navigation bar

To navigation for what service you want to use Blog Posting, Document and Knowledge Base Article

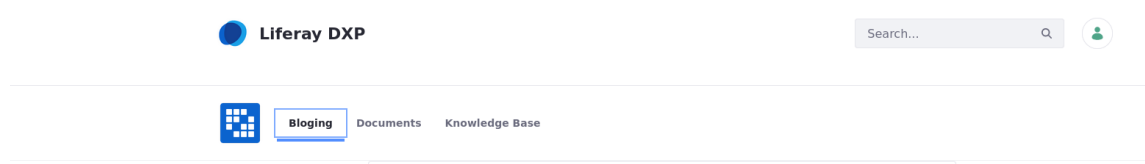


Figure 3.4: Navigation bar

3.8.2 Create/Add Entries

Using POST method of APIs

- Blog Posting

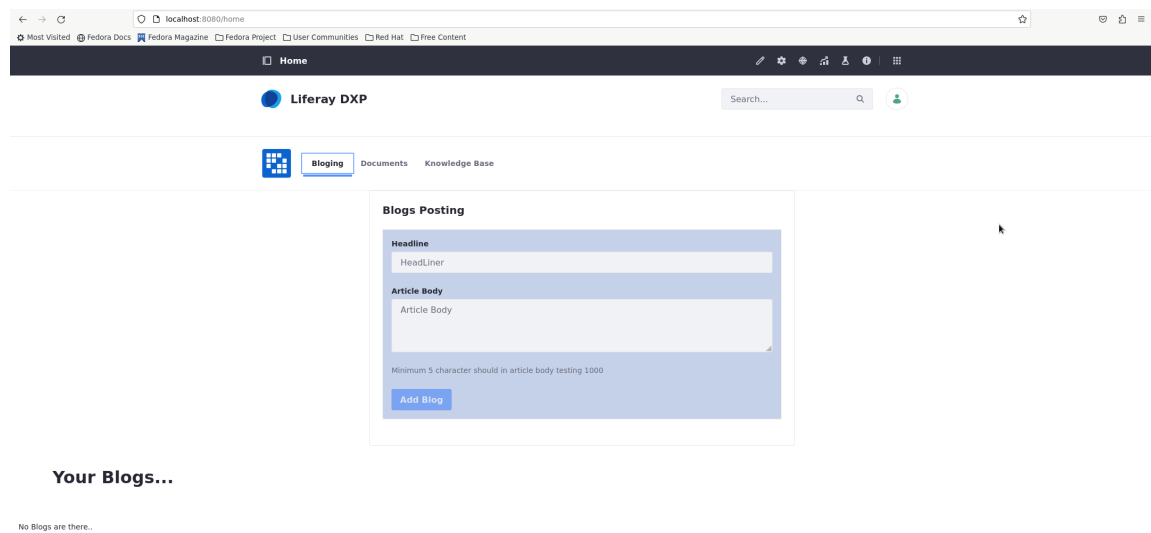


Figure 3.5: Adding Blog Posting

- Document

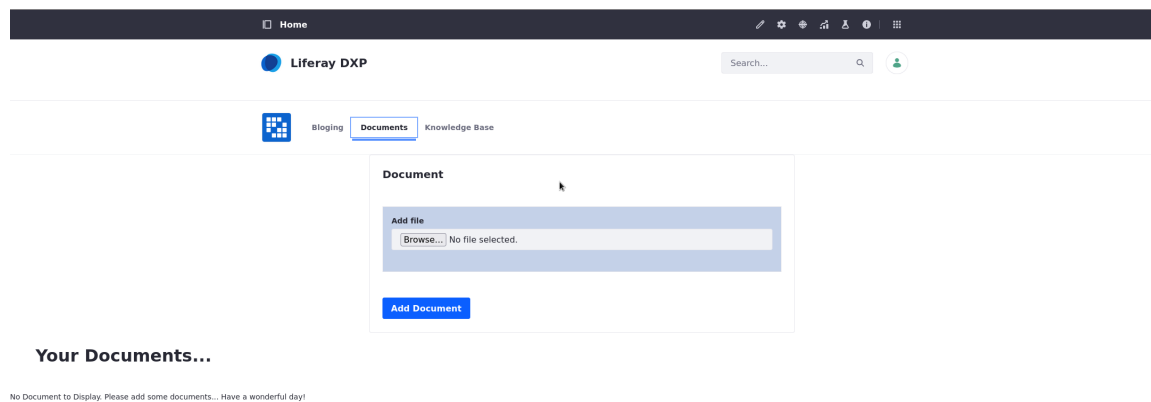


Figure 3.6: Adding Documents

- Knowledge Base Article

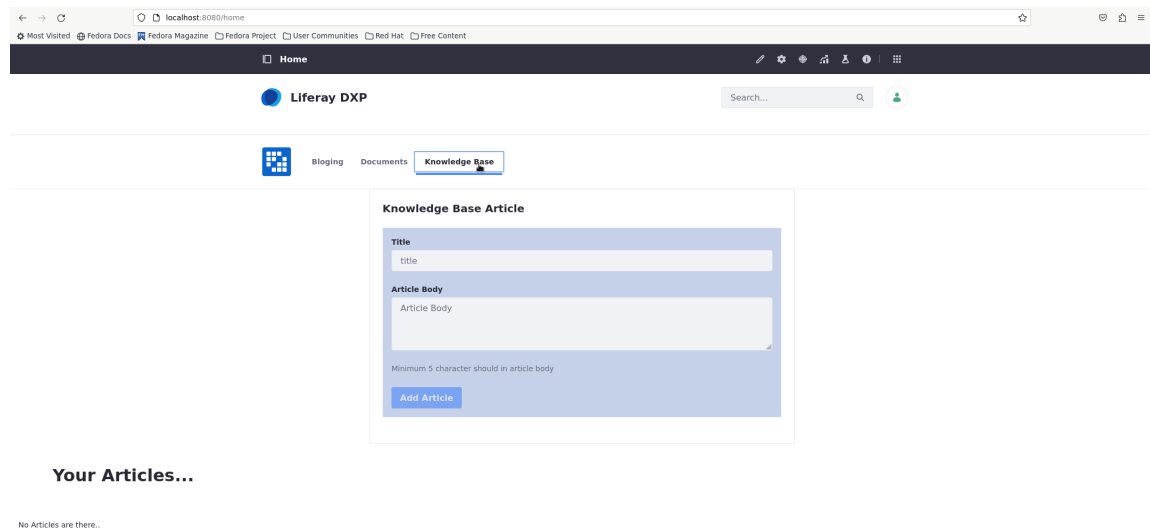


Figure 3.7: Adding Knowledge Base Article

3.8.3 Read/Fetch, Update and Delete entries

After adding some entries, By using **GET** Method i am fetching data from server.

For deleting and updating **DELETE** and **POST** method of respective APIs are used here. Which takes ID(unique number to identify) of that particular entry that need to be deleted or updated

- Blog Posting

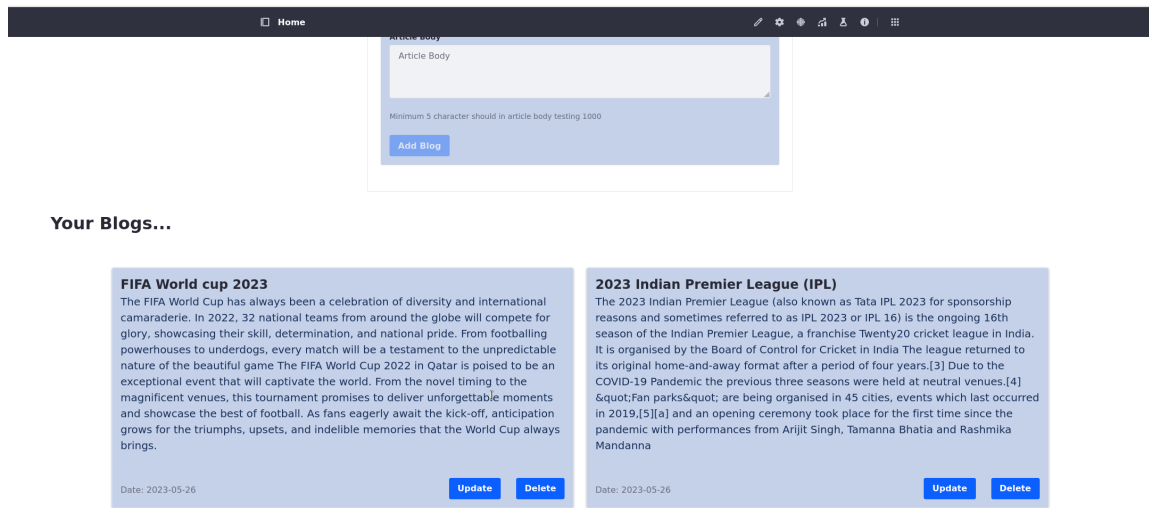


Figure 3.8: Fetching Blog Posting

- Document

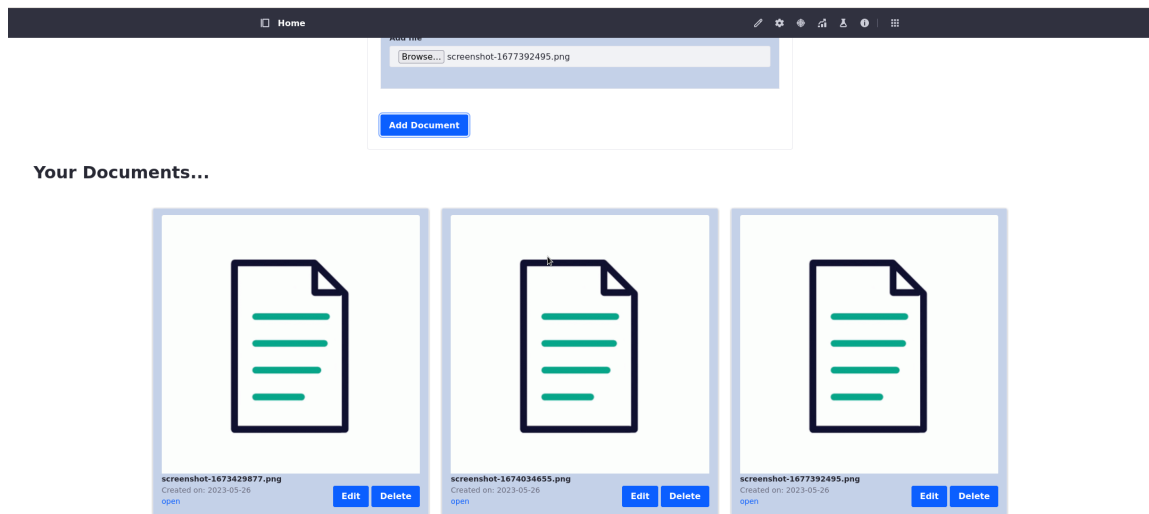


Figure 3.9: Fetching Document

- Knowledge Base Article

Home

Knowledge Base Article

Title

title

Article Body

Article Body

Minimum 5 character should in article body

Add Article

Your Articles...

Exercises to Tone Every Inch of Your Body

Challenging your balance is an essential part of a well-rounded exercise routine. Lunges do just that, promoting functional movement while also increasing strength in your legs and glutes. Start by standing with your feet shoulder-width apart and arms down at your sides. Take a step forward with your right leg and bend your right knee as you do so, stopping when your thigh is parallel to the ground. Ensure that your right knee doesn't extend past your right foot. Push up off your right foot and return to the starting position. Repeat with your left leg. This is one rep. Complete 3 sets of 10 reps.

Date: 2023-05-26

UpdateDelete

Figure 3.10: Fetching Knowledge Base Article

Chapter 4

Present Work on Upgrade project

4.1 Introduction

This chapter addresses the important task of upgrading a Liferay 6.2 application, which includes a form build in Liferay 6.2 for collecting user data and saving it in a MySQL database, to the latest version, Liferay 7.4 DXP. The goal of this upgrade is to take advantage of the improved features, better performance, and enhanced security offered by Liferay 7.4 DXP, resulting in a smooth and optimized user experience.

The existing Liferay 6.2 application consists of a basic form where users input their information, which is then stored in a MySQL database. However, due to advancements in technology and the availability of Liferay 7.4 DXP, it has become necessary to upgrade the application. This upgrade ensures that the application remains compatible with current digital trends and harnesses the new features provided by Liferay's latest version.

Database used in version 6.2 is MYSQL 5.7 which is compatible and database used in 7.4 is MYSQL 8.0 according to its compatibility matrix.

4.2 Why we need to Upgrade?

We need to upgrade from Liferay 6.2 to 7.4 to benefit from improved features, enhanced performance, and advanced security offered by the newer version, resulting in a better user experience and compatibility with the latest technologies and trends.

The database upgrade is necessary to ensure compatibility with the upgraded Liferay version and to take advantage of any improvements or optimizations made to the database management system. When upgrading to a new version of Liferay, there might be changes in the database schema, such as the addition or deletion of columns or tables. Upgrading the database ensures that the data structure aligns with the updated application requirements, allowing for proper data storage and retrieval. It helps maintain data integrity and ensures the compatibility between the application and the database in the upgraded environment.

4.3 Approach

In the Liferay upgrade process, the update typically occurs in two steps: the database update and the code update.[4]

- **Database Update:** The first step involves updating the underlying database to align with the requirements of the upgraded Liferay version. This process may involve schema changes, such as adding or removing tables and columns, modifying data types, or altering indexes. The purpose of this update is to ensure that the database structure is compatible with the upgraded Liferay environment and can effectively store and retrieve data.
- **Code Update:** After the database update is complete, the next step is to update the application's codebase to reflect the changes introduced in the upgraded Liferay version. This involves modifying or rewriting the existing code to utilize the new features, APIs, and configurations provided by the upgraded Liferay version. Additionally, any customizations made to the previous version may need to be adapted or integrated into the updated codebase.

4.3.1 Database Upgrade

Upgrading Liferay's DXP and Portal environments is made easy and safe through the tools and instructions provided by Liferay. For smaller data sets in non-clustered environments, an upgrade can be performed using a Docker image, which simplifies the process and ensures a quick and secure upgrade.

However, for more complex environments with larger data sets or numerous customizations, the Database Upgrade Tool is utilized. This tool is specifically designed to handle the challenges that come with upgrading DXP and Portal environments that have intricate configurations and extensive customizations. It ensures that the database is upgraded seamlessly while preserving data integrity and compatibility with the upgraded environment.

By offering these tools and instructions, Liferay ensures that organizations can safely and efficiently upgrade their Liferay environments, regardless of their complexity or data size. This enables organizations to leverage the latest features and improvements while minimizing disruption to their existing systems.

Backup Database

A backup of the database is crucial during the upgrade process to ensure the safety and integrity of the data. In case of any unexpected issues or errors during the upgrade, having a backup allows for easy restoration of the database to its previous state, minimizing the risk of data loss or corruption. Using **mysqldump**, you can perform full backups of the entire database, specific tables, or even select specific rows or columns. This flexibility enables you to customize the backup process according to your requirements.

- **Backup:**

```
docker exec <CONTAINER> /usr/bin/mysqldump -u <user>  
|password=<password> <DATABASE> > backup.sql
```

- Restore:

```
docker exec -i some-mysql sh -c 'exec mysql -u<user> -p<password>
<database>' < /some/path/on/your/host/backup.sql
```

Upgrading Via Docker

Here are the steps for upgrading with a Docker image:[7]

1. Create an arbitrary folder to use with the new Liferay Docker image and create subfolders called files and deploy
 - **mkdir -p new-version/files**
files: The Docker container copies files from this folder to the container's Liferay Home folder.
 - **mkdir -p new-version/deploy**
deploy: The Docker container copies artifacts from this folder to the container's auto-deploy folder.
2. Copy and merge the Liferay Home files and application server files from your backup to their corresponding locations in the files folder (your new [Liferay Home]). For example, copy your activation key to new-version/files/license/. The files may include but are not limited to these:
 - /license/*: Activation keys. (Subscription)
 - /log/*: Log files.
 - /osgi/configs/*.config: OSGi configuration files.
 - portal-*.properties: Portal properties files, such as portal-ext.properties.
 - setenv.sh, startup.sh, and more: Application server configuration scripts.
 - web.xml: Portal web application descriptor.
3. Run the Docker image mounted to your new version folder using the following command. Substitute the image name, tag, and environment values as needed.

```
docker run -it -m 8g -p 8080:8080 \
-v $(pwd)/new-version:/mnt/liferay \
-e LIFERAY_UPGRADE_PERIOD_DATABASE_PERIOD_AUTO_PERIOD_RUN=true \
liferay/[place image name here]:[place tag here]
```

4. Use the following environment variable when running the new DXP/Portal Docker image to auto-run the Database Upgrade Tool on the configured database.

```
-e LIFERAY_UPGRADE_PERIOD_DATABASE_PERIOD_AUTO_PERIOD_RUN=true
```

This environment variable trigger database upgrade

4.3.2 Code upgrade

Upgrading Your Development Environment

A Liferay Workspace is a generated environment that is built to hold and manage your Liferay projects. It is intended to aid in the management of Liferay projects by providing various build scripts and configured properties.

Liferay Workspace is the recommended environment for your code migration; therefore, it will be the assumed development environment in this section

Liferay adopted the OSGi framework starting from Liferay 7, which introduced the concept of OSGi modules known as OSGi bundles. These bundles provide a more modular and flexible architecture for developing and deploying applications within the Liferay platform.

Therefore, Liferay 6.2 does not follow the OSGi-based architecture, while newer versions such as Liferay 7 and beyond are built on the OSGi framework, enabling better modularity and extensibility

Upgrading Custom Development

The Liferay Upgrade Planner provides an automated way to adapt your installation's data and legacy plugins to your desired Liferay DXP upgrade version. You can also use the planner to upgrade your data; this is a separate process that must be done independently from the code upgrade process.

Manually upgrade is also option, manual code upgrades require a deep understanding of Liferay's architecture, APIs, and coding practices. They can be complex and time-consuming, so it is recommended to plan and allocate sufficient resources for testing and troubleshooting during the upgrade process.

4.4 Experiment and results

4.4.1 Web Application that is going to be updated

This form collects user data and saves in MYSQL 5.7 database

Most Visited Fedora Docs Fedora Magazine Fedora Project User Communities Red Hat Free Content

Warning! Due to inactivity, your session has expired. Please save any data you may have entered before refreshing the page.

Liferay

Welcome

Welcome

Form

Your request completed successfully.

Add Entry

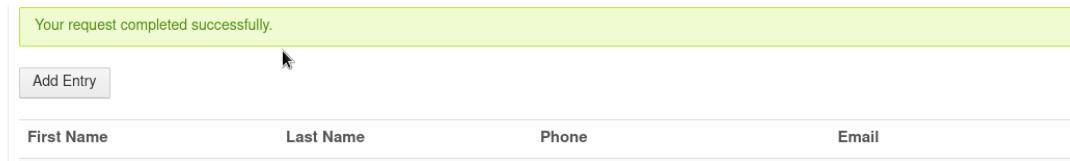
First Name	Last Name	Phone	Email	Address	City	Pin Code
spider	man	9999999999	manaspider@gmail.com	spider lane	spider city	1000
super	man	1000000000	mansuper@gmail.com	super verse	super land	9999
Bat	Man	8888888888	manbat@hotmail.com	dark knight	black city	555555
wonder	women	66666666	womenwow@yahoo.com	wonder land	wonder city	77777777

Sign In

You are signed in as Test Test.

Powered By Liferay

Figure 4.1: Basic Form a web application which collects user details and it is deployed on liferay 6.2

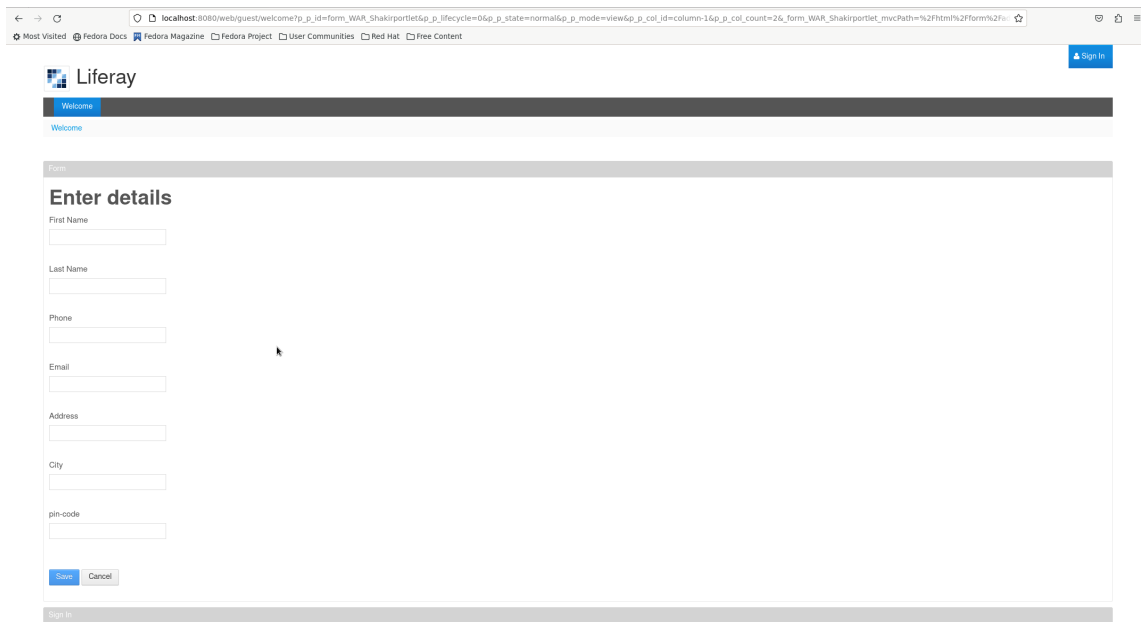


Your request completed successfully.

Add Entry

First Name	Last Name	Phone	Email
------------	-----------	-------	-------

Figure 4.2: Clicking on Add Entry redirect you to the Form page



Liferay

Welcome

Welcome

Form

Enter details

First Name

Last Name

Phone

Email

Address

City

pin-code

Sign In

Figure 4.3: By clicking saves after entering details it saves data in Database

4.4.2 DataBase Upgrade

Following database upgrade process via docker image after restoring database backup, which i took from MYSQL 5.7 of liferay 6.2 in MYSQL 8 is going to upgrade

```

2023-04-10 10:39:35.549 INFO [com.liferay.portal.osgi.web.wab.extender.internal.WabFactory-BundleTrackerOpener][ThemeHotDeployListener:108] 1 theme for speedwell-theme is available for use
2023-04-10 10:39:35.586 INFO [com.liferay.portal.osgi.web.wab.extender.internal.WabFactory-BundleTrackerOpener][ThemeHotDeployListener:108] 1 theme for admin-theme is available for use
2023-04-10 10:39:35.637 INFO [com.liferay.portal.osgi.web.wab.extender.internal.WabFactory-BundleTrackerOpener][ThemeHotDeployListener:108] 1 theme for classic-theme is available for use
2023-04-10 10:39:35.687 INFO [com.liferay.portal.osgi.web.wab.extender.internal.WabFactory-BundleTrackerOpener][ThemeHotDeployListener:108] 1 theme for minium-theme is available for use
2023-04-10 10:39:35.727 INFO [com.liferay.portal.osgi.web.wab.extender.internal.WabFactory-BundleTrackerOpener][ThemeHotDeployListener:108] 1 theme for dialect-theme is available for use
2023-04-10 10:39:46.408 INFO [main][UpgradeProcess:127] Upgrading com.liferay.content.dashboard.document.library.internal.upgrade.registry.v_1_17.AssetVocabularyUpgradeProcess
2023-04-10 10:39:46.451 INFO [main][UpgradeProcess:145] Completed upgrade process com.liferay.content.dashboard.document.library.internal.upgrade.registry.v_1_17.AssetVocabularyUpgradeProcess in 45 ms
2023-04-10 10:40:19.926 INFO [main][UpgradeProcess:127] Upgrading com.liferay.portal.search.tuning.synonyms.web.internal.upgrade.v1_0_0.SynonymSetsDatabaseImporterUpgradeProcess
2023-04-10 10:40:19.992 INFO [main][UpgradeProcess:145] Completed upgrade process com.liferay.portal.search.tuning.synonyms.web.internal.upgrade.v1_0_0.SynonymSetsDatabaseImporterUpgradeProcess in 65 ms
2023-04-10 10:40:20.012 INFO [main][UpgradeProcess:127] Upgrading com.liferay.portal.search.tuning.rankings.web.internal.upgrade.v1_0_0.RankingsDatabaseImporterUpgradeProcess
2023-04-10 10:40:20.038 INFO [main][UpgradeProcess:145] Completed upgrade process com.liferay.portal.search.tuning.rankings.web.internal.upgrade.v1_0_0.RankingsDatabaseImporterUpgradeProcess in 25 ms
2023-04-10 10:41:34.632 INFO [main][UpgradeProcess:127] Upgrading com.liferay.adaptive.media.document.library.thumbnails.internal.upgrade.v1_0_0.DocumentLibraryThumbnailsConfigurationUpgradeProcess
2023-04-10 10:41:34.632 INFO [main][LoggingTimer:83] Starting com.liferay.adaptive.media.document.library.thumbnails.internal.upgrade.v1_0_0.DocumentLibraryThumbnailsConfigurationUpgradeProcess#doUpgrade
2023-04-10 10:41:34.672 INFO [main][LoggingTimer:44] Completed com.liferay.adaptive.media.document.library.thumbnails.internal.upgrade.v1_0_0.DocumentLibraryThumbnailsConfigurationUpgradeProcess#doUpgrade in 39 ms
2023-04-10 10:41:34.672 INFO [main][UpgradeProcess:145] Completed upgrade process com.liferay.adaptive.media.document.library.thumbnails.internal.upgrade.v1_0_0.DocumentLibraryThumbnailsConfigurationUpgradeProcess in 41 ms
Executing verify com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess
2023-04-10 10:41:39.254 INFO [main][VerifyProcess:77] Verifying com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess
2023-04-10 10:41:39.254 INFO [main][LoggingTimer:83] Starting com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess#verifyAccountRoles
2023-04-10 10:41:39.627 INFO [main][LoggingTimer:44] Completed com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess#verifyAccountRoles in 372 ms
2023-04-10 10:41:39.628 INFO [main][LoggingTimer:83] Starting com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess#verifyAccountGroup
2023-04-10 10:41:39.631 INFO [main][LoggingTimer:44] Completed com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess#verifyAccountGroup in 4 ms
2023-04-10 10:41:39.632 INFO [main][VerifyProcess:90] Completed verification process com.liferay.commerce.account.internal.verify.CommerceAccountServiceVerifyProcess in 379 ms
Executing verify com.liferay.portal.security.service.access.policy.internal.verify.SAPServiceVerifyProcess
2023-04-10 10:41:39.642 INFO [main][VerifyProcess:77] Verifying com.liferay.portal.security.service.access.policy.internal.verify.SAPServiceVerifyProcess
2023-04-10 10:41:39.643 INFO [main][LoggingTimer:83] Starting com.liferay.portal.security.service.access.policy.internal.verify.SAPServiceVerifyProcess#verifyDefaultSAPEntry
2023-04-10 10:41:39.676 INFO [main][LoggingTimer:44] Completed com.liferay.portal.security.service.access.policy.internal.verify.SAPServiceVerifyProcess#verifyDefaultSAPEntry in 32 ms
2023-04-10 10:41:39.676 INFO [main][VerifyProcess:90] Completed verification process com.liferay.portal.security.service.access.policy.internal.verify.SAPServiceVerifyProcess in 34 ms
Executing verify com.liferay.commerce.product.internal.verify.CommerceProductServiceVerifyProcess
2023-04-10 10:41:39.687 INFO [main][VerifyProcess:77] Verifying com.liferay.commerce.product.internal.verify.CommerceProductServiceVerifyProcess
2023-04-10 10:41:39.688 INFO [main][LoggingTimer:83] Starting com.liferay.commerce.product.internal.verify.CommerceProductServiceVerifyProcess#verifyCPMeasurementUnits

```

Figure 4.4: During database upgrade

4.4.3 DataBase Upgrade

After manually upgrade the code of 6.2 to 7.4 which follows OSGi-based architecture by copying lines of code and configuration.

Welcome

Liferay DXP

Search...

Welcome

SIGN IN

You are signed in as Test Test.

Form

FORM

Add Entry

First Name	Last Name	Phone	Email	Address	City	Pin Code
demo	demo	555555	demo@gmail.com	demo lane	kolkata	707070707
john	wick	89898989898	john@gmail.com	address unknown	unknown	777777777
Bat	man	66666666	manbat@hotmail.com	demo lane	black city	90909
spider	man	66666666	bksmnta@gmail.com	hulu lulu lane	spider city	1000
after	upgarde	88888	upgard@liefrey.com	lulu hulu	kolkata	700012

Figure 4.5: After code upgrade and deployment to 7.4 DXP

After upgrading the custom code and deploying it in Liferay DXP 7.4, connected to the upgraded MySQL 8 database, a new table is created. To retain access to the previous data (before the upgrade) while incorporating the new data (after the upgrade), the content from the old table is copied to the new table. This ensures that both the pre-upgrade and post-upgrade data can be accessed and utilized effectively in the upgraded environment.

Chapter 5

Conclusion

This project report involves development of react-based web application using liferay as build tool and Upgrade of a liferay project which is build and deployed on liferay version 6.2 and going to upgrade to be deployed in liferay version 7.4.

Developing a React app with Liferay React, Clay, and Liferay's Headless APIs for blog postings, documents, and knowledge base articles offers an excellent approach for creating modern and interactive web applications. This combination leverages React's component-based architecture, Clay's reusable UI components, and Liferay's Headless APIs for seamless content management integration. The result is a visually appealing and user-friendly app that delivers a smooth and intuitive experience, enabling users to interact with dynamic and relevant content. By utilizing React, Clay, and Liferay's Headless APIs, developers can build powerful and responsive applications that align with current web development practices, providing an enhanced user experience.

Upgrade of project mainly focuses on the crucial task of upgrading a Liferay 6.2 application, specifically a basic form that collects user data and saves it in a MySQL database, to Liferay 7.4 DXP. The upgrade is necessary to leverage the enhanced features, improved performance, and advanced security provided by Liferay's latest version. By transitioning from MySQL 5.7 to MySQL 8.0, the application aligns with the compatibility requirements of Liferay 7.4 DXP. This upgrade ensures the application remains up-to-date, compatible with current technologies, and capable of delivering an optimized user experience.

Bibliography

- [1] calling apis. <https://help.liferay.com/hc/en-us/articles/360036343212-Using-REST-APIs>. Accessed: 2023-05-29.
- [2] Headless rest apis. <https://help.liferay.com/hc/en-us/articles/360028726992-Headless-REST-APIs>. Accessed: 2023-05-29.
- [3] Liferay react portlet. <https://liferay.dev/blogs/-/blogs/liferay-react-portlets>. Accessed: 2023-05-29.
- [4] Liferay upgrade. <https://learn.liferay.com/w/dxp/installation-and-upgrades/upgrading-liferay/upgrade-basics>. Accessed: 2023-05-29.
- [5] Mvc portlet. <https://help.liferay.com/hc/en-us/articles/360018159451-Liferay-MVC-Portlet>. Accessed: 2023-05-29.
- [6] Service builder. <https://help.liferay.com/hc/en-us/articles/360018160851-What-is-Service-Builder>. Accessed: 2023-05-29.
- [7] Upgrade via docker. <https://learn.liferay.com/w/dxp/installation-and-upgrades/upgrading-liferay/upgrade-basics/upgrading-via-docker>. Accessed: 2023-05-29.
- [8] What is lifeary. <https://help.liferay.com/hc/en-us/articles/360018183831-Introduction-to-What-is-Liferay>. Accessed: 2023-05-29.