VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A WEB-BASED DIGITAL LIBRARY FOR FINDING SIMILAR IMAGE USING NEURAL NETWORK

By
Đỗ Đông Quân - ITITIU19043

A thesis submitted to the School of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam
2023

A WEB - BASED DIGITAL LIBRARY FOR FINDING SIMILAR USING NEURAL NETWORK

APPROVED BY:

_____ ,

Ly Tu Nga, Ph.D

_____

Assoc. Prof. Nguyen Van Sinh

_____

_____

_____

THESIS COMMITTEE

ACKNOWLEDGMENTS

To begin with, it is with deep gratitude and appreciation that I acknowledge the professional guidance of Dr. Ly Tu Nga. Her constant encouragement and support helped me to achieve my goal and complete my thesis on time.

Secondly, my gratitude goes to my friends, who are always by my side to support and exchange helpful knowledge when facing things that have not been learned. I give special thanks to my colleagues who have studied and completed our last courses before we graduate this semester, especially this thesis.

I am grateful to the faculty of the School of Computer Science and Engineering of the International University, which provides me with the best conditions throughout my time at International University. Gratitude is also expressed to the members of my reading and examination committee.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In the advancing digital era, the vast amount of information on the internet has made it increasingly challenging to find relevant and similar resources, including books and documents. This has led to the development of digital libraries that utilize supportive technologies for information retrieval and recommendation, encompassing both text and image-based content. This project focuses on creating a web-based digital library that employs neural network techniques to facilitate the retrieval of book-related information using images.

The limitations of physical libraries, such as disrupted research activities and time-consuming processes of searching for available books, especially considering the large volume and diverse subjects, can be addressed through a digital library. Additionally, while keyword-based (text, field, title, author, etc) searching is the primary method in digital libraries, incorporating image-based search, as effectively demonstrated by e-commerce websites, can enhance the user experience and provide more accurate search results. To address these issues, this project builds a digital library website with basic functionalities for users, including book searching, viewing book information, and selecting reservation dates. Furthermore, the addition of image-based search functionality using a machine learning model based on neural networks will be implemented. Through the research and development process, a successful web-based digital library has been built, providing book information to users and allowing them to check the availability of books. The project also includes a dashboard for administrators to manage the data of the digital library.

Moreover, training a machine learning model based on neural networks to extract image features and perform similar image retrieval has demonstrated its compatibility with book cover images and accurate search results. To enhance the performance of the project, the utilization of optimization techniques using neural networks can improve the model's output. Additionally, a model for cropping the main content of input images can contribute to improving the efficiency of the information retrieval function. In terms of the digital library website, the addition of more diverse data to align with real-world scenarios and optimizing the interaction between administrators and the system's dashboard are crucial.

Keywords: web-based digital library, website, neural network, machine learning

Chapter 1 : INTRODUCTION

1.1. Background

As mentioned earlier, Covid-19 has brought us great difficulty when it is impossible to resume public activities and areas that attract people are completely closed. Therefore, everyone has to spend most of their time at home when everything is limited, especially when students cannot go to school and have to switch to online learning. As a result, most people can only access, and search all information and knowledge through the Internet. This problem makes students more and more capable of searching for information through the vast sources of knowledge out there, which few are as detailed as the school's library. As a result of the pandemic, some activities have been discontinued during that time, and the school library, an important resource for student knowledge, is no exception. This is a huge loss for students or those doing academic research when they desperately need information from books and articles in the school library to complete their reports and lessons.

In today's advanced technology era, the digital library is the most suitable solution to the problem mentioned above, is a platform that helps the school librarian to keep a close eye on the resources in the library easier, is a place where students can stay at home to find information about the book or article they need without going to the library to search and waste time with it. Besides, this system also helps students see if the books they need can be borrowed, if so, they can borrow on the days they want and then go to the library to receive the books. These applications of the "digital library" will help the confusing interaction between librarians with too many students, which will reduce pressure for both parties and save more time.

In addition, a basic digital library with image-based searching applies Machine Learning (ML) to help students to find the book easily they need when they have the cover image. This function makes the search process more diverse and easier, besides searching for books based on fields, author's name, and book title when students can't remember these information. This is the reason why image-based searching helps the output for students to be more accurate instead of you having to search through huge data of thousands of books to find the right book. Because of the problems mentioned above, the goal of this thesis is to create a digital library website that applies image search to satisfy and solve the problems mentioned.

1.2. Problem Statement

This section shows the problems that need to be solved in this project. There are 3 common problems that people will often encounter in the process of searching and researching from

library resources as well as how to manage resources in physical libraries for librarians including (1) Time-consuming and information overload, (2) Mistakes during managing resources of the library, and (3) simple way for searching books.

To begin with, *time-consuming and information overload* is the common first problem encountered during the research phase. The fact that the library is not working or going to the library to find books will affect many researchers. This takes a lot of time, forcing them to prolong their research and study time and slow down what they are doing. Besides, the fact that all students go to the library to search for books and borrow books will be overloaded and many unnecessary contacts can happen again like during the epidemic even though we have returned to normal life. But you still have to be on guard for all possible scenarios.

Some *mistakes during managing* a large number of books, second, is also another problem. The number of books that are too many and full of fields is considered to be very large even if it only stops at a few thousand, but it will still be difficult for readers to search at the library or not know if anyone has borrowed that book yet and waste time searching. Furthermore, people are capable of making mistakes when it comes to managing the number of books in the school library and the number of students using the library's services to borrow and read books. When such situations arise, it will be overwhelming for a librarian and time-consuming for students who need books.

Another problem is *the way to search for books* from digital libraries, usually, we will have to search by keywords such as the field, title, or author of a book. But what if we have a book cover image and do not have enough information to enter the necessary keywords to search? This has been supported on e-commerce platforms, when we look at online reviews and see a certain product but do not know the name, or brand, we can use that image to search, and the system will offer products related to the photo. It would be good if digital libraries also adopt this function as it allows users to choose the search method that is suitable for the information they have.

Therefore, the use of technology to manage a digital library as well as the image search function will help readers find the book information they need or be introduced to new books with images close to their offers the most and can check the availability of books and borrow them from the library.

1.3. Scope and Objectives

Having clear objectives for the system is necessary to be able to determine the steps for building a system like the itinerary objective. The defined objectives are outlined in the following table:

Objectives	Content
1	Build a digital library system for users, admin dashboard for admin
2	Provide an additional option for finding books by cover images.
3	Building a machine learning model for retrieving a similar image in the dataset with the query image from the user.

Table 1.1: Objective for the thesis

[Obj.1] *Build a digital library system:* Designing a "Digital Library" website requires some basic knowledge of how a library works as well as getting a basic rundown of the functionality of e-commerce websites. A complete digital library system will include a large database for user information (students and admin website) and book content. Therefore, the main goal of this project is just to help students find and borrow the book they need as well as help librarians store and manage data on Admin Dashboard and simplify all interactions. interaction between the user and the website. So, some basic functions for users will be as follows:

- Student (users):
    - Search book
    - View book information
    - Borrow book

It solves time-consuming and information overload during researching time (see Section 1.2)

- Admin (work with Admin Dashboard):
    - Manage books information (create, delete, view, edit)
    - Manage user accounts (create, delete, view, edit)
    - Manage slots of books (create, delete, view, edit)

The mistakes from managing resources in physical libraries for librarians are solved (see Section 1.2)

[Obj.2] *Provide additional options and find books:* A special function added to the digital library will be a search option for users when they can search for book information based on the cover image. Here the user can provide a book cover image that the user needs to find, the system will receive and return the books with the same cover image along with the details of the book for direct search on the digital library.

[Obj.3] *Train and apply ML model:* In response to the above image search option, it is imperative to train a machine learning model to help find similar images. This is a model that can support searching for book information based on the cover image in addition to searching for other information. The model trained here will be based on the neural network and extract the image into many layers to get the features of the image and then compare these features to give close images. It gives an additional way for searching books (see Section 1.2).

1.4. Assumption and Solution

As mentioned earlier, the interaction between account types with two systems such as the admin dashboard and the digital website is an interaction between the user and the system. Specifically, users can search for book information based on the basic information of the book such as the field, and the year of publication to shorten the scope of their search, thereby viewing the details of the book they are interested in such as author, publication year, ISBN, publisher,… then decide to borrow the book or not, but still have to log in with the account provided by the school for each student to proceed and check the status of the book to borrow. At this point, the system will provide books that can be lent for the desired period of the student, and not select books that have been previously lent by other students.

In addition, the image search function is also applied as an additional support tool for the digital library website, from which searchers can find the book they are holding through the input image they provide (An image of the entire book cover without cropping or rotating) to the system each step of this model is generally described as follows

- Step 1: Input the image
- Step 2: Feature extraction
- Step 3: Similarity measure
- Step 4: Output the image (according to the ratio of the match)

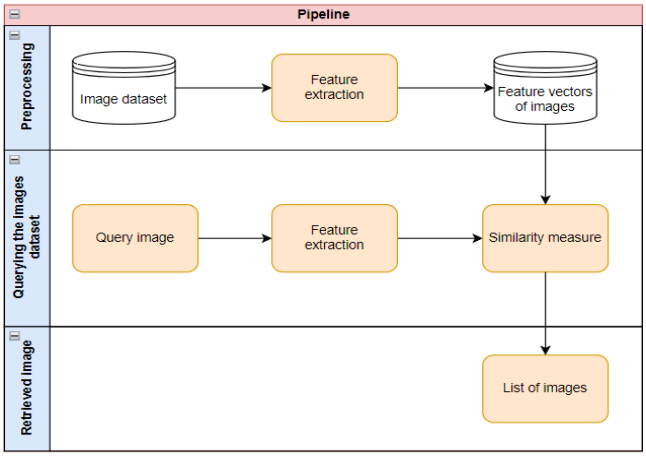


Figure 1.1: Pipeline of image search

1.5. Structure of thesis

The proposition contains 6 chapters, each of which will offer different perspectives on the issue and clarify it.

- Chap 1: Introduction – Overview of the research
- Chap 2: Literature review – Some works related to the paper
- Chap 3: Methodology – Technology is used
- Chap 4: Implementation – The resulting step by step
- Chap 5: Discussion – Look back and preliminary assessment
- Chap 6: Conclusion & Future Work

Chapter 2 : LITERATURE REVIEW

2.1. Digital Library

Digital library is a phrase that has been known and interesting for a long time, but it only emerged as an extremely important service in March 2020 when the Covid-19 pandemic appeared and had a significant impact on the entire physical library service worldwide, including at universities. To briefly describe what a digital library is, it is similar to a traditional library in that it contains books, articles, images, audio, and video files, but it does so digitally (as opposed to with the hardware format in the library). According to this book [3], the digital library and the World Wide Web differ in the following points: if the digital library has resources collected and created by each separate organization, the Web is different since it has all data publicly available but lacks the necessary tools to select and organize information sources. Therefore, digital libraries have the perfect combination of a collection of useful information from traditional libraries with the ability to access, search and manage based on technology or artificial intelligence applications of a real website.

The closure of physical library facilities due to the pandemic has prompted universities to focus on adapting their digital services and resources to incorporate their libraries. In the article [1], the authors have shown some examples of applications that universities can apply through digital libraries such as supporting learning and continuing research with the image, learning online, creating online collections to showcase materials, etc. In addition, universities expand access to e-books and collaborate with faculty to provide materials and online courses for students. These things have emphasized the importance of digital libraries in supporting teaching, learning, and research from the post-pandemic period to the future.

2.2. Finding Similar Image

Finding similar images involves using computer vision algorithms and techniques to identify and retrieve images that are visually like a query image. Several different applications, including e-commerce, content management, and search engines, can benefit from this. Finding comparable photos often entails extracting visual characteristics from the query image, such as color, texture, and form, and comparing them to a database of images to locate the ones that have those characteristics. Due to the differences in lighting, scale, orientation, and object stance that may alter how a image appears visually, finding related photographs can be difficult. Yet, improvements in computer vision and machine learning have made it feasible to create efficient methods for precisely locating photos that are similar.

There are numerous approaches to finding comparable images, and the best one relies on the particular use case and the data at hand. Many popular techniques for locating comparable photos include as follows

- Content-based image retrieval (CBIR): This method searches a database for related photos using visual cues including color, texture, form, and spatial arrangement. In image search engines and content management systems, CBIR is often employed.
- Feature matching: this is a technique that compares the commonalities between two images, such as corners or edges, in order to get a similarity score. Applications like object identification and image stitching frequently employ feature matching.
- Deep learning-based methods: These techniques employ neural networks to identify similarities between images and extract characteristics from them. Approaches based on deep learning have demonstrated promising outcomes in a variety of applications, including face recognition and image categorization.
- Clustering: Based on their aesthetic characteristics, comparable photographs are grouped together using this approach. Large image collections may be organized using clustering, and comparable photos can be found inside a cluster.
- Hybrid approaches: These techniques increase the precision of detecting related images by combining many techniques, such as Content-based image retrieval (CBIR) and deep learning.

In this project, the CBIR method will be used as the main option to implement the Finding Similar Image function for the Digital Library.

2.2.1. Content-based image retrieval

The concept of content-based image retrieval involves the use of computer vision algorithms to retrieve images. According to the definition from this study [4], this approach is commonly used in large databases to automatically search for digital photographs that precisely match the image to be retrieved. The term "Content-based" alludes to the method's active search and analysis of each image's content as opposed to the tags, keywords, or descriptions that go with it. Instead, the "image content," such as color, shape, texture, or any other information that may be derived from the image, will be used to retrieve images from enormous data warehouses.

2.2.2. Content-based image retrieval applications

The development of storage technology has made it possible to combine images into large image databases. As it can allow the retrieval of images that are often encoded into feature vectors, the CBIR approach is suitable to successfully handle such image collections and the results that are the feature's closest matches to the image we're looking for. CBIR technology

is widely used in a variety of industries, including crime prevention, medicine, historical research, digital libraries, and biodiversity information systems [5].

2.3. Similar existing Website and Feature

2.3.1. Open Library

Open Library [7] is a digital library that was founded in 2006. It is a project of the non-profit organization Internet Archive, which is dedicated to building a digital library of Internet sites and other cultural artifacts in digital form. Open Library provides free access to millions of digitized books, including many rare and out-of-print titles, which can be read online or downloaded in various formats.

The Book Sponsorship initiative of Open Library, which enables people and organizations to support the digitization of certain books, is one of its distinctive characteristics. Under this approach, more books are now available to the general public and the Open Library's collection has grown. Together with its collection of digitized books, Open Library also offers a platform for users to publish and distribute their own digital works, as well as a community-driven cataloging system that enables users to add and modify information for books in the collection.

When we come to the main page of Open Library, users will be introduced by the system to a series of books in different fields that are most interesting and read by many users.

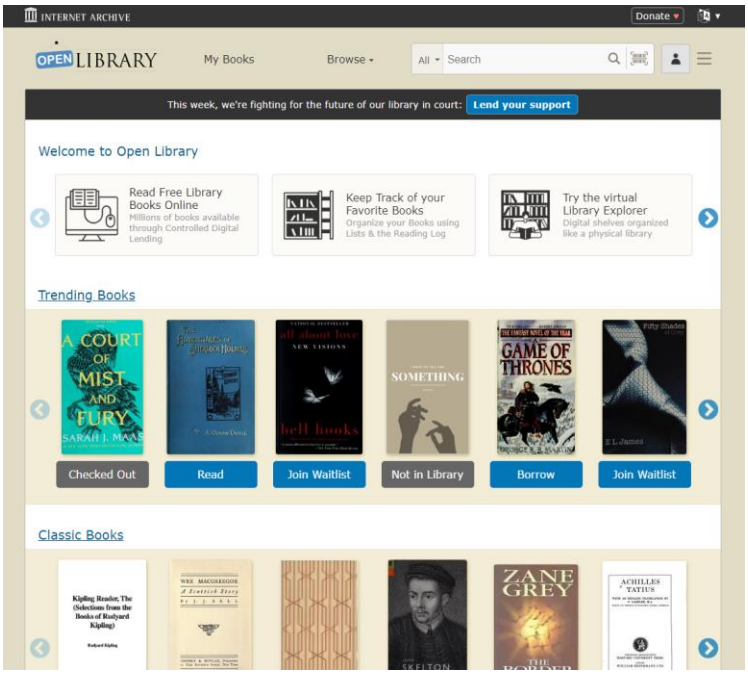


Figure 2.1: Open Library homepage [7]

One useful thing about this library is that it supports a lot of languages, which can be used by all users all over the world.

This digital library website supports users with the feature of searching for books by the book's information including title, author, and field, etc and a special feature is searching by barcode. Immediately turn on the device's camera to use and search.

Besides, an advanced search function is when the user needs to fill in all the information and the book will filter it out and give you the right book you need.

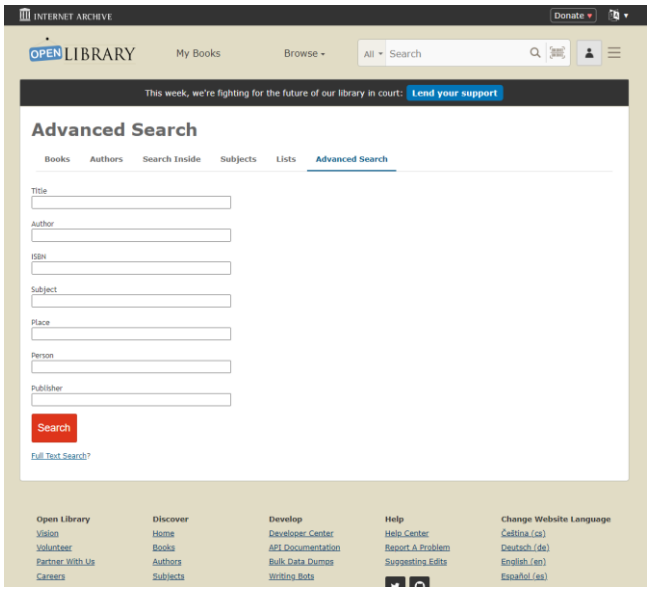


Figure 2.2: Open Library's advanced search [7]

After the user enters the keyword to search, Open Library will display the books with relevant information along with the current book status on the system such as being read by others, not yet updated on the system, or maybe read now. Besides, users can also add to their list for later viewing.

Users will be redirected to a page to read the e-book and have a limited time when selecting "Borrow Book". A special, unique thing about Open Library is that users can also choose to Listen to Book mode to let the machine read to users.

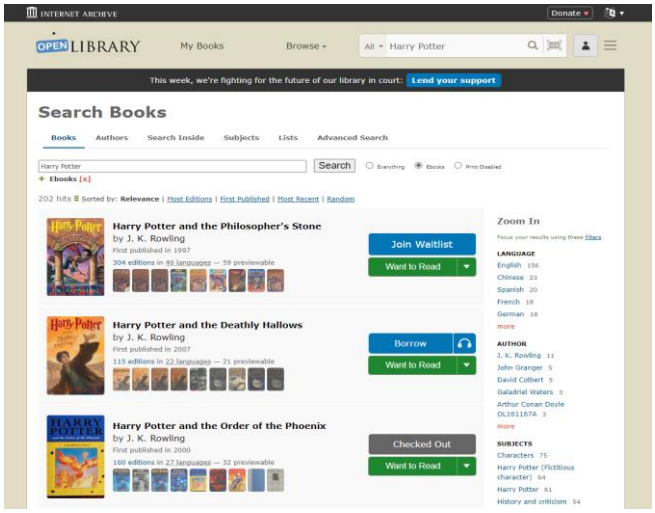


Figure 2.3: Books list after searching of Open Library [7]

2.3.2. IU Library

Unlike Open Library, a digital library for everyone in the world, IU Library [8] is a product developed by International University. This is a website developed and used since 2003.

The strength of IU Library is its focus on a collection of book content and educational data developed and collected over about two decades. All content in the library is focused on providing students and faculty of the school. This can be considered as an internal learning resource for International University.

In addition to the knowledge sources books, the library also synthesizes all the lectures of the school as well as the graduation theses of the graduates.

On the main page of IU Library, users will be able to see the search for books and faculties, and departments that the school is teaching so that when the user selects it, the system will go to the book sources, information about that faculty is available at in the library.

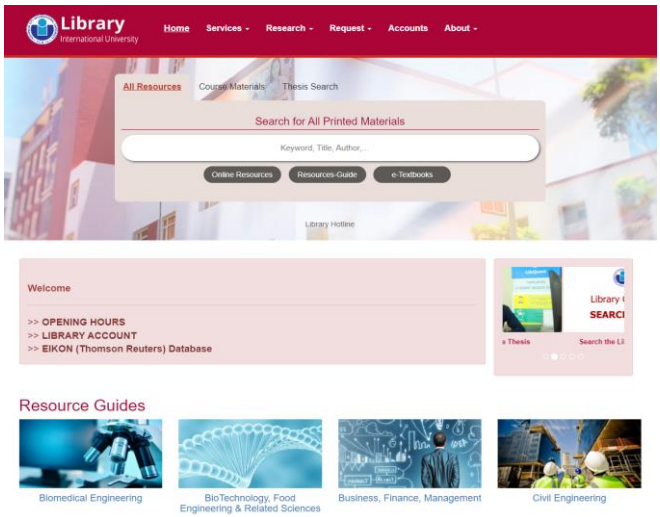


Figure 2.4: IU Library homepage [8]

Like Open Library, IU Library also has unique search support by entering information by author name, title, and field. When the user searches, the system redirects you to the list of books with information related to the search content of the National University Central Library.

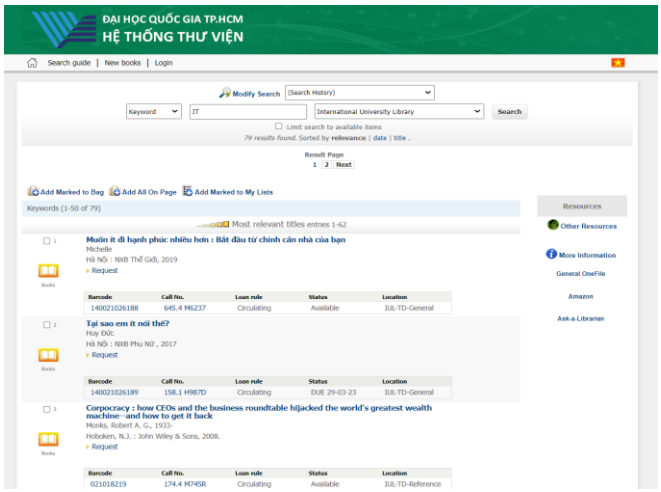


Figure 2.5: “Books list after searching” of IU Library [8]

The special thing is that the information about the book rights and the status of the book shows the user if the book is being borrowed and when it will be available to borrow again. Because the IU Library system is developed jointly with the system of the National University of Ho Chi Minh City library, it will include information about the location of the book in which school of the system.

This system's book lending is completely different at Open Library when it comes to the addition of physical copies and books available in the library. Users can make an appointment to book the website and have to pick it up directly at the school's library.

2.3.3. Shopee

Coming to the Shopee [9] website, will not be an example of a digital library, but the Finding Similar Image function that this e-commerce platform applies to their system.

Briefly talk about Shopee, this is a leading e-commerce platform in Southeast Asia and Taiwan. It was founded in Singapore in 2015 and has since expanded to several countries including Indonesia, Malaysia, the Philippines, Thailand, Vietnam, and Taiwan.

Shopee provides an online marketplace for buyers and sellers to conduct transactions, offering a wide range of products and services such as electronics, fashion, beauty, home and living, and many more. It also offers various features to enhance user experience, including in-app chat, secure payment options, and logistics support for sellers.

When coming to Shopee, the system displays trending product categories as well as product categories that users often search for in their search.

Another difference in Shopee's search compared to the two websites above is that they have added an image search function with a camera icon next to the keyword search bar.

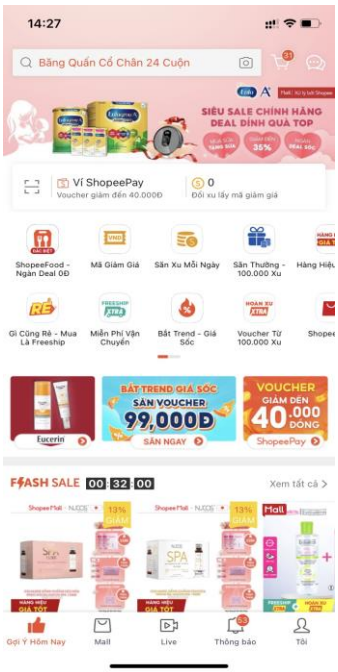


Figure 2.6: Homepage of Shopee Mobile App [9]

A special thing here is that this image-based search feature is only developed on mobile applications, and on the website, there is only a keyword search.

When users upload the image, they want to search on the system. The system will search and cut the main content of the image as shown below. In addition, users can also edit and cut the main content to be found on the uploaded image so that the system supports searching for you.

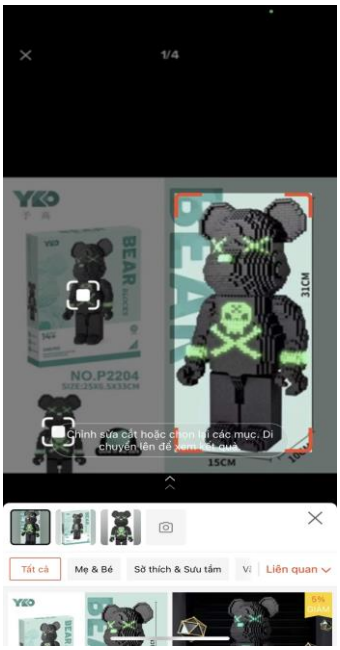


Figure 2.7: Result after searching by image [9]

Finally, Shopee will give a list of products with images that are most similar to the query image as well as basic information about the price, name, shop address, and the number of people who have purchased these products.

Due to the focus on Shopee's finding similar image features, the special online shopping features of this application will not be mentioned.

2.4. Background for Web-based Digital Library

Several core parts make up a typical web application. Here are some of the most important ones:

- Front-end: The front-end, or client-side, of a web application, is the part that the user interacts with directly. It typically consists of HTML, CSS, and JavaScript code that is executed in the user's web browser.
- Back-end: The back-end, or server-side, of a web application is the part that handles requests from the front end and performs any necessary processing before sending a response back to the client. It typically consists of server-side programming languages such as PHP, Python, or Node.js, and a database to store and retrieve data.
- Database: The database is where the web application stores and retrieves data. It can be a relational database like MySQL or PostgreSQL, or a NoSQL database like MongoDB or Cassandra.
- Web server: The web server is responsible for receiving and responding to requests from the front end and delivering static content like HTML and CSS files.
- APIs: APIs, or Application Programming Interfaces, are a set of rules that allow different software applications to communicate with each other. In a web application, APIs are used to enable the front end to communicate with the back end and retrieve data from the database.
- Security: Security is a critical part of any web application. It includes measures like SSL encryption, authentication, and access control to ensure that only authorized users can access sensitive information.

2.4.1. Front-end

2.4.1.1. HyperText Markup Language (HTML)

HTML [22], which stands for HyperText Markup Language, is the standard markup language used to create web pages and other online content. It provides a way to structure content on a web page by using a set of tags, which describe the content and its structure. These tags are interpreted by web browsers to display the content in a structured and readable format.

HTML was first introduced in 1991 by Tim Berners-Lee, the inventor of the World Wide Web, and has since evolved into several versions, with HTML5 being the latest and most widely used version. HTML5 includes new features like improved multimedia support, semantic elements, and enhanced form controls.

HTML is a markup language and is not a programming language. It is used to define the structure and content of a web page, but cannot be used to create dynamic behavior or complex logic. For this reason, HTML is often used in conjunction with other languages like CSS for styling and JavaScript for interactivity.

2.4.1.2. Cascading Style Sheets

CSS, or Cascading Style Sheets [23], was developed by a group of individuals, rather than a single inventor. The first version of CSS, known as CSS Level 1, was created in 1996 by Håkon Wium Lie, a Norwegian web pioneer and current CTO of Opera Software, and Bert Bos, a Dutch computer scientist who was working at the World Wide Web Consortium (W3C) at the time.

CSS is a stylesheet language used to describe the presentation of web pages and other online content. It allows developers and designers to control the layout, typography, color, and other visual aspects of a web page or application, separate from the actual content.

CSS works by providing a set of rules, or styles, that are applied to HTML elements. These styles can be specified in a separate CSS file or included directly in the HTML code using a style tag or inline style attribute. CSS also includes selectors, which allow you to target specific HTML elements to apply styles.

Over the years, CSS has evolved into several versions, with CSS3 being the latest and most widely used version. CSS3 includes new features like media queries, which allow developers to create responsive web pages that adapt to different screen sizes and devices, and new layout modules like Flexbox and Grid.

2.4.1.3. ReactJS – Front-End Techonology

It allows developers to create reusable UI components that can be combined to build complex web applications.

ReactJS [24] was first introduced in 2011 and has since become one of the most popular JavaScript libraries for web development. It uses a declarative approach to programming, where developers describe the desired state of the UI and let React handle the updates automatically.

One of the key features of ReactJS is its use of a virtual DOM (Document Object Model) [25]. The virtual DOM is a lightweight representation of the actual DOM, and React uses it to

optimize updates and minimize the number of changes that need to be made to the actual DOM. This results in faster rendering and better performance.

2.4.2. Back-end

2.4.2.1. NodeJS – Back-End Technology

NodeJS [26] is an open-source, cross-platform JavaScript runtime environment that allows developers to build server-side applications using JavaScript. It was created by Ryan Dahl in 2009 and has since become a popular tool for building scalable, high-performance web applications.

NodeJS has a modular architecture, with a core set of modules and a large ecosystem of third-party modules available through the npm (Node Package Manager) registry. This makes it easy for developers to find and use the modules they need to build their applications.

NodeJS is often used in conjunction with popular web frameworks like ExpressJS and NestJS, as well as with databases like MongoDB and MySQL. It also supports other programming languages through addons, allowing developers to build applications using C++, Python, and other languages.

One of the main benefits of NodeJS is its ability to handle large amounts of data and traffic with high performance and scalability. This makes it a popular choice for building real-time web applications, chat applications, and streaming applications.

2.4.2.2. JSON WebToken

JSON Web Token (JWT) [27] is an open standard for creating and verifying tokens that securely transmit information between parties as a JSON object. It is often used for authentication and authorization purposes in web applications.

A JWT consists of three parts: a header, a payload, and a signature. The header contains information about the type of token and the algorithm used to sign the token. The payload contains the data that is being transmitted, such as user information or authorization details. The signature is used to verify the authenticity of the token and ensure that it has not been tampered with.

JWTs are often used for authentication in web applications. When a user logs in, the server generates a JWT that contains the user's information and sends it back to the client. The client then stores the JWT and includes it in subsequent requests to the server. The server can then verify the authenticity of the token and use the user's information to provide access to protected resources.

2.4.3. Database

2.4.3.1. MongoDB – Database Technology

MongoDB [28] is a cross-platform, open-source NoSQL database that provides high performance, high availability, and easy scalability. It was first released in 2009 and has since become one of the most popular NoSQL databases in use today.

MongoDB stores data in flexible, JSON-like documents, making it easy to work with for developers who are familiar with JavaScript. It provides a flexible data model that allows for dynamic and ad-hoc changes to the data structure, without requiring a predefined schema.

One of the key features of MongoDB is its ability to scale horizontally, which means it can easily handle large amounts of data and high traffic loads. It also provides built-in support for replication and sharding, which makes it highly available and fault-tolerant. Moreover, It also provides a cloud-based platform, MongoDB Atlas, which makes it easy to deploy and manage MongoDB databases in the cloud.

2.5. Machine Learning Model

2.5.1. Convolutional Neural Network

Convolutional Neural Network (CNN) is one of the widely used neural network architectures in deep learning for image processing and analysis. It is built on the idea of allowing a machine to learn features on large image datasets by dividing them into smaller features using convolutional layers applied to the input image. CNN models trained on large datasets like ImageNet have achieved high accuracy and are an important tool in image processing and computer vision.

The structure of CNN includes convolutional layers, pooling layers, and fully connected layers. Each layer plays an important role in learning features on image data and classifying them. Below is the detail of the structure of CNN:

1. Convolutional layer: This is the first layer in CNN, it performs convolution between a filter and the input image to create feature maps. These feature maps will find important features of the image such as edges, curves, colors, etc.
2. Activation layer: This layer is used to apply the activation function to the feature maps to activate their important features and remove unimportant ones.
3. Pooling layer: This layer is used to reduce the size of the feature map by taking the average or maximum value in each region of the feature map. This helps to reduce the number of parameters in the model and extract higher-level features.

4. Fully connected layer: This layer is used to connect all neurons in the previous layer to all neurons in the current layer, helping to classify the image. Finally, this layer will make a prediction about the object in the input image.

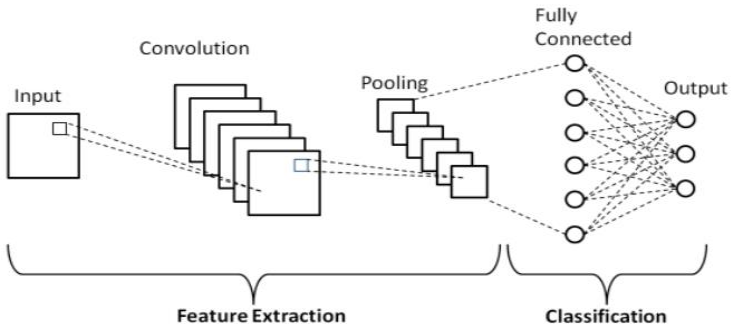


Figure 2.8: CNN Architecture [6]

Following the research [6], CNN is one of the most well-liked deep neural networks, and has achieved good results in applications related to pattern recognition such as image processing and speech recognition.

Some examples of CNN applications:

- Face recognition: With the ability to learn and recognize facial features, CNN is used in face recognition applications, from security login applications to criminal identification.
- Image classification: CNN is used to classify images, from simple classification like dogs and cats to classifying types of flowers.
- Medical image analysis: CNN can be used to analyze medical images, from detecting tumors in X-ray images to detecting signs of Alzheimer's disease in MRI images.

2.5.2. Hyperparameters in CNN

In CNN, hyperparameters are parameters used to adjust the model's performance during training due to this paper [14]. The most important hyperparameters in CNN include:

- Activation function [15]: A non-linear function used to limit the output of a neuron within a certain range. The activation function helps the model learn non-linear relationships between inputs and outputs and plays an important role in determining the performance of the model. Popular activation functions in neural networks include:
    - The sigmoid function: maps input values to a range between 0 and 1, and is commonly used in binary classification models.
    - The tanh function: maps input values to a range between -1 and 1, and is commonly used in multiclass classification models.
    - The ReLU (Rectified Linear Unit) function: sets input values to 0 if they are negative and retains the input values if they are positive.

- The Leaky ReLU function: similar to the ReLU function, but returns a small positive value instead of 0 when the input value is negative.
- The softmax function: used in multiclass classification models to map input values to a probability distribution over the different classes.

Each activation function has its own characteristics and applications, the choice of the appropriate activation function can impact the model's performance.

- Dropout rate: Dropout is a regularization technique used in neural networks to reduce overfitting and improve the model's generalization ability. The dropout rate is the ratio of weights that are randomly dropped out during training. Specifically, during training, Dropout randomly selects some neurons in the network and removes them, based on a predetermined dropout rate. The dropout rate is usually chosen from 0.2 to 0.5, depending on the specific problem and complexity of the model. Dropout is often applied after the fully connected or hidden layer of the neural network. When the model is deployed, Dropout is not used, and all neurons in the network are retained.
- Learning rate: A critical hyperparameter in the machine learning model training process that determines the rate at which the model's weights are updated based on the error and gradient of that error. Choosing an appropriate learning rate value is important to achieve optimal results in the model training process. A low learning rate will lead to slower convergence of the model.
- Batch Normalization: A technique for normalizing input data used in neural networks to speed up model convergence and reduce the risk of overfitting. When training a neural network, the input values from different layers can have different scales and distributions, which can make training difficult. Batch normalization helps to normalize the input data, making it easier to train the model. Batch normalization is typically applied after each convolution or fully connected layer of the neural network.

2.5.3. Transfer Learning

Transfer learning is a technique of reusing a part or all of a previously trained model on a different dataset to solve a new problem. Instead of training the entire model from scratch, we keep the basic part of the model and modify the last part to fit the new problem. Transfer learning helps to save time and computational resources [18], as well as improve the accuracy of the new model, especially when the training dataset is small.

There are several methods to use transfer learning, including:

- Feature Extraction: This method takes in a model that has been trained on a large dataset and uses the learned features in this model to extract features from new images. Then, these features are fed into a new classification model to perform the classification task.
- Fine-tuning: This method also uses a pre-trained model, but instead of using the learned features, we will continue training the model on a new dataset to fine-tune the model's weights. However, the weights of the model are kept intact, and only a few last layers are retrained.
- Multi-task Learning: This method uses a pre-trained model to perform multiple tasks simultaneously. The model will be trained on all tasks at once, so we can leverage common features across tasks to improve the results.
- Domain Adaptation: This method addresses the problem when the training and testing datasets come from different sources. To solve this problem, we use a pre-trained model on a dataset from a different source and make changes to the model to adapt to the new dataset.

Each of these methods has its own advantages and disadvantages and depends on the specific context to choose which method to use in a problem.

2.5.4. VGG16

The Visual Geometry Group (VGG) research team at the University of Oxford in the United Kingdom has suggested a famous convolutional neural network (CNN) architecture in the field of image processing known as VGG16 [19]. This architecture's name is derived from the team name's abbreviation and the number of levels (e.g 16 layers).

Following Figure 2.9 in [16], the image analysis structure of VGG16 consists of 6 blocks with 5 blocks of Convolution layer + Max-pooling and 1 block of 3 fully-connected layers. There are 13 Convolution layers each with a 3x3 kernel in the first 5 blocks with the first 2 blocks having the structure “Convolution - Convolution”, and the next 3 blocks having the structure “Convolution - Convolution - Convolution”. After each block always have a max-pooling layer that shrinks the image size to 0.5. And the final block has 3 fully-connected layers to stretch the image, which analyzed smaller form in the last max-pooling layer, to 4096 - 4096 - 1000 nodes.

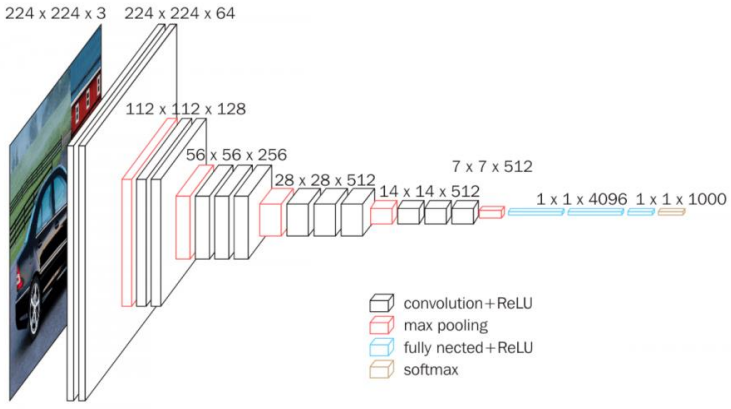


Figure 2.9: VGG16 Architecture [16]

2.5.5. Nearest Neighbor

k-Nearest Neighbor (kNN) is one of the simplest (and occasionally successful) supervised learning techniques. Since it does all computations when it needs to predict the result of new data rather than during training and does not learn anything from the training data, this approach is referred to as lazy learning. The kNN approach may be used to resolve both supervised learning issues requiring classification and regression [13]. It is also known as instance-based or memory-based learning algorithms.

kNN is a popular classifier when it comes to classification. Data is categorized using local or adjacent training examples in a certain location. This method is used because it computes quickly and is simple to use. It uses the Euclidean distance, which has the formula represented in Figure 2.10, to identify its nearest neighbors when working with continuous data.

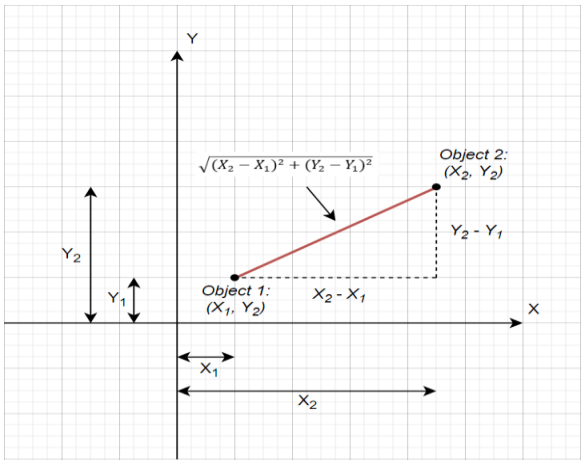


Figure 2.10: Euclidean distance [12]

Euclidean distance is a poor metric in high dimensional space, according to this paper [12]. In general, high-dimensional data points are very different from one another. This lessens the relative distance difference between a particular data point's closest and furthest neighbor. Therefore, in this thesis, the use of a model to extract features of the image and then convert it

to vectors is suitable for applying Euclidean distance to find the images with the closest vector to the input image.

2.6. Related work

2.6.1. Neural Network for Content-based image retrieval

In this paper [31], the authors argue that finding similar images from an large dataset is a challenging problem for the computer vision research community. Having a search engine that retrieves images based on image features will help users in addition to traditional text-based approaches such as annotations. Therefore, the authors have proposed deep neural network models that can be used in CBIR including CNN, Autoencoders, and Generative Adversarial Networks (GAN). They analyze the model structure as well as the advantages and disadvantages of each method.

Go deeper into the detailed application of Deep Convolutional Neural Networks - CNNs in CBIR system. Muhammad Imran Razzak has provided about architectures, layer structures, and notable techniques of popular CNN structures [33] such as: LeNet, AlexNet, VGGNet, ResNet, etc. The author has given the different methods and structures of CNN that can be applied to the CBIR system as well as the advantages and disadvantages of each used structure.

Besides image feature retrieval methods, distance measure is also an extremely important step in CBIR system. A. Alzu'bi, A. Amira and N. Ramzan introduced and analyzed popular distance measurement methods that can be applied to CBIR system [32] such as: Euclidean distance, Cosine similarity (calculate the angle of the two vectors), Chi-square distance (calculating the difference between the frequency distributions of image features), etc. They argue that there are many methods applied to train a model with good performance depending on the model structure and image dataset.

2.6.2. Summary

Related works	Features	This thesis	Description
Latif, A., Rasheed, A., Sajid, etc [31]	Convolutional Neural Network (CNN)	☒	There are many methods mentioned and analyzed in the article, but this project will only apply CNN to extract features of image data (see Section 4.2.1).
	Autoencoders	☐	
	Generative Adversarial Networks (GAN).	☐	

Muhammad Imran Razzak [33]	LeNet	☐	CNN structures are introduced a lot, but VGGNet will be an easily accessible and suitable choice for transfer learning for beginners. (see Section 4.2.1)
	ResNet	☐	
	VGGNet	☒	
A. Alzu'bi, A. Amira and N. Ramzan [32]	Euclidean distance	☒	Only the Euclidean distance method is used to measure the distance of image features drawn on the 2D axis. (see Section 4.2.1)
	Cosine Similarity	☐	
	Chi-square distance	☐	

Table 2.1: Related works summary

Chapter 3 : METHODOLOGY

3.1. Overview

As the information has been introduced before, the website only has similarities between the user, and the admin with the system, and each party will have the same functions, but the admin will be licensed to do more functions. The first and most important will be the user, here will be the students in the school, they can find the books by any information and date that they want to borrow and even the cover image of the books. After that, the system will give a list of books including input information which students are looking for with the information of that book such as title, author, publication year, and publisher,... and can proceed to borrow books later. However, booking will only be allowed when the user has logged into the system with the account provided by the admin (school), students will know which books can be borrowed and which can not decide to borrow a book. As for the “admin” accounts, which can be seen as librarians, they have right to access to the “Admin Dashboard” to view, create, and delete accounts, books, book slots, as well as similar functions developed for student accounts.

Besides developing a website "Digital Library", I will also apply a Machine Learning model for the image search function to the system which will be developed mainly based on CNN's VGG16 algorithms for the feature extraction of the dataset image and the input image from the students, the formula from Nearest Neighbor to compute the images with the most similar features.

To develop both a digital library simulation website as well as apply a machine learning model in finding images of book covers that match the input, it is necessary to clearly define how to design a whole system. The system will be built as figure below:

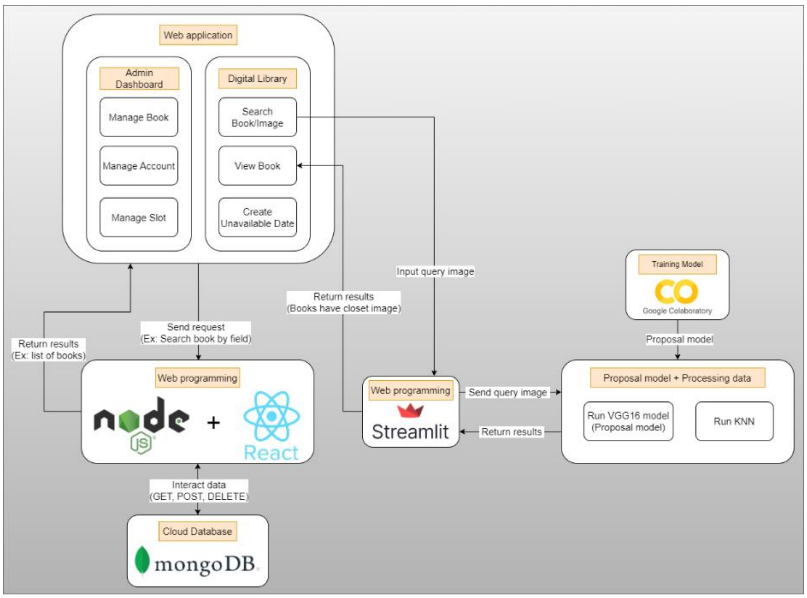


Figure 3.1: Digital Library System Tier 2 Diagram

- Developing web-based Digital Library:
    - Cloud Database: The database of a whole digital library will be managed by MongoDB. Storing all the data of the digital library in the Cloud of the mongoDB account will help the local machine reduce the processing of too large amounts of data and make it easier for developers to access and manage. Developers will be able to directly access and change data easily when they have an account logged into the Cloud Database associated with the web application.
    - Web programming: ReactJS and NodeJS are two popular tools used to build a webiste platform for any field. In this project, ReactJS is used to build the user interface and interaction with the interface. Besides, NodeJS is a tool for handling tasks from client and interact with database to meet user requirements.
    - Web application: The system will be simulated and displayed directly on Microsoft Edge, the default browser is used in the local machine. This is a browser that displays all that has been developed from "Web programming", running all functions for the systems, and receives interaction from users. Besides, this is also the platform used to launch the image search model through Streamlit.
- Developing Machine Learning Model
    - Training Model: The environment used to build the machine learning model is Google Colaboratory when it can quickly process the training and store the data used to train the model easily and does not require the local machine to be good enough to develop.
    - Proposal model + Processing data: This is the step used to process the data that will be used for the Web Digital Library, the data is collected in Excel format and processed into an image form with labeled information. Then use VSCode to run the proposal model and KNN algorithms with the selected data.
    - Web programming: Streamlit is used as a tool to develop a machine learning model for finding similar images to website. This is the foundation that allows a model to be built and interacted with users like web applications.

Besides, the interaction and linkage between the components of the system is also very important:

- For the “Develop web-based Digital Library” part, users will interact directly on “Web-application” both Digital Library and Admin Dashboard for admin, all interactions will be recorded and transmitted to “Web programming” for handling. The processing here

will include navigating the website, displaying the requested information. To be able to display the data that the user needs, “Web programming” will access and interact with the data stored in “Cloud Database” and return the results to the user in “Web application”. For instance, when people want to see the list of books with selected information, the “Web application” will send the request for “Web programming”, then it will call API “POST” to get all books of selected information from “Cloud Database” and return the list for the user on “Web application”.

- For the “Developing Machine Learning Model” part, the proposal machine learning model will go through the Streamlit platform of “Web programming” to display to the user in Edge, the user will enter the image to be searched and the system will send a request to run the model in “Proposal model + Processing data” and return the results to the user in “Web application”.

3.2. User requirement analysis

There is Digital Library for users and Admin Dashboard for admin, each site will have a different function. For Digital Library is a website that allows interactive user accounts to search, view book information and borrow books. As for the Admin Dashboard, it will allow admins to manage data such as accounts, books and book locations. So, the process for both sites for accounts would be as follows:

Only admin and student (normal account) will be the main users for the system as follows:

- Student (users): users can access the digital library website to be able to search for the book they need based on information such as field, year of published, title, and author, besides the image is further developed. In addition, users can also see if the status of the book is borrowed or not on the days, they want to borrow it to avoid the situation of going to the library to find books that have already been borrowed by other friends. In addition, users need to log in to the system to be able to see the status of the remaining books, whether they can be borrowed or not, as well as proceed to borrow the book.
- Admin: this is also a user account that can do all the benefits of a normal account, but besides that, it is also granted permission to log in to the admin dashboard. In the admin dashboard, admins can check book information, accounts, number of books, and orders, etc. In addition, you can also create new accounts and books, but you must enter the correct requirements of the book, each content that the system requires in each new content creation.

Both types of accounts need to be granted by an admin and will only be provided to students like what is available in the system of educational accounts that IU students are granted in the first year. All information will be created by the admin and users do not have the right to know and access information of other accounts.

The additional function as mentioned, when users need to search by images, the system will redirect to a page containing a finding similar image model that has been developed before. After the user provides an input image, the system will return results with a similar book cover image with their book's information. Users will then rely on this information and return to the Digital Library to search by information.

The main foundation for finding similar image functions is to build on the existing VGG16 model to train a new image feature extraction model, and from that to analyze the images contained in the database used for the website as well as for the user input image. After extracting the image features in the database, these features will be saved along with the labeled information for the books. Finally, compare the distance of the stored features with the feature of the query image and find the books with the cover image that most closely matches the book that the user wants to search.

3.3. Use Case and Analysis

3.3.1. Use Case and Description

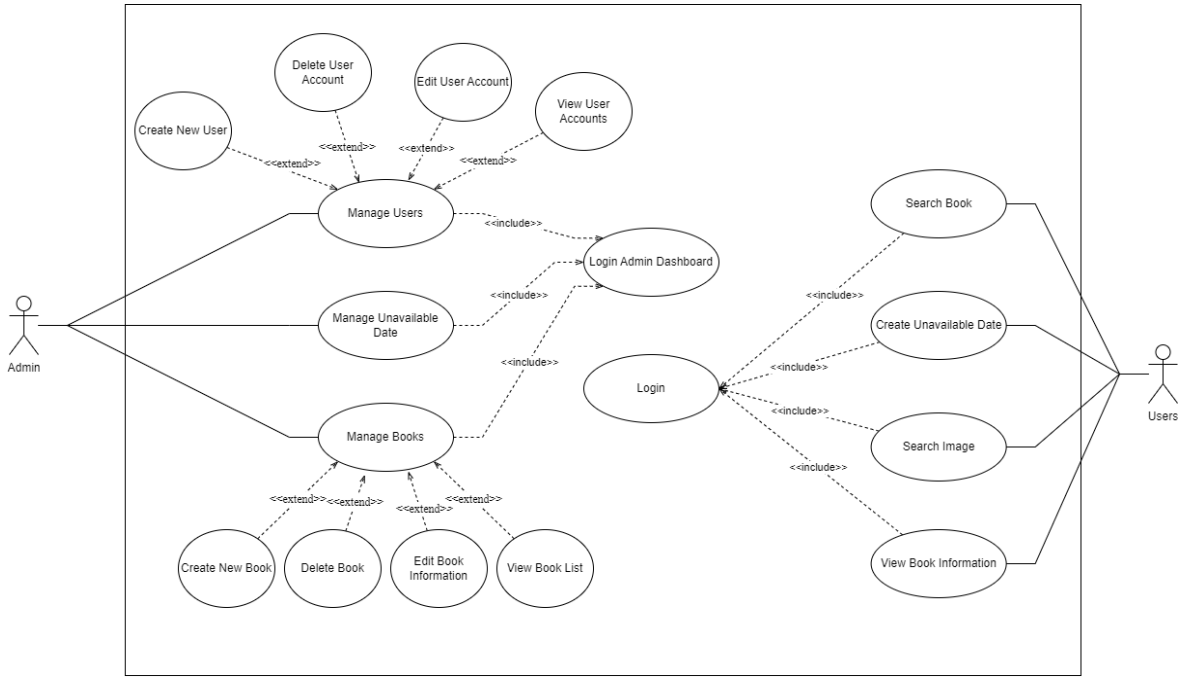


Figure 3.2: Digital Library's Use Case Diagram

Name	UC-1: Search Book

Actor	Admin/User
Preconditions	(none)
Trigger	Admin/User wants to search book by book's information.
Postconditions	Admin/User see list of books by request category.
Main Scenario	1. Admin/User enters the information of a book they want to find. 2. Admin/User chooses the date they want to borrow the book. 3. Click "Search". 4. The systems show the list of books based on the requested category.
Extensions	(none)

Table 3.1: Description for "Search Book" use case

Name	UC-2: Search Image
Actor	Admin/User
Preconditions	(none)
Trigger	Admin/User wants to find books by book's image.
Postconditions	Admin/User sees the list of books whose cover image is closest to the input image.
Main Scenario	1. Click the "Camera icon". 2. Admin/User chooses the image they want to find. 3. Admin/User chooses the date they want to borrow the book. 4. Click "Search". 5. The systems show the list of books based on the input image.
Extensions	(none)

Table 3.2: Description for "Search Image" use case

Name	UC-3: View Book Information
Actor	Admin/User

Preconditions	Admin/User search book by book's information/image.
Trigger	Admin/User see suggested book list.
Postconditions	Admin goes to the Book's Information page.
Main Scenario	1. Admin/User chooses a book they want to see more information about when they view the list of books. 2. Click the "See availability". 3. The systems move to the Book's Information page.
Extensions	(none)

Table 3.3: Description for "View Book Information" use case

Name	UC-4: View User Accounts
Actor	Admin
Preconditions	Admin logged in the Admin Dashboard.
Trigger	Admin wants to check/edit the user list.
Postconditions	A list of user accounts is shown.
Main Scenario	1. Click "Users" on the Sidebar. 2. The system shows the list of all user accounts in the database.
Extensions	(none)

Table 3.4: Description for "View User Accounts" use case

Name	UC-5: Delete User Account
Actor	Admin
Preconditions	Admin logged in and views user accounts.
Trigger	Admin wants to delete a user account.
Postconditions	A selected user account is deleted and updated in the database.
Main Scenario	1. Admin chooses a user account that they want to delete.

	2. Click the “Delete” button on the account they want to delete. 3. The system deletes that account and saves it to the database.
Extensions	(none)

Table 3.5: Description for “Delete User Account” use case

Name	UC-6: Edit User Account
Actor	Admin
Preconditions	Admin logged in “Cloud Database” and views user accounts.
Trigger	Admin wants to edit a user account’s information.
Postconditions	Information of the selected user account is edited and updated in the database.
Main Scenario	1. Admin chooses a user account that they want to edit. 2. Admin updates new content for what account information they want to change. 3. Click “Update” and the system will save it to the database.
Extensions	(none)

Table 3.6: Description for “Edit User Account” use case

Name	UC-7: Create New User
Actor	Admin
Preconditions	Admin logged in and views user accounts.
Trigger	Admin wants to create a new user account.
Postconditions	A new user account is added and added to the database.
Main Scenario	1. Click the “Add New” button. 2. The system moves to an Add New User page with a form for Admin to create a new user account. 3. Admin enters content on all information of a user account that they want to create.

	4. Click “Ok” and the system will create and save it to the database.
Extensions	(none)

Table 3.7: Description for “Create New User” use case

Name	UC-8: Create Unavailable Date
Actor	Admin/User
Preconditions	Admin/User logged in and chose the book.
Trigger	Admin/User wants to borrow a book.
Postconditions	Unavailable dates are set for a book. (No one can not choose)
Main Scenario	1. Click “Reserve or Order Now!”. -- Check if Admin/User logged in, then it goes to step 2. 2. The system shows the information sheet of the number of books. 3. Tick on the box of books you want to borrow. 4. Click “Order”. 5. The system saves the unavailable dates to the database.
Extensions	1a. If Admin/User did not log in, return to the Login use case.

Table 3.8: Description for “Create Unavailable Date” use case

Name	UC-9: Manage Unavailable Date
Actor	Admin
Preconditions	Admin logged in “Cloud Database” and viewed the information of books.
Trigger	Admin wants to edit the unavailable date of a book.
Postconditions	An unavailable date of a slot is changed/deleted.
Main Scenario	1. Admin chooses a slot that they want to edit on an unavailable date. 2. Input new unavailable date or delete existing unavailable date. 3. The system saves it to the database.

Extensions	(none)

Table 3.9: Description for “Manage Unavailable Date” use case

3.3.2. Sequence Diagram

This section only displays sequence diagrams for some important use cases, the rest of the remaining use cases that are not shown in this part are placed in Appendix.

UC-11: Login Admin Dashboard (see use case in Appendix 2)

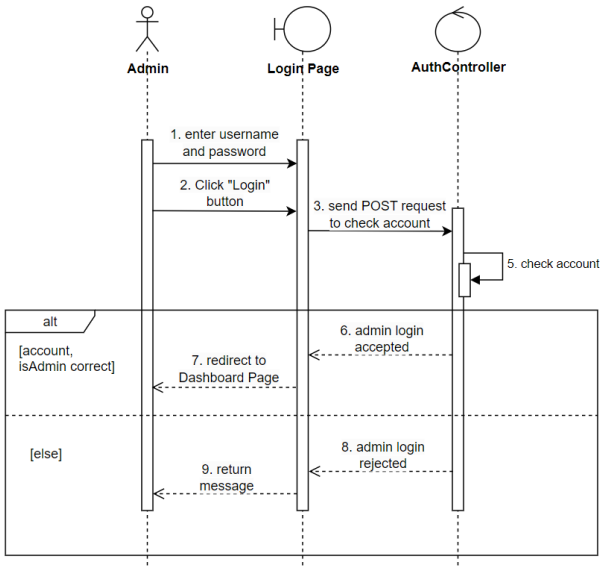


Figure 3.3: Sequence diagram for “Login Admin Dashboard”

Figure 3.3 shows that this use case is required when wanting to use the Admin Dashboard because only accounts that are authorized isAdmin=true have login permission. The system will always display the login page first, Admin is required to enter the login username and password. Then, the system will use the input username and password to use the test API, if both the username, password, and isAdmin attribute = true, the admin will be redirected to the Dashboard page. If 1 of the 3 variables has an incorrect result, the system will refuse to login and display a message to the user.

UC-1: Search Book

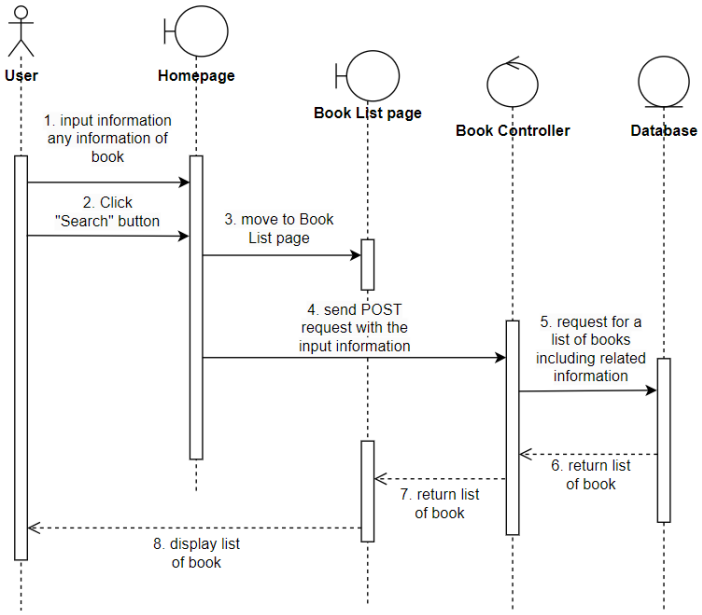


Figure 3.4: Sequence diagram for “Search Book”

Figure 3.4 shows that this use case can be used without forcing the user to log into the system. There will be a box for the user to enter the information of the book and click the “Search” button so that the system can search for the right book according to the user's input information. The system will redirect the "Book List Page" page as well as send an API request to get data from the database and display the book to the user.

UC-8: Create Unavailable Date

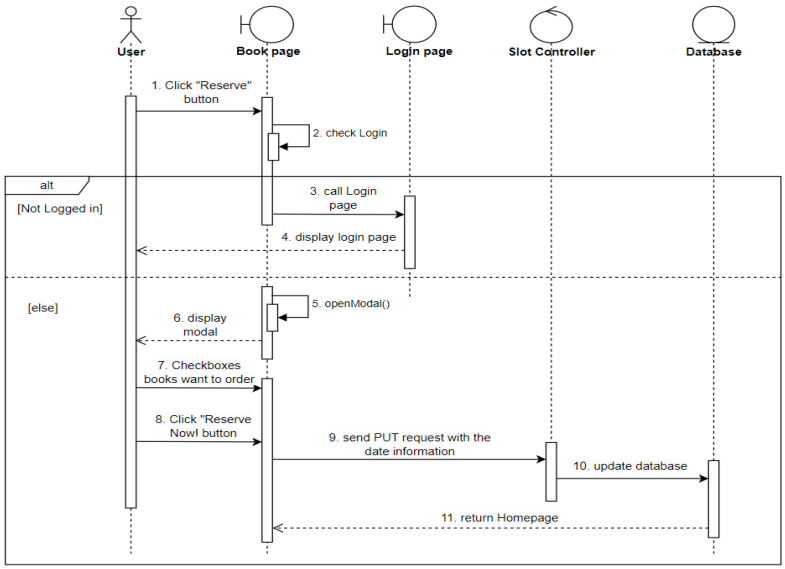


Figure 3.5: Sequence diagram for “Create Unavailable Date”

Figure 3.5 shows that this use case will be completed only after the user has logged into the system. If the user is not logged in yet but clicks the “Reserve” button, the system will automatically switch to the “Login page” to ask the user to log in. If the user has logged into the system, the system will open a small table showing the number of books that the user wants

to borrow, here the user just needs to checkbox(es) of the book that the user wants to borrow, then press the button “Reserve Now!”. The system receives the request and updates the user's borrowing time into that book's information in the database and redirects back to the homepage.

UC-5: Delete User Account

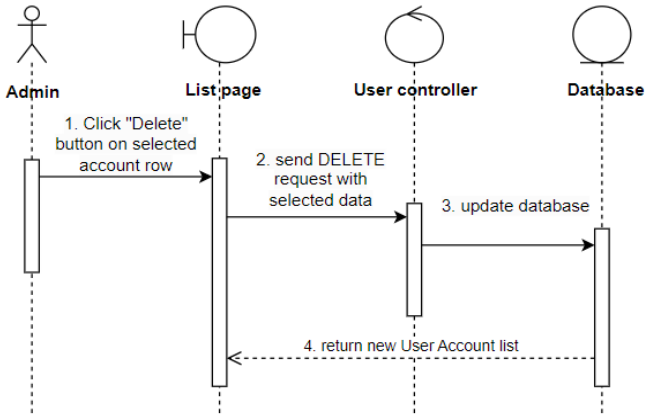


Figure 3.6: Sequence diagram for “Delete User Account”

Figure 3.6 shows that this use case will only take place when the Admin is logged into the Admin Dashboard. Admin is looking at the list of accounts and intends to delete an account, they just need to click the "Delete" button displayed at the bottom of each account. Then, the system sends a DELETE request to the User Controller and deletes that account from the database, and updates the list of accounts on the system.

UC-7: Create New User

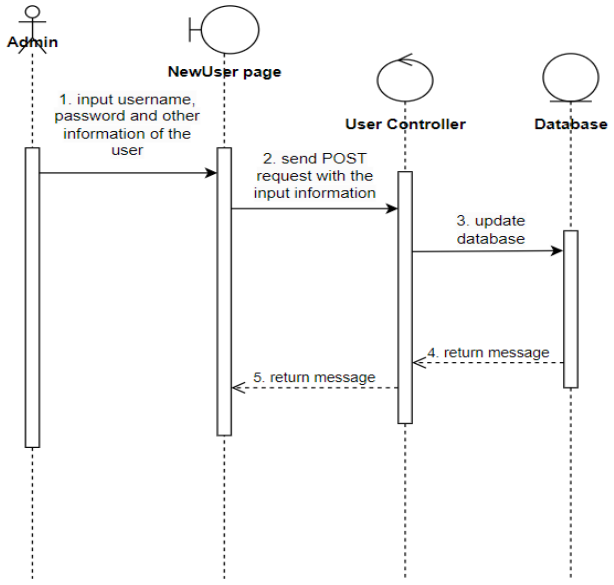


Figure 3.7: Sequence diagram for “Create New User”

Figure 3.7 shows that this use case will only take place when the Admin is logged into the Admin Dashboard. Admin will have to fill in the information that the system requires to

create a new account and press the “Create” button when completing the form. The system receives the POST request and proceeds to create an account as well as update the list of accounts on the system.

3.3.3. System workflow

3.3.3.1. Create Unavailable Date

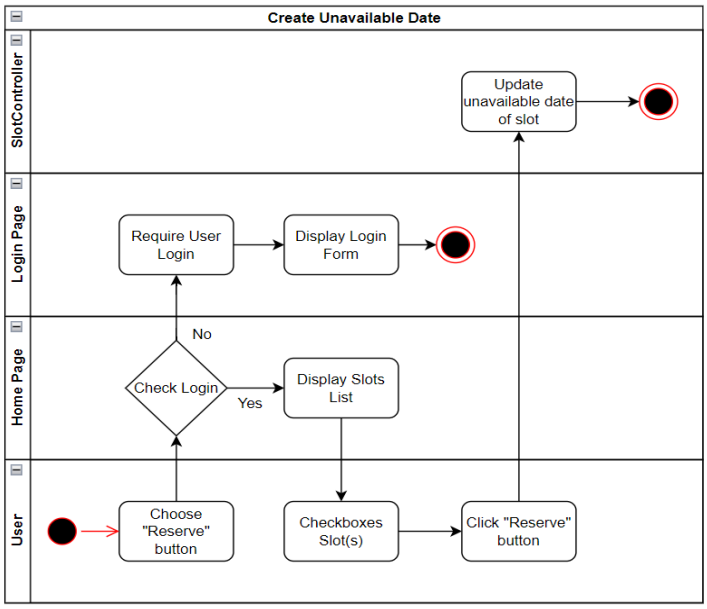


Figure 3.8: Create Unavailable Date workflow

The function to create an unavailable date is as follows: The user will choose to borrow a book on the book information page, and the system will check if the user is logged in. If the user is not logged in, the system will redirect the user to the login page, and if the user is already logged in, the system will display a list of book slots. Users will tick the checkboxes where they want to book and select "Reserve", the system will update the borrowed date information into the book's slot information so that other users can not choose that book slot anymore.

3.3.3.2. Search Book

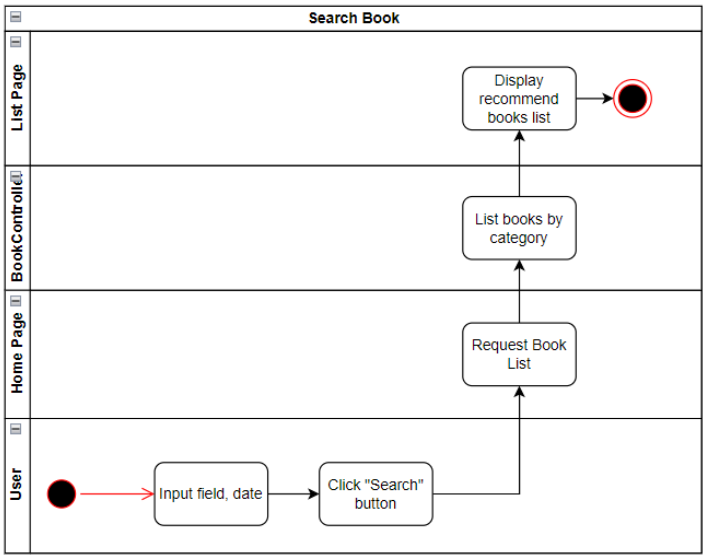


Figure 3.9: Search Book workflow

The function search book is as follows: Users will have 2 options to search for books they need including book information or search by book cover images. For searching by information, users will have to fill in any information of books to search and select the dates to borrow. This method will send a request to the system to process when the user presses the "Search" button, the request will be sent to the BookController to perform the filtering and search APIs and redirect to the List Page displaying books that are recommended based on the information the user needs to find.

3.3.3.3. Manage User Account on Admin Dashboard/Cloud Database

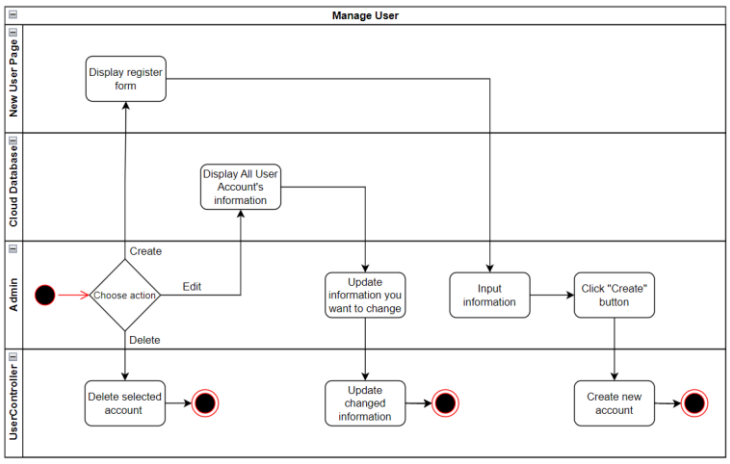


Figure 3.10: Manage User workflow

User account management functions are as follows: Admin has 3 options on the user list page of Admin Dashboard: delete, create new, and edit. In the delete option, the system will call the API to delete the account by id after the admin presses the "Delete" button. If admin want to edit existed user account's information, they need to login the Cloud Database, the

system will display a table containing the user's previously created information so that the admin can edit the information, the admin updates the information he wants, then clicks the button “Update”, the system will update the edited information. Finally, if the admin choose to create a new account, the system will transfer the admin to the page to create a new user account, where a form is available for the admin to fill in the necessary information to create an account, then an account will be created after the user clicks the “Send” button.

3.3.4. ERD Diagram

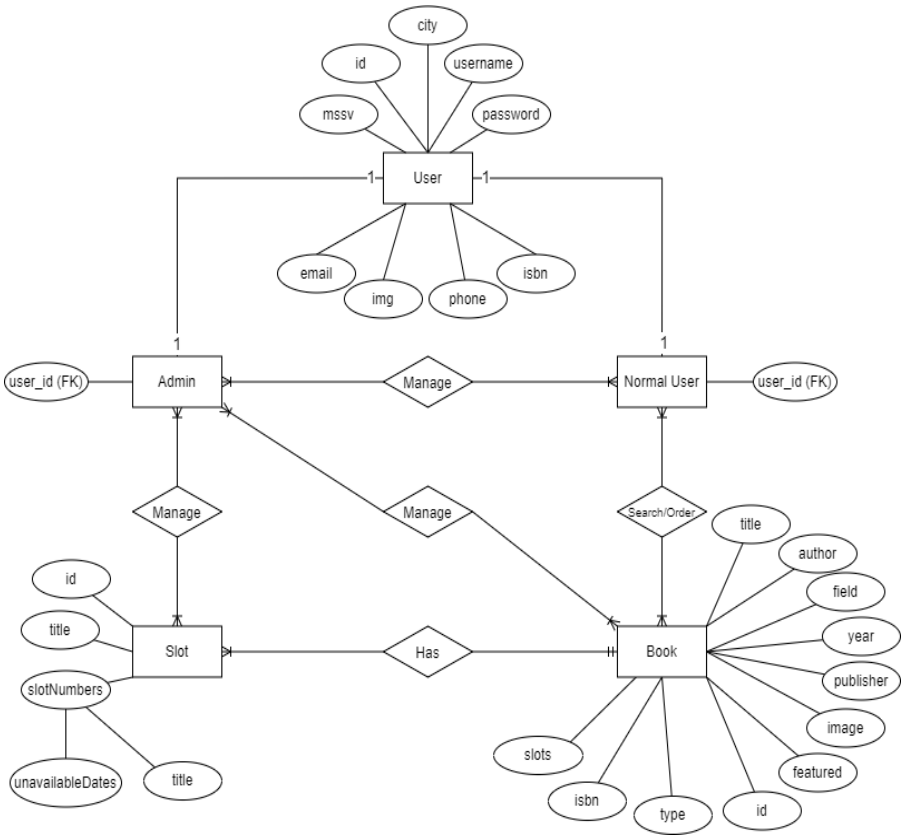


Figure 3.11: Digital Library Entity Relational Diagram

The User entity of the system will have 9 specific fields including id, username, mssv, password, email, phone, img, city, and isAdmin to distinguish between regular users and admins. For the User entity, the id will be the primary key as it will be unique and immutable for each user account in the system. Besides, username, email, and mssv are 3 fields that must be set as unique and can only have one such value in the whole system (no repetition). The img field is used to save the image representing the user if the admin puts it in. Phone and city are additional information added to the account. There is a special field here, isAdmin, which is used to determine which accounts will be normal users, and which accounts have access to the Admin Dashboard to manage the system.

Since the user account is only hierarchical to Admin or Normal User using the isAdmin field, these two entities will also have a primary key user_id which is a foreign key inherited from

the User entity. Besides, Admin and User have a many-to-many relationship as the accounts can be managed by one or more Admin accounts if needed.

There are 10 fields in the Book entity: id, isbn, author, title, field, type, year (published year), publisher, image, featured, and slots. With a feature is used to allow the book to be advertised on the homepage if the user wants it and the remaining fields will be the details of a book that students need. Slots field is used to store the all Slots id that a book has.

For the Slot entity, there will be 4 fields: id, title, slotNumbers, while id will be the primary key. The slotNumbers field will have “unavailableDates” to store the borrowed time of each book.

Book and Slot will have a one-to-many relationship as the books will have different quantities and each slot will represent 1 book for students to borrow on the system. These slots will be segregated because they must include a Book_id to be able to point to them.

3.3.5. Class Diagram

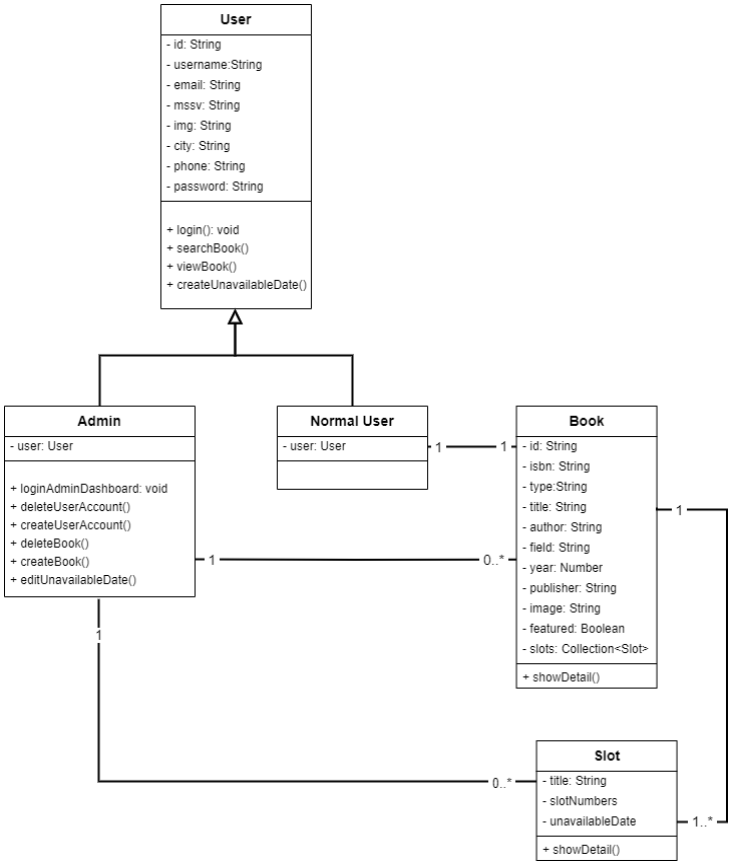


Figure 3.12: Digital Library Class Diagram

This system, shown in Figure 3.12, will have the following functions: Users will use the search book function, search by image to be able to search for the type of book they want to search and see the list of suggested books of the system. Users can borrow books but must login to the system to be allowed to view information about the remaining books to borrow. For the

Admin account, you can use the login function to the Admin Dashboard to manage the whole system such as managing user accounts, books, number of books, and borrowing date of each book. Admin's management of the system will include seeing all the lists of accounts and books and having the right to create new content or delete accounts and books when necessary.

3.3.6. User Interface Design

The figures below show the mockup user interface designed for the application

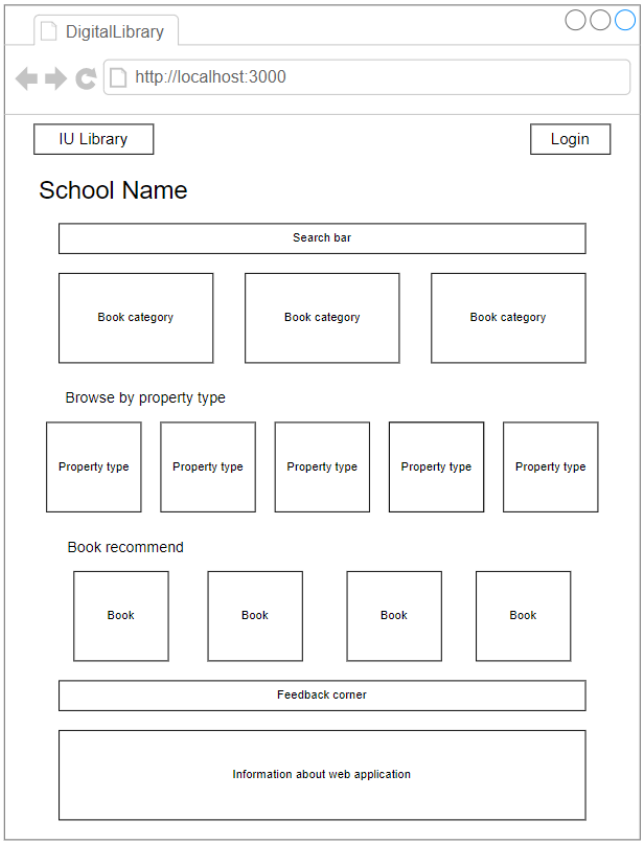


Figure 3.13: User interface of Digital Library Homepage

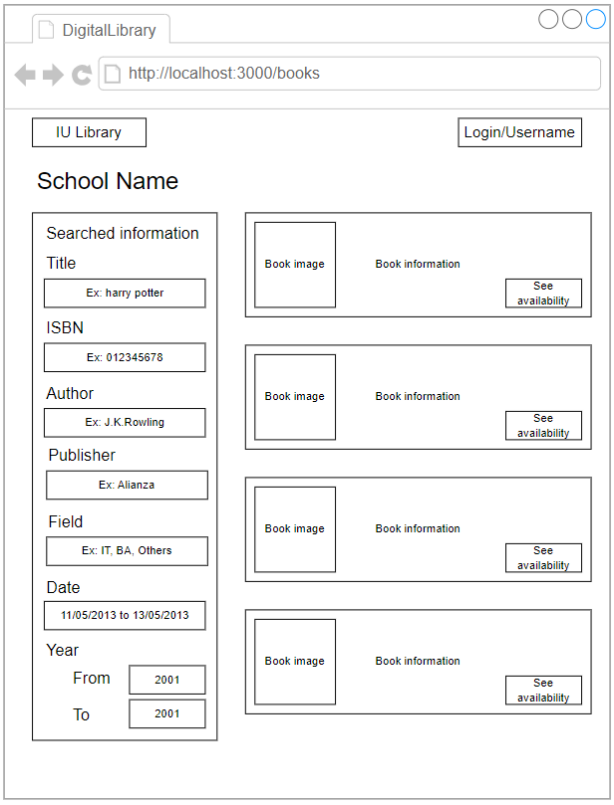


Figure 3.14: User interface of books's list page

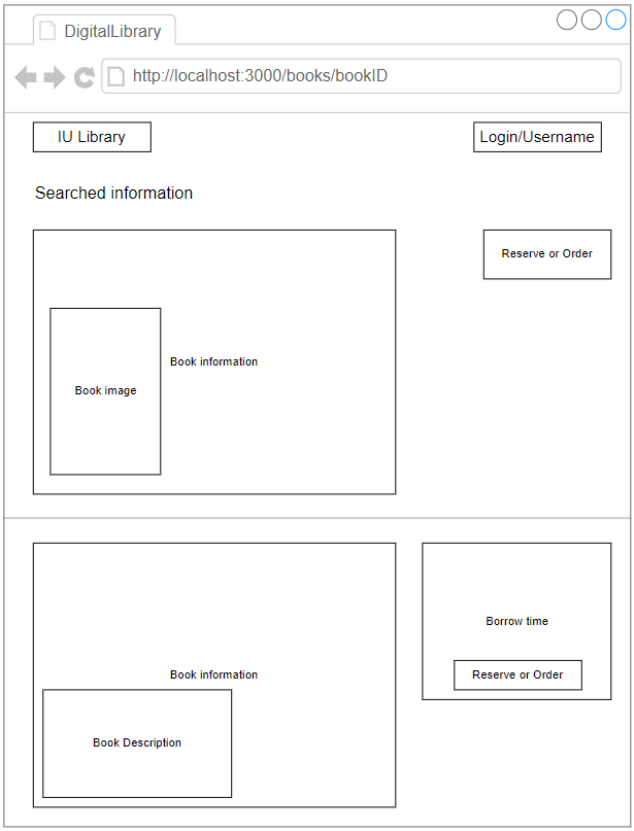


Figure 3.15: User interface of book page

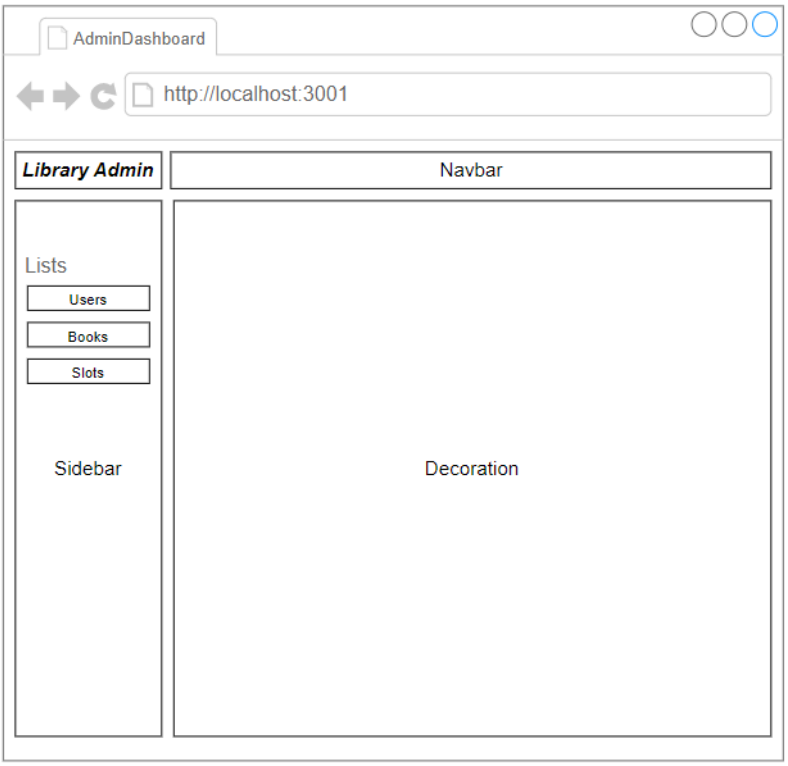


Figure 3.16: User interface of Admin Dashboard Homepage

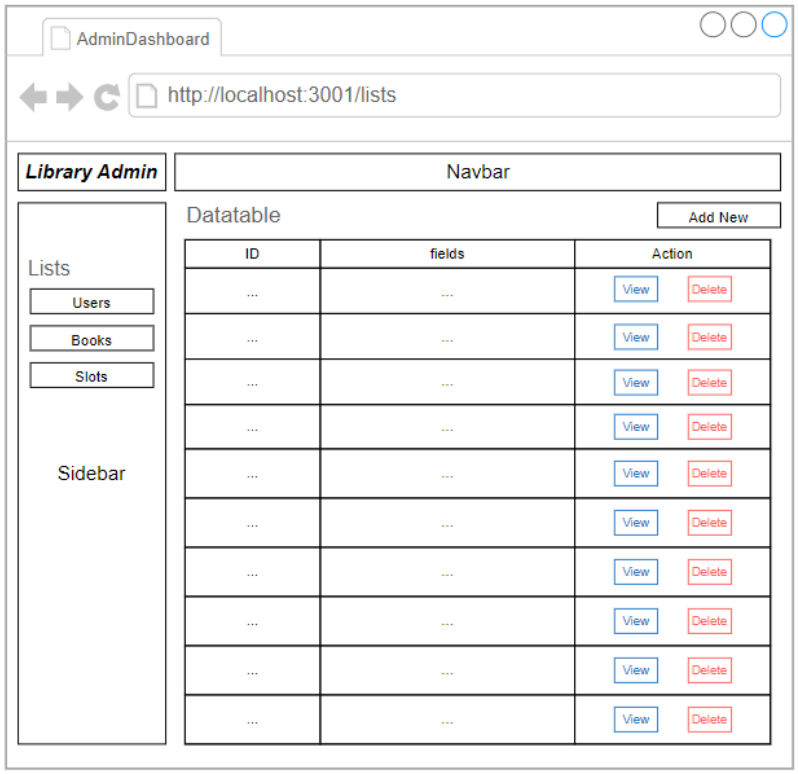


Figure 3.17: User interface of list page

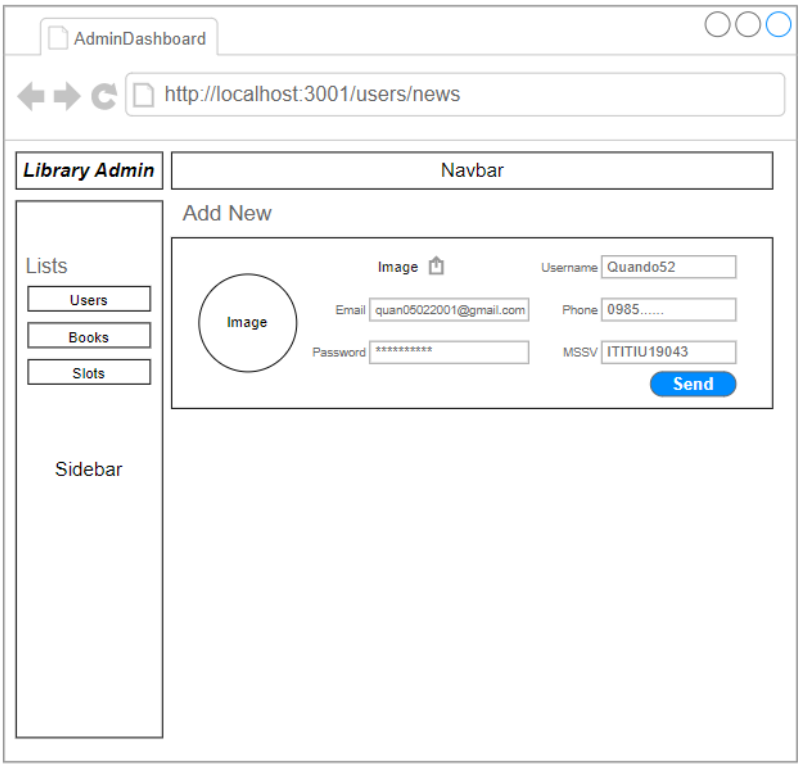


Figure 3.18: User interface of create new account page

3.4. Machine Learning Model

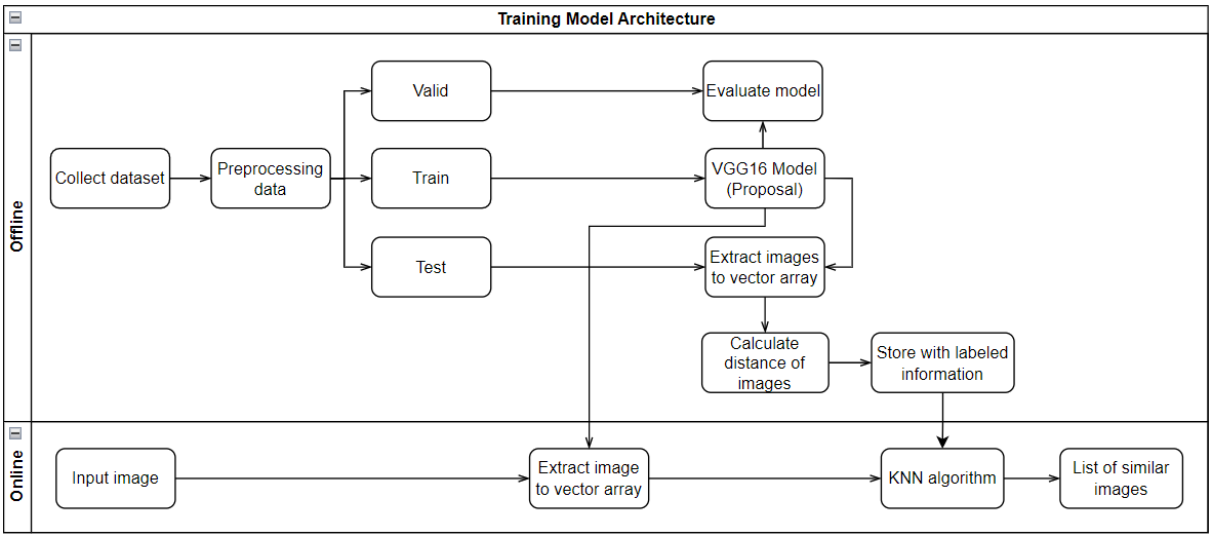


Figure 3.19: Pipeline system for Finding similar image using Neural Network

3.4.1. Model Architecture

The transfered learning model is based on the VGG16 pre-trained model architecture trained on the ImageNet dataset [29] with 14 million images. However, the model has been fine-tuned to fit the image classification task of "Book" and "Comic-book" with two output classes. First, the use of VGG16's layers (functional) remains unchanged to load the weights of ImageNet and extract features from the input image. Next, after the Max-Pooling 2D layer (see Section 2.3.1) is created, it is used to reduce the output size from VGG16 and take the maximum value in a

fixed window on the input. Additionally, a Flatten layer is used to flatten the output from the previous layer into a 1D vector to connect with fully connected neural networks. Two Dense layers are then used, with a Batch Normalization and Dropout layer (see Section 2.3.2) in between to randomly turn off some neurons during training to avoid overfitting. The first Dense layer has 2048 fully connected neurons using the ReLu activation function, while the final layer of the model has 2 fully connected neurons, representing the 2 categories, and uses the Softmax function to calculate the probability of each output class. Batch Normalization is used to normalize the output of the previous layer.

The model is trained using data generated by ImageDataGenerator and optimized by the Adam optimizer with a learning rate of 0.001, using cross-entropy loss for classification tasks. The model is trained for 200 epochs, and the ModelCheckpoint callback is used to save the weights with the best accuracy on the validation set. In addition, EarlyStopping is used to stop the training process if there is no progress on the validation set for 20 consecutive epochs. Finally, the accuracy and loss are saved during the training process and used to evaluate the performance of the model.

3.4.2. Dataset

To prepare a sufficient amount of data for both training a VGG16 CNN model and applying it to a digital library website, it is important to carefully select data, including images for training the model and labels for the book covers. Kaggle is a community platform for data scientists and machine learning experts around the world that provides resources and tools to help those interested in machine learning get started with their first model training. Two datasets, "Book Recommendation Dataset" [11] and "Comic Books Classification" [10] will be used to some extent for both model training and the data of the digital library website.

The "Book Recommendation Dataset" file contains information about 10,000 books rated by about 53,424 users on Goodreads, a popular dataset used in recommendation system-related projects. This dataset includes labels of book information along with book cover images provided by Amazon, which will be suitable for the digital library website data, and the images of each book cover will be used to train the VGG16 model along with images from the other dataset file. Meanwhile, the "Comic Books Classification" file contains information about comic books and their main characters. This dataset includes more than 23,000 images with 12 different labels, including famous characters such as Batman, Spider-Man, and Superman. The dataset has a considerable number of images and focuses on comic book types, so the model data will be divided into two types: books and comics.

A portion of the data from both datasets will be used for both model training and digital library data. Since the datasets are not divided into train, valid, and test folders, only a sufficient amount of data will be used for these three different folders. For the image data in the two datasets, over 4000 images from both datasets will be used for training and more than 1300 images for validation. For training the CNN, the image dataset used needs to include class folders containing images representing that class, so there will be two main classes: "Book" and "Comic-book". Therefore, there will be almost 2000 images for each category in the training folder and more than 600 images for each category in the test and validation folders.

Chapter 4 : IMPLEMENT AND RESULTS

4.1. Web application development

All important steps such as building front-end and back-end are developed and tested on personal computers. In addition, the VGG16 machine learning model is built and trained using Google Collab to perform faster but then the test data processing and testing part is done on personal computers. For the Database, all data is stored using MongoDB's Cloud and connected to Front-end, Back-end using the "Mongoose" library to connect and interact between the website and the database stored on MongoDB through Node.js. In addition, Front-end development uses the React environment and its supporting libraries such as react-date-range, react-dom, and react-router-dom, and uses CSS for a more beautiful interface design. The backend is developed using Node.js along with other support libraries such as bcryptjs, cookie-parser, cors, and JSON web token. In addition, Axios is also used to be able to handle requests from the website and process them through the API created in Back-end.

For the system to work smoothly, the arrays must be connected to support and transmit requests as well as extract data on the database to display on the system. First, the database will be stored in Cloud MongoDB and will be accessed from Back-end by using the "Mongoose" library through Node.js. Next, when the front-end receives requests to view data, create and delete data will be passed HTTP into the back-end through the "Axios" library to get user data needed and interact with it. In addition, a special feature in Admin Dashboard will that is creating data by providing additional images such as avatars for users or illustrations for book information will be uploaded directly to the Cloud storage of "Cloudinary" and take that image link stored into an attribute of entities in MongoDB.

A digital library website will have basic functions that help students view book information and search, as well as librarians can control those who have access rights and the number of books. Therefore, the system will have some features as follows:

No.	Features	Details
1	Searching book by their information	Users can search for books based on its related information such as title, author, publisher, ISBN or field.
2	Select a date to find the book and borrow it	In addition to input the information, users can choose the period they want to borrow books on the calendar to see if the books can be borrowed during the time they choose or not.

3	Hide reserve button for borrowed slots	Books that have been reserved within a time cannot be borrowed by others if they also want to borrow during this period.
4	Different login permissions for Admin/User	Accounts will be assigned different permissions by the attribute isAdmin to distinguish which account is used for Admin Dashboard.
5	Search books by cover image	Users can search for books by book cover image using a model developed and receive suggested results, then use that information to find the book they need to borrow.
6	Have Admin Dashboard to control books, slots, and users	There is an Admin Dashboard that only allows Admin accounts to log in and manage the number of accounts and number of books, slots.

Table 4.1: Functions list of Digital Library

4.1.1. Implementation

4.1.1.1. Setting up

Both front-end and back-end are developed using React and NodeJS and programmed on Visual Studio Code, running a demo on a localhost website. The process of connecting a search model by image will also be demonstrated on another local host opened by Streamlit [21] to link between the website and the model.

4.1.1.1.1. Back-end

Before proceeding with software development, some libraries need to be added to the system to develop some necessary functions as well as install NodeJS for the Visual Studio Code programming IDE.

Figure 4.1: Libraries used to build Back-end

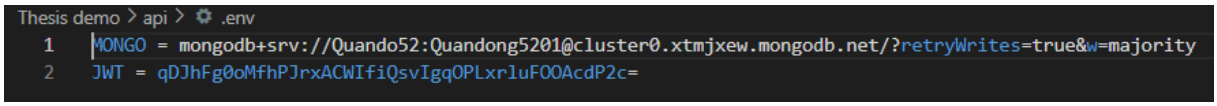


Figure 4.2: Connection path format

The libraries used to support the system to complete the functions include:

- Bcryptjs: This is a library that provides a method for encrypting passwords in Node.js applications to keep these passwords secure when stored in any database. In this system, after the account is created, the password will be stored in the Cloud MongoDB database with an encrypted form that even the owner of the right to edit the database cannot know.

Figure 4.3: Password has been encrypted by bcryptjs

- Cors: A middleware that allows the system to share resources between web pages such as APIs or images. Cors is used so that 2 localhosts Digital Library and Admin Dashboard can use data as well as use APIs to call for displaying and changing data with the interaction of both users and admin.
- Jsonwebtoken: This is a library for creating JWT (JSON Web Tokens) authentication codes in Node.js applications used to authenticate users. For the developed system, when users log into the system, the system will create a JWT string based on input

information to authenticate accounts on the database. After authentication, that Token string will be sent back to users as a cookie like "You are not authenticated!" or "Wrong password or username!" if login information does not match saved data.

- Mongoose: This is a library that supports connecting and interacting between MongoDB and Node.js applications. In this system, the Mongoose is used to link back-end programming with MongoDB storage on Cloud through the storage address shown in Figure 4.2. Therefore, any addition or change when programming will be updated on MongoDB without having to edit on Cloud and vice versa.
- Express: This is a framework that has one function in many things which is to define the path and determine that path performs functions with HTTP methods such as (GET, POST, PUT, DELETE,...). In this system, it will also define different paths, and each path will have a different function such as "localhost:8800/api/login" will be called when users log into the system and "loccalhost:8800/api/register" when the admin needs to create a new account.

4.1.1.1.2. Front-end

For the development of both the Digital Library and Admin Dashboard front-end, React environment will be used along with supporting libraries to perform basic functions for each website.

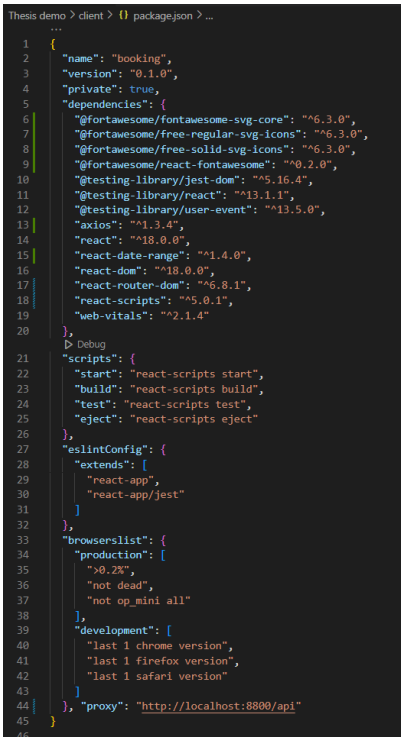


Figure 4.4: Libraries used to build Front-end of Digital Library

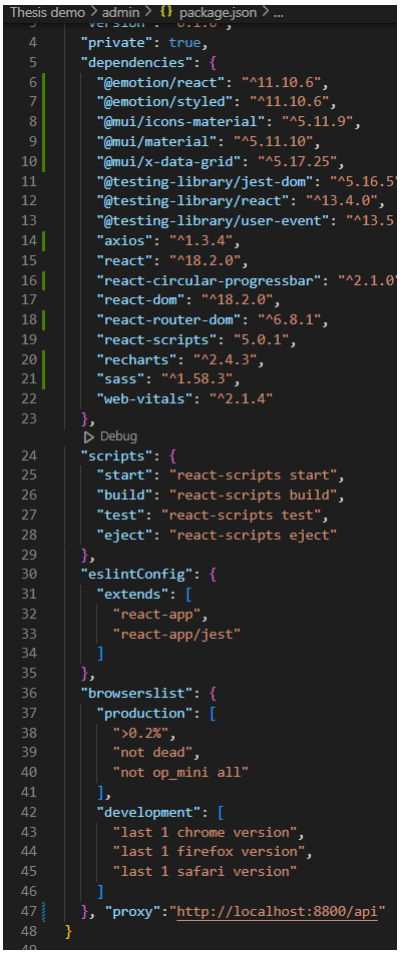


Figure 4.5: Libraries used to build the Front-end of the Admin Dashboard

The libraries used and their functions in the system include:

- React: a JavaScript library where users can build interactive user interfaces (UI). Allows websites to be more eye-catching and user-friendly. The two Hooks "useEffect" and "useState" are most commonly used in react and are also used in this project. For "useEffect", it will allow users to use new values after updating the URL and manipulate those values on the UI. As for useState, it is used to manage components present on a website, it helps to render that component as well as update the interface as required by the user.
- React Router DOM: This is a library used to navigate between single-page applications. It helps to display the correct corresponding UI components when navigated according to the requested URL. useLocation() and useNavigate() are used in this project when the "useLocation" hook is used to get information from the navigation URL to determine whether this website displays the content of which component and updates the UI of that single page, while the "useNavigate" hook is used to navigate to a new page (new URL), this hook is mainly used when users click on page navigation buttons such as navigate(/home) then the system will switch to the Home page after clicking the button.

- React-date-range: this is a library that allows you to get the date, month, and year ranges in the calendar. For this project, there is a function to select a date to be able to book a book, so this library will help to get the number of days that the user selects to see if it matches the information of the book that has been booked or not and also update that number of days for the book that the user chooses to borrow.
- Axios: is a JavaScript library used to easily perform HTTP requests. For this system, Axios is used to perform requests to display login forms and update and retrieve corresponding data with the required URL path to save and display for users.

In addition, a "proxy" command is used to connect to the developed back-end file and linked with Cloud MongoDB allowing both Digital Library and Admin Dashboard to retrieve and add data for the entire system's database.

4.1.1.1.3. Database

The database will be built using MongoDB's cloud because it makes storage easier, can be accessed anywhere if there is account information, and can store more than relying on data on a personal computer. The data will be created according to 3 entities along with the attributes programmed on the VSCode of the personal computer (see Figure 3.12) and directly connected to Cloud through the Mongoose library, when any changes are made, they will be updated immediately on Cloud.

In addition, searching and modifying data on the cloud will also be easier with MongoDB when it is displayed as different data variables along with their attributes.

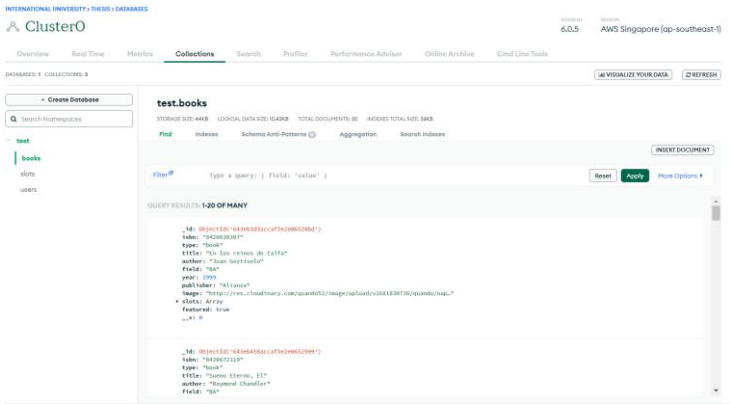


Figure 4.6: Stored data format

4.1.1.2. Code

The code structure is clearly distinguished so that it is easy to imagine the function of each separate folder such as Model, Controller in the API folder of the back-end and the Components and Page folders in the admin and client folders of the front-end.

4.1.1.2.1. Front-end

The front-end code structure will be easy to understand when it includes files such as Components used to build UI for each component that will be displayed on a Main Page. To make it easier to understand, Figure 4.7 will show us the structure of a page that will include many components arranged in a clear order (can see the structure in code at Appendix 2). The UI of each component will be coded separately and the page will only point to those components. What's special about this code structure is that it uses Javascript JSX so there is no need to create separate JavaScript and HTML files to call each other but a JSX file will have both HTML and functions that perform similar functions as when calling into regular JavaScript files.

Figure 4.7: Example of the structure of a page

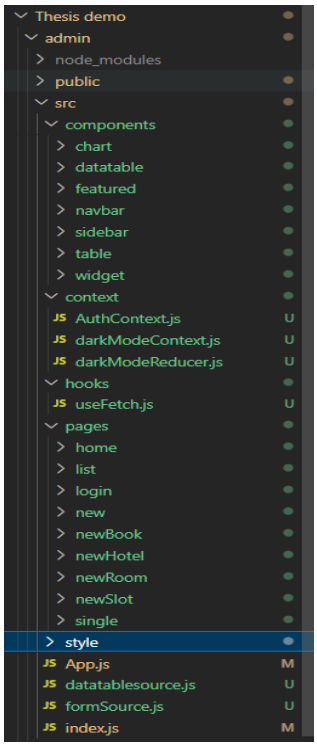


Figure 4.8: Code structure of Admin Dashboard

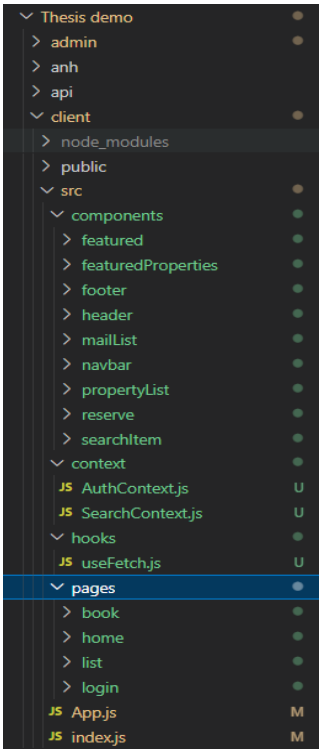


Figure 4.9: Code structure of Digital Library

We have the following folders with separate functions for each file as follows:

- Components: This is the file to contain files that build UI for small components that will appear on different single pages.
- Context: This is the folder used to contain Contexts used to update different functional states such as search and login from users for the system. For the AuthContext file used

to create contexts used to manage user login status in both the Digital Library website and Admin Dashboard. As for the search status, it will be used to determine the search options from users to help the system display correctly when users complete search requests.

- Hooks: This is the file used to send requests to a specific URL in the back-end after receiving search requests, and user choices. For example, when users search for books in the field of Information Technology, hooks will receive that request and send it to a URL containing a list of Information Technology books so that the system can know and return information correctly with what users need.
- Pages: Both Admin Dashboard and Digital Library will have separate single pages that can show different content. These will be files containing the structure of each page including Components combined to create a complete page for the website. For Admin Dashboard, similar files like newBook will be information pages used to create new objects in the database. As for files like List, they will be used to display lists of objects that users search based on information of Digital Library and Book, Home will be used to display the main page of the website, and Login will display the login page for users.

4.1.1.2.2. Back-end

The code structure is divided into separate folders and each folder will contain files to perform functions for different entities such as Book, Slot, User as well as Athuority for login. Below are images of files and folders built for back-end development.

Figure 4.10: Code structure of Back-end

We have the following folders with separate functions for each file as follows:

- Controllers: This is a file built to provide control functions for Book, Slot, and User objects including creating, editing, deleting, getting information, and counting the number of each object. Generally, these are files used to create interaction functions with the database for objects to be able to use URLs to call and perform functions similar to calling HTTP APIs (GET, PUT, DELETE, POST). As for the auth.js file, it is used to develop account creation and login functions for users into the system and database.
- Models: This is a file used to format the attributes of Book, Slot, and User entities.
- Routes: This is the folder used to create different URL paths for each function developed in the Controller file. Each path formatted in this file will call a different function of the 3 objects mentioned above along with Authentication's register and log in.
- Utils: This is a file used to create Token strings used to send back to users as cookies after the system receives and authenticates login information sent from users as JWT strings into the system compared to the database.

4.1.2. Result

To be able to run the entire project, a process needs to be followed to open each file until the entire system including Digital Library and Admin Dashboard is ready for use on a personal computer. The process of opening the project is as follows:

1. Log in to Cloud MongoDB and access the project's Database (as shown in Figure 4.6)
2. Run the backend API file.

3. Run the front-end file of Digital Library and Admin Dashboard.
4. Run the Finding Similar Image model file (as shown in Figure 4.15)

After completing the process of opening files and localhosts of 3 different interfaces as well as connecting to the database on Cloud MongoDB, it is possible to interact with both Digital Library and Admin Dashboard websites as shown below.

4.1.2.1. Digital Library

Login

Logging into Digital Library is not necessary if the user only needs to view information about books, but when they need to borrow books, the user must log in.

The user will only need to enter their Username and Password when provided by the admin. If the user enters the wrong Username, the system will display the message "User not found!"

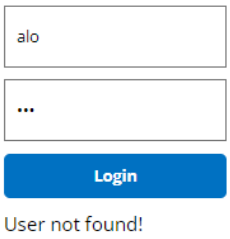


Figure 4.11: Login with the wrong username

If the user enters a Password, the system will display the message "Wrong password or username!".

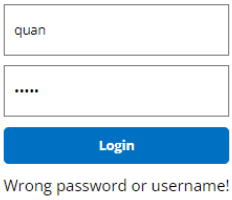


Figure 4.12: Login with wrong password

Homepage

When visiting the Digital Library website, users can see different components on the homepage, with each component displaying different content.

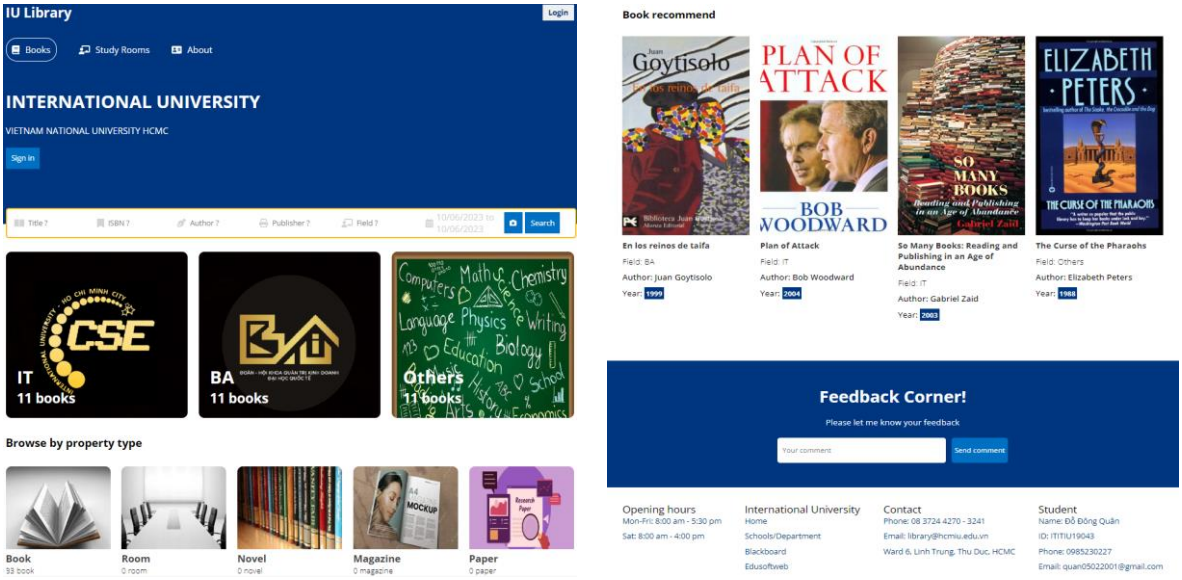


Figure 4.13: Digital Library homepage

At the homepage, users can see buttons labeled "Login" to log in to the system to borrow books. However, logging into the system is not necessary if users only need to view information about the books currently available in the library.

The books in the database are divided into three different fields such as Information Technology, Business Administration, and Others. These books are randomly divided into these three fields only to simulate the search functions and other basic functions of a digital library.

The "Browse by property type" section, will display the types of documents and services that the digital library can provide, including books, rooms (study rooms for students), novels, magazines, and papers. The books added to the database can adjust the attribute "type" to these types of documents and services, then the number will increase and be displayed on the homepage. However, these 4 types of documents and services are only added for simulation purposes but will be further developed in the future.

Next is the "Book Recommend" section, which displays popular books as in other digital library sections. But in this system, a boolean feature when adding a book to the database is used to help display it in this section. If the feature variable has a "true" result, it will be displayed in this list.

Finally, at the bottom of the page are the "Feedback" and "Information of homepage" sections built to help make the homepage look beautiful and complete information referenced from 3 websites in section 2.3.

The most important thing on this homepage is the book information search area as shown in the below figure.

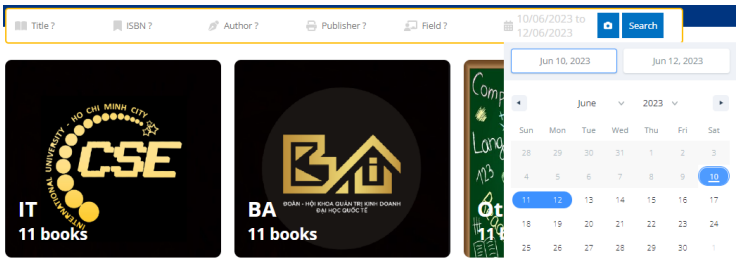


Figure 4.14: Searching area

The user will have the option to search for book information using an image of the book cover by clicking on the button with the "Camera" icon. Then, the system will redirect the user to another localhost URL as shown in the figure below.

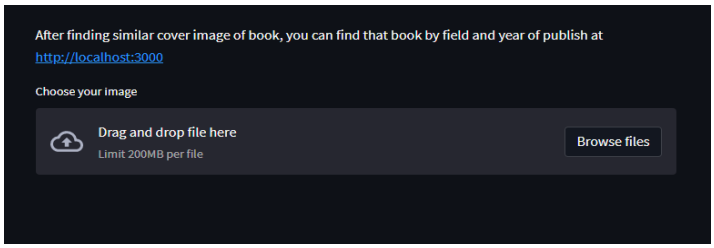


Figure 4.15: Web-based for running Image Searching model

The user clicks on "Browse files" to select an image file with an allowable size on their computer. Then, the system will receive and provide images of books that are most similar to the returned results as follows:

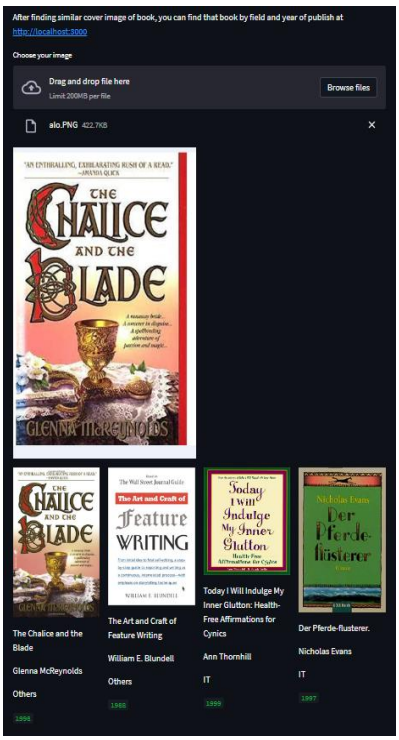


Figure 4.16: Result of image searching

After the system suggests information about books that are similar to the input image, users will rely on information about their desired book to return to search on the homepage with the URL "http://localhost:3000" announced at the top of the page, which satisfied “Feature 5” in Table 4.1.

In the search area in Figure 4.14, users can use the book information suggested on the image search page or search according to their own preferences. Users need to fill any information they want to search for in the digital library such as title, ISBN, author, publisher, and field. Moreover, users do not need to input correctly information since the system can help them find books including the entered information .

In addition, choosing a borrowing date is also very important as it helps the system identify and inform users which books can be borrowed on subsequent pages. For example, in the Figure 4.14, the borrowing date is filled from 10/06 to 12/06 and is highlighted in blue on the displayed calendar. If users do not choose a borrowing date, the system will default to the borrowing and return dates being the same as the date they searched.

After users have filled in search information, click on the Search button to go to a page displaying a list of books belonging to the information that users have chosen. For example, here we will take the book’s information, which users search on the “Finding similar image” page, is the "Others" field.

Then the system will lead users to a web page displaying a list of books belonging to the "Others" field, which satisfied “Feature 1” in Table 4.1.

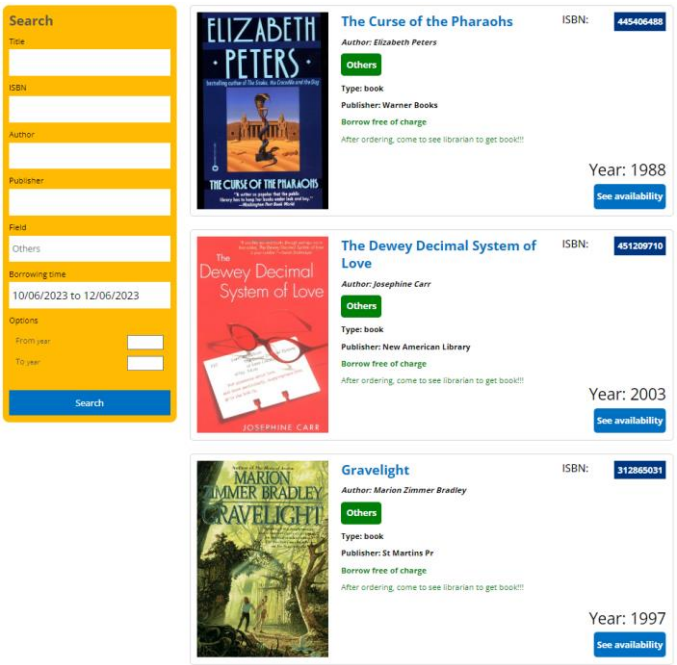


Figure 4.17: Books List page

On this page, the system displays books and their information such as image, title, author, year of publish, isbn, publisher, and field. There is also a special section where users can filter these books by year of publish in the yellow frame.

For example, if the book searched in Figure 4.16 has a year of publish of 1998, it will be filtered as after 1997 or before 1999. With "From year", it will display books published after 1997 and with "To year", it will display books published before 1999.

When filtering by year of publish, the system will display books with a year of publish of 1998 as shown in the figure below.

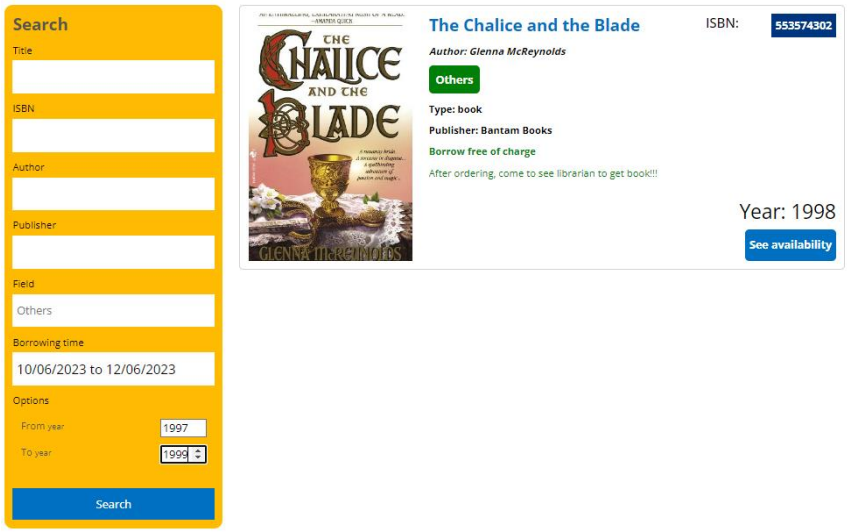


Figure 4.18: Result after filtering the “year”

Besides this search, the user can search based on other relevant information such as publisher’s name, then the system will display all the books have the same publisher as requested by the user as shown in Figure 4.19 (satisfied “Feature 1” in Table 4.1).

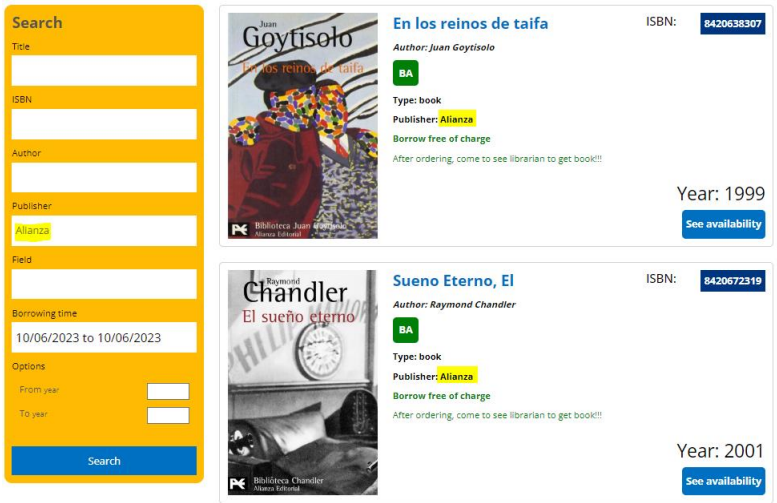


Figure 4.19: Result when searching by “Publisher’s name”

When users have found the book, they want to borrow via clicking on the "See availability" button to view information about the book to borrow and proceed to place an order. The system will redirect users to the book information page.

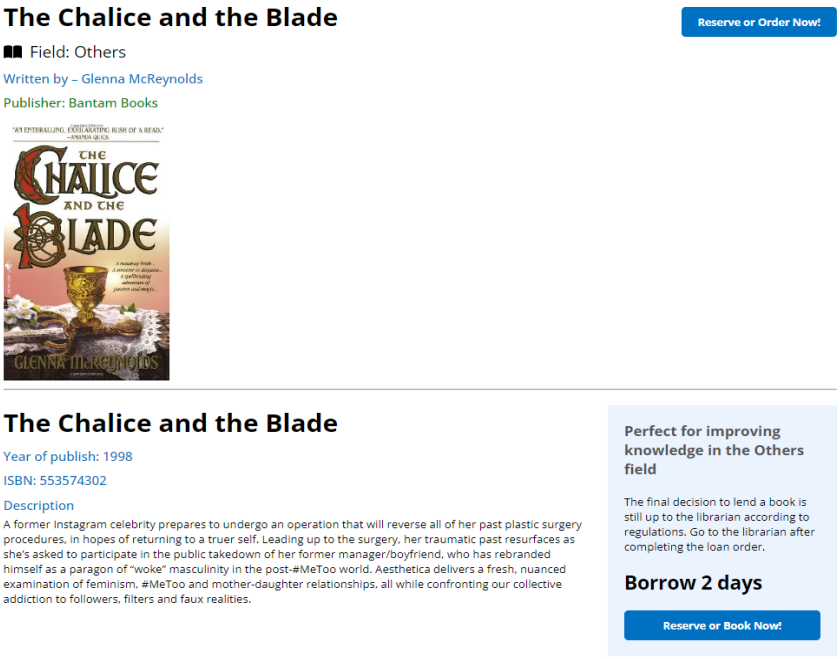


Figure 4.20: Book information page

Here, the system will also display all book information but with additional book reservation buttons including the number of days the book can be borrowed. For example, here it is 2 days from June 10 to June 12 as shown in Figure 4.18 (which satisfied “Feature 2” in Table 4.1)

When users click on the "Reserve or Book Now!" button, the system will redirect them to the Login page if they have not logged in yet and display a book reservation modal slot if they have logged into the system.

If users have logged in, the system will display their Username on the header bar at the top right corner.

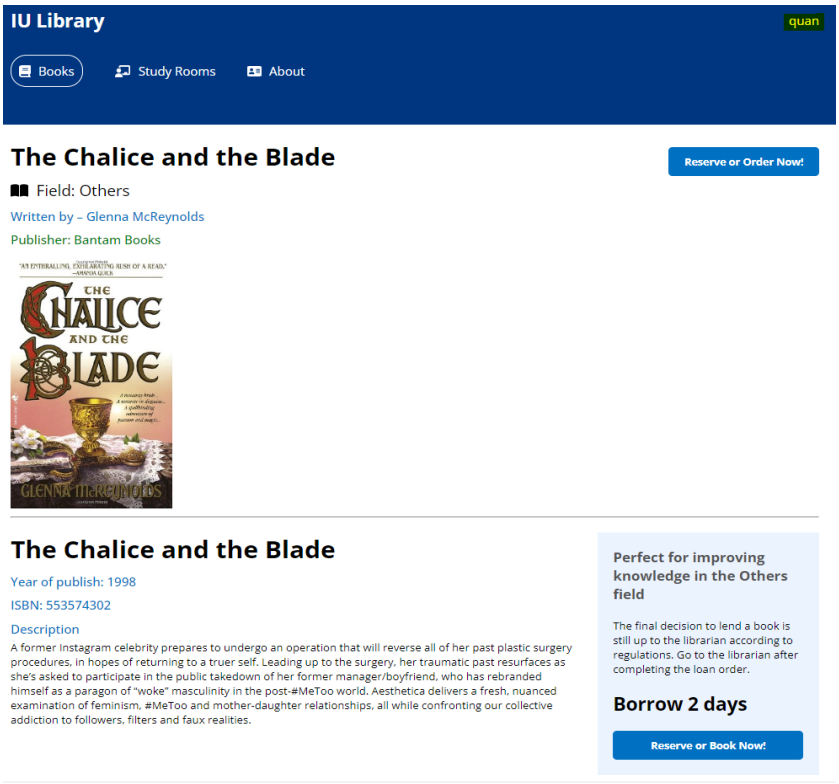


Figure 4.21: Book information page when logged in

However, when logging into the system, the user will return to the homepage and repeat the search steps as mentioned above.

Then the system will display a modal with the number of books available in the library. Users can choose any slot, in the image below, the book "The Chalice and the Blade" has 3 books, and book number 1 is selected and reserved when clicking on the "Reserve Now!" button. Then the system will take users back to the homepage.

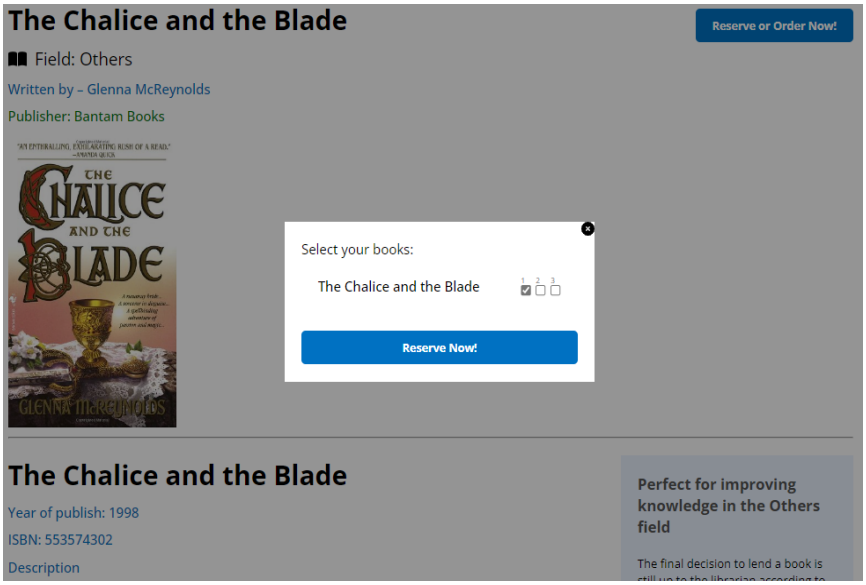


Figure 4.22: Selecting slot of the book

After the book in 1st slot has been reserved from 10/06 to 12/06, it will not be possible to select that book for these two borrowing days anymore and the user must choose 2nd and 3rd Slot to borrow, which satisfied “Feature 3” in Table 4.1.

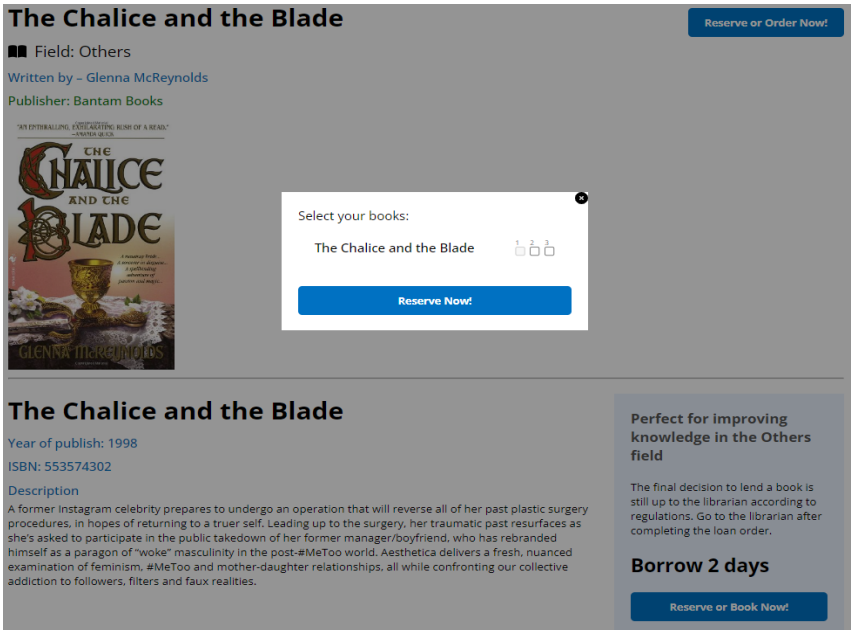


Figure 4.23: Slot status when booked

But if the user changes the borrowing date from 13/06 to 16/06 as in Figure 4.24 then the book in Slot 1 can be reserved because it does not overlap with the previously reserved date Figure 4.18.

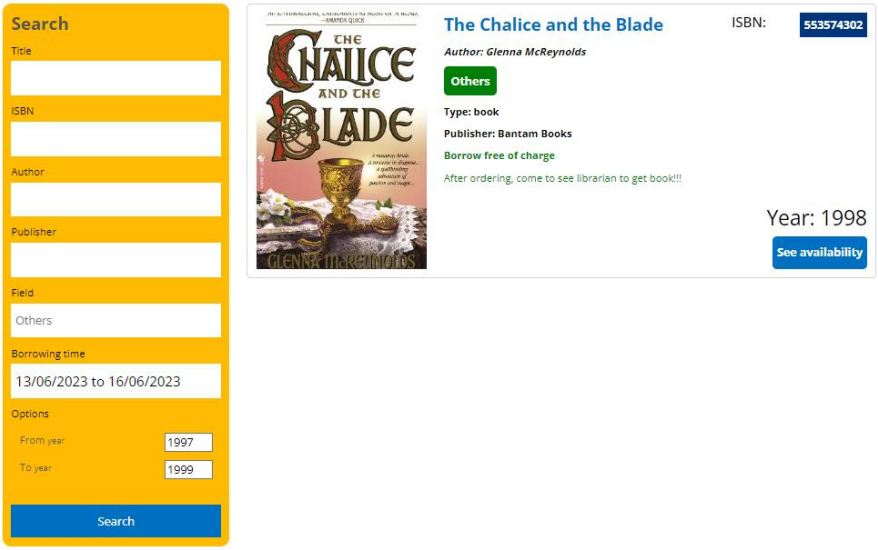


Figure 4.24: Changing the borrow date

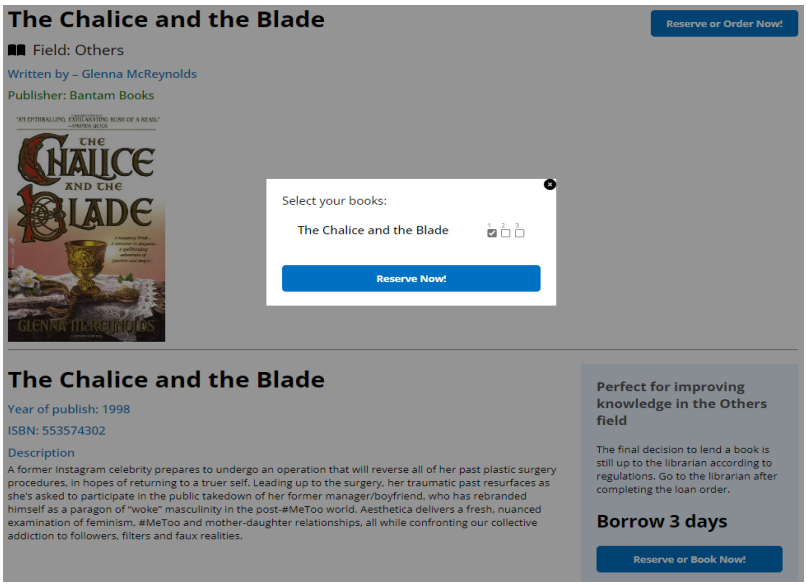


Figure 4.25: Slot status after changing the borrow date

And that is all the interface and functions that this project's Digital Library can perform.

4.1.2.1. Admin Dashboard

With the Admin Dashboard website, only users with the "isAdmin" attribute set to true in the database can log in here and manage library data. If unauthorized accounts try to log in, the system will recognize it and notify the user that they do not have access, which satisfied “Feature 4” in Table 4.1.

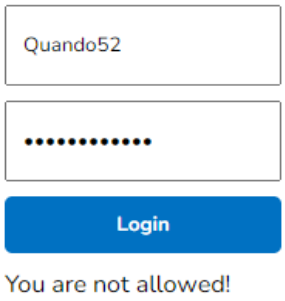
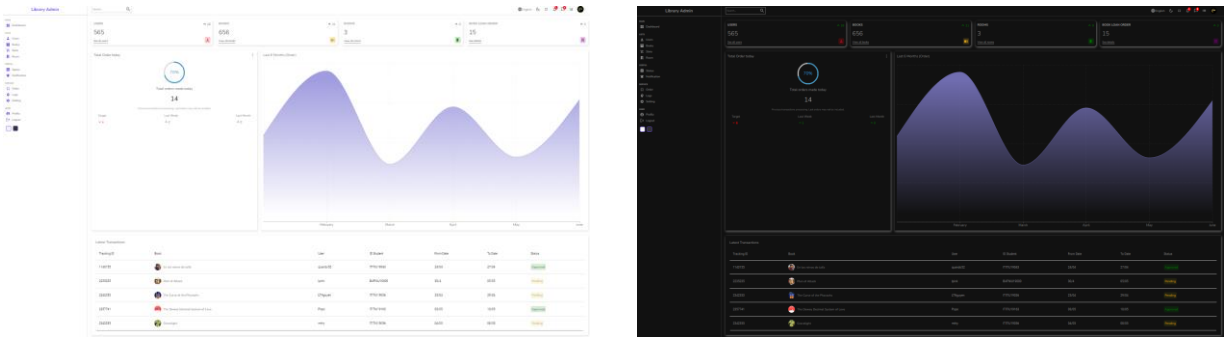


Figure 4.26: Login with unauthorized account

After logging in with an authorized account, users will go to the Dashboard homepage where information about digital library data such as the number of books, number of accounts, book borrowing requests, and their status will be displayed. However, what is designed for UI on this homepage is only developed to make it look better but cannot be interacted with.

Only Books, Users, and Slots in the Sidebar can be interacted with and it will lead users to other pages to interact with system data. In addition, the Admin website can also use white and black backgrounds as shown in Figure below.



a) Light mode b) Dark mode

Figure 4.27: Admin Dashboard homepage with

a) Light mode b) Dark mode

Users

When clicking on the "Users" button on the sidebar, the system will redirect to a page displaying a list of all user accounts in the database. Here, the admin has two options to interact with the data: delete data when clicking on the "Delete" button and create a new user account when clicking on the "Add New" button. (which satisfied “Feature 6” in Table 4.1)

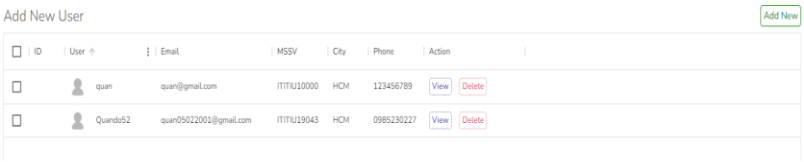


Figure 4.28: User account list

When clicking on "Add New", the admin will be redirected to a new page with a form used to create a user account.

Here, the admin must fill in all information of the account (image is not required), however, Username, MSSV, and email are information that does not exist in the previous database. If the admin enters information that matches existing accounts in the database, then the account will not be created.

After filling in all information, click on the "Send" button for the system to record creating a new user account in the database.

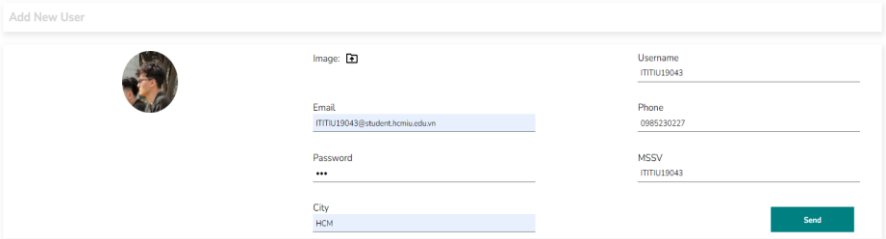


Figure 4.29: “Create new account” page

When returning to the page displaying the user list, the newly created account will be displayed.

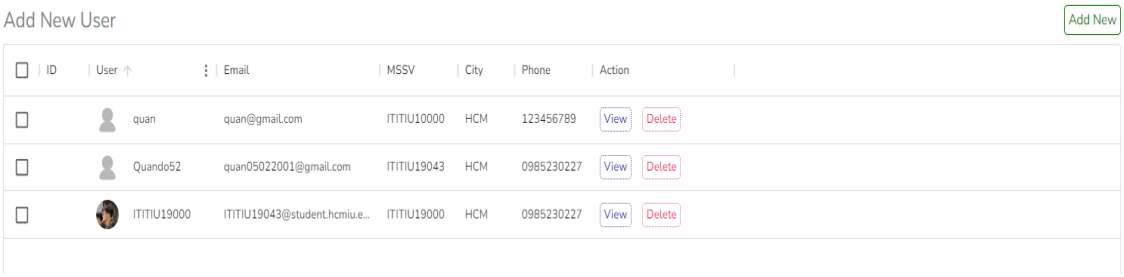


Figure 4.30: User account list after creating a new account

Books

When clicking on the "Books" button on the sidebar, the system will redirect to a page displaying a list of all books in the database. Here, the admin has two options: delete a book by clicking on the "Delete" button and add a new book to the database by clicking on the "Add New" button, which satisfied “Feature 6” in Table 4.1.

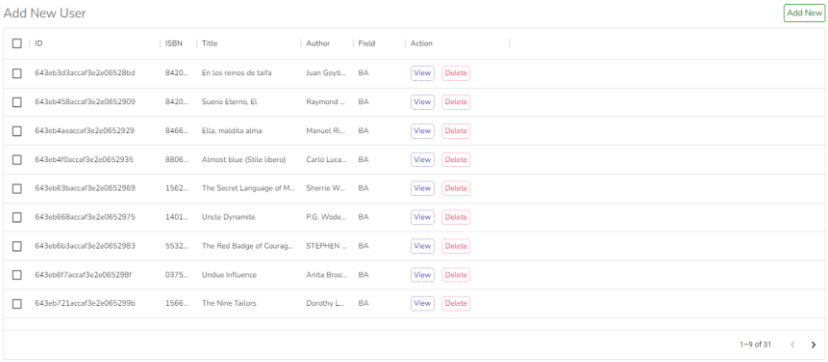


Figure 4.31: Books list

When clicking on "Add New", admin will be redirected to a new page with a form used to add a new book to the library.

Here, the admin must fill in all information about the book such as ISBN, title, author, year of publication, and publisher. In addition, the admin needs to fill in information such as data type (Book, Magazine, Novel) to help the system display the number of each type in the "Browse by property type" section in Figure 4.13. Moreover, the Field section also needs to be filled correctly as IT, BA, or Others so that the system can classify them at Figure 4.13 and easily display them when users search.

There is one special thing that needs to be added which is Slots quantity at the bottom. Admin needs to tick on the slot that has the same name with the book (already created in Slots page) so that the system can display how many copies of that book are actually available in the school library.

After filling in all information, click on the "Send" button for the system to record creating a new book in the database.

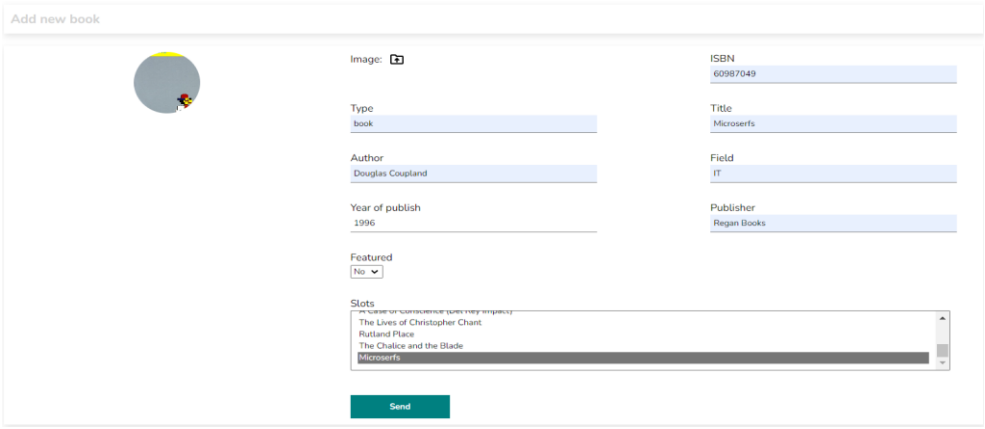


Figure 4.32: “Add new book” page

When returning to the Books List page, the newly created book will be displayed at the end of the list.

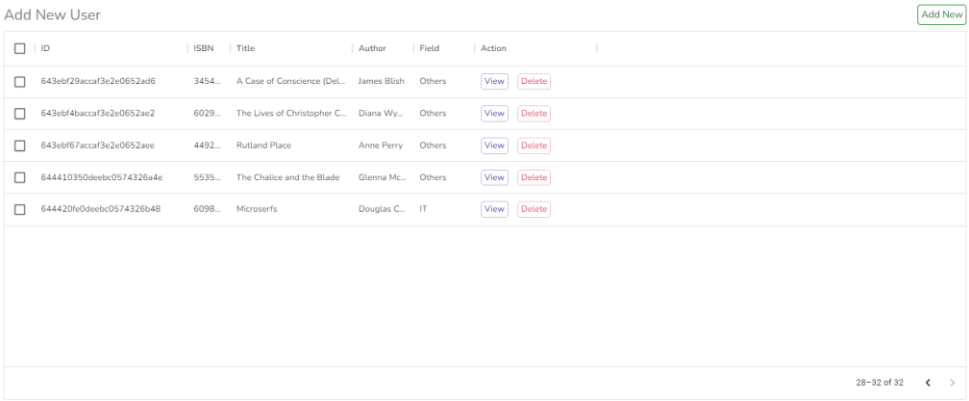


Figure 4.33: Books list after adding new book

Slots

When clicking on the "Slots" button on the sidebar, the system will redirect to a page displaying a list of all slotsin the database. Here, the admin has two options: delete a slot by clicking on the "Delete" button and add a new slot to the database by clicking on the "Add New" button, which satisfied “Feature 6” in Table 4.1.

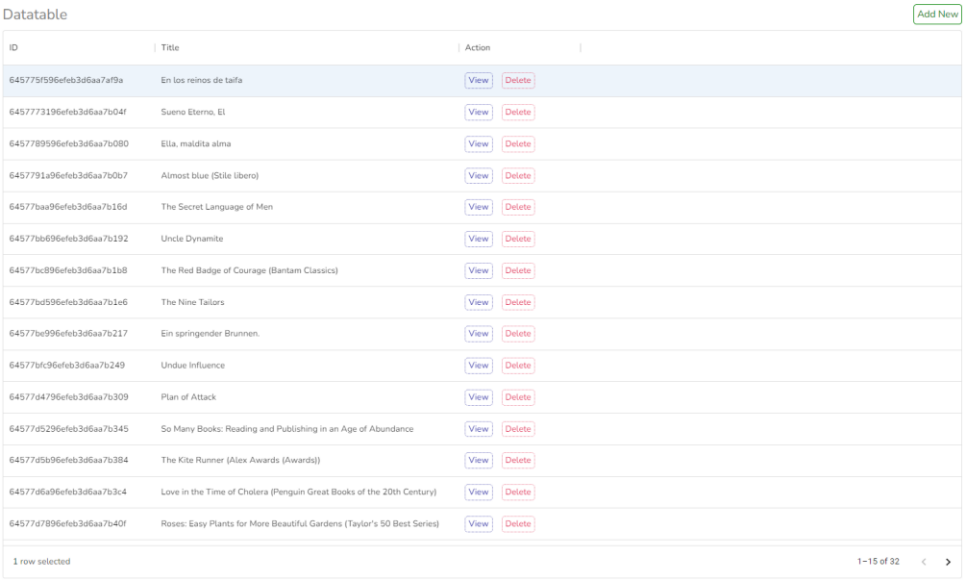


Figure 4.34: Slots list

When clicking on "Add New", admin will be redirected to a new page with a form used to add a new slot for an existed book in the database.

Here, the admin must fill in all information about the book such as title for slots of the book, and the number of slots (with a comma between each slot number). In the Figure 4.35, we can see the “Slots” information is filled with “1, 2” and Microserfs book was selected in the “Choose a book” section.

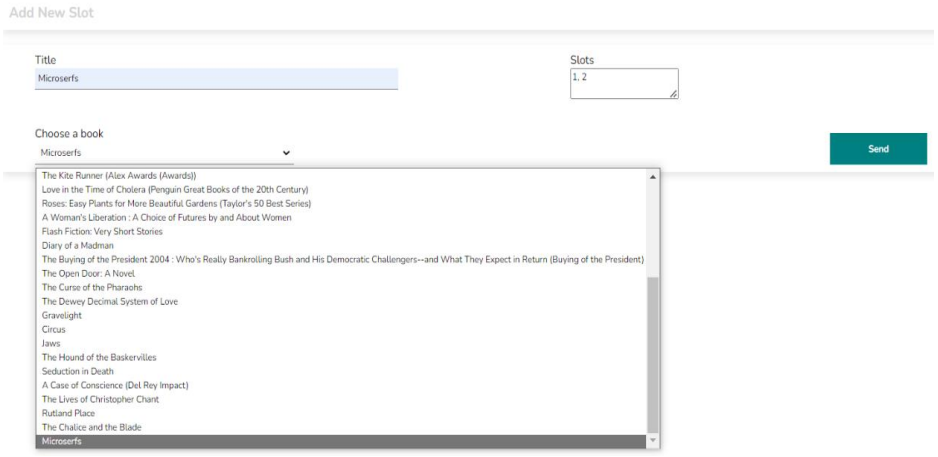


Figure 4.35: “Add new slot” page

After filling in all information, click on the "Send" button for the system to record creating a new slot for the chosen book in the database. And we can see all slots of “Microserfs” book is shown in the Figure 4.36.

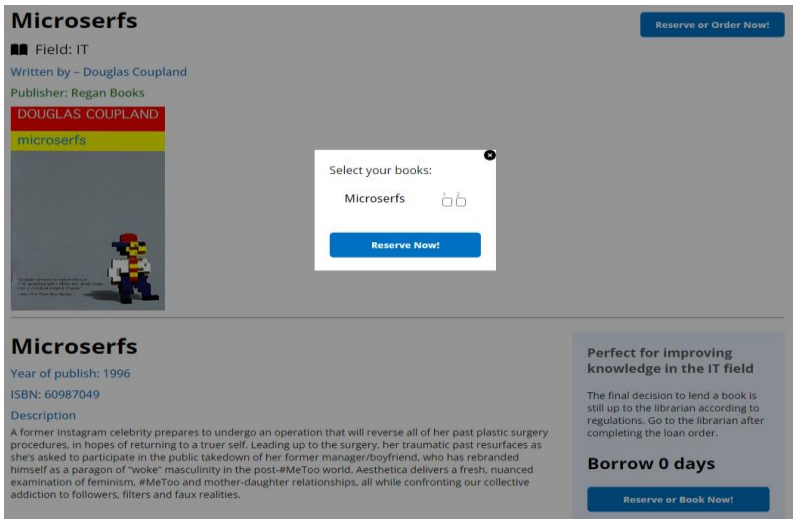


Figure 4.36: Slots after creating

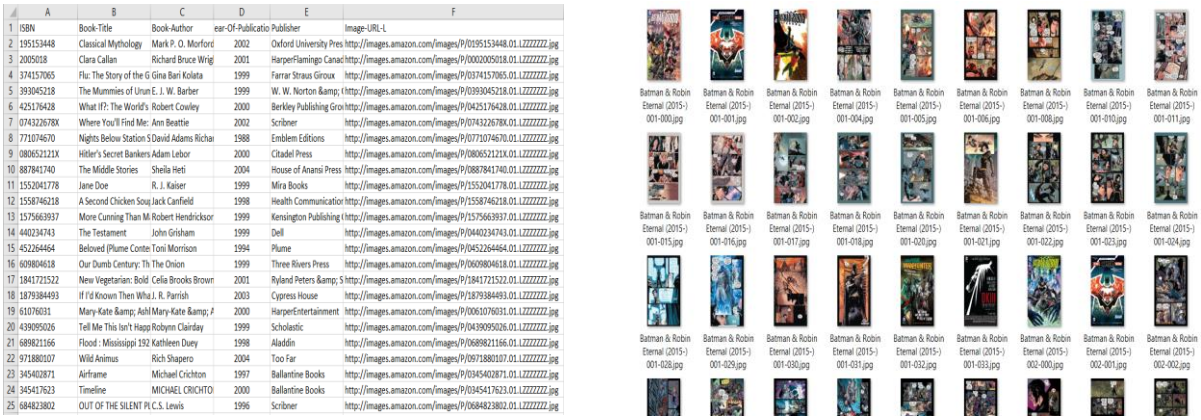
4.2. Machine learning model

The entire process of model training will be developed, built, processed, and tested on a local machine, which is referenced in this GitHub [20]. Since the amount of data used for the complete model training is a relatively large image file and it would take a lot of time if processed on local IDEs, Google Collab is a more optimal choice. After a model is trained, it is mandatory to test its effectiveness with tagged data. Next, the data processing steps will be performed to suit the trained model, as well as make it easy to apply data to a web-based digital library, and to improve the operational performance when applying the model to the web. The image data processing steps, along with the accompanying information and KNN combination, will all be processed on the PyCharm IDE on the local machine for better management. Finally, to test the demo functionality of the model for a web-based application, Streamlit will be developed and completed for deployment on the web local host.

4.2.1 Implementation

4.2.1.1. Dataset

As mentioned in Section 3.4.2, the data used to train the model will be sourced from two different datasets on Kaggle.com, namely the "Book Recommendation Dataset" and the "Comic Book Classification" dataset. The former contains book information along with the corresponding cover image link, while the latter consists of comic book images collected from various comic book series. Both datasets will be divided into different train, validation, and test folders to train the model and evaluate its performance.



	A	B	C	D	E	F
1	ISBN	Book-Title	Book-Author	ear-Of-Publicatio	Publisher	Image-URL-L
2	195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Pres	http://images.amazon.com/images/P/0195153448.01.LZZZZZZZ.jpg
3	2005018	Clara Callan	Richard Bruce Wrig	2001	HarperFlamingo Canad	http://images.amazon.com/images/P/0002005018.01.LZZZZZZZ.jpg
4	374157065	Flu: The Story of the G	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.amazon.com/images/P/0374157065.01.LZZZZZZZ.jpg
5	393045218	The Mummies of Urun	E. J. W. Barber	1999	W. W. Norton & (	http://images.amazon.com/images/P/0393045218.01.LZZZZZZZ.jpg
6	425176428	What If?: The World's	Robert Cowley	2000	Berkley Publishing Gro	http://images.amazon.com/images/P/0425176428.01.LZZZZZZZ.jpg
7	07432678X	Where You'll Find Me:	Ann Beattie	2002	Scribner	http://images.amazon.com/images/P/07432678X.01.LZZZZZZZ.jpg
8	771074670	Nights Below Station S	David Adams Richar	1988	Emblem Editions	http://images.amazon.com/images/P/0771074670.01.LZZZZZZZ.jpg
9	080652121X	Hitler's Secret Bankers	Adam Lebor	2000	Citadel Press	http://images.amazon.com/images/P/080652121X.01.LZZZZZZZ.jpg
10	887841740	The Middle Stories	Sheila Heti	2004	House of Anansi Press	http://images.amazon.com/images/P/0887841740.01.LZZZZZZZ.jpg
11	1552041778	Jane Doe	R. J. Kaiser	1999	Mira Books	http://images.amazon.com/images/P/1552041778.01.LZZZZZZZ.jpg
12	1558746218	A Second Chicken Soup	Jack Canfield	1998	Health Communication	http://images.amazon.com/images/P/1558746218.01.LZZZZZZZ.jpg
13	1575663937	More Cunning Than Me	Robert Hendrickson	1999	Kensington Publishing (	http://images.amazon.com/images/P/1575663937.01.LZZZZZZZ.jpg
14	440234743	The Testament	John Grisham	1999	Dell	http://images.amazon.com/images/P/0440234743.01.LZZZZZZZ.jpg
15	452264464	Beloved (Plume Conte	Toni Morrison	1994	Plume	http://images.amazon.com/images/P/0452264464.01.LZZZZZZZ.jpg
16	609804618	Our Dumb Century: Th	The Onion	1999	Three Rivers Press	http://images.amazon.com/images/P/0609804618.01.LZZZZZZZ.jpg
17	1841721522	New Vegetarian: Bold	Celia Brooks Brown	2001	Ryland Peters & S	http://images.amazon.com/images/P/1841721522.01.LZZZZZZZ.jpg
18	1879384493	If I'd Known Then Wha	J. R. Parrish	2003	Cypress House	http://images.amazon.com/images/P/1879384493.01.LZZZZZZZ.jpg
19	61076031	Mary-Kate & Ashl	Mary-Kate & A	2000	HarperEntertainment	http://images.amazon.com/images/P/0061076031.01.LZZZZZZZ.jpg
20	439095026	Tell Me This Isn't Happ	Robynn Clairday	1999	Scholastic	http://images.amazon.com/images/P/0439095026.01.LZZZZZZZ.jpg
21	689821166	Flood : Mississippi 192	Kathleen Duey	1998	Aladdin	http://images.amazon.com/images/P/0689821166.01.LZZZZZZZ.jpg
22	971880107	Wild Animus	Rich Shapero	2004	Too Far	http://images.amazon.com/images/P/0971880107.01.LZZZZZZZ.jpg
23	345402871	Airframe	Michael Crichton	1997	Ballantine Books	http://images.amazon.com/images/P/0345402871.01.LZZZZZZZ.jpg
24	345417623	Timeline	MICHAEL CRICHTO	2000	Ballantine Books	http://images.amazon.com/images/P/0345417623.01.LZZZZZZZ.jpg
25	684823802	OUT OF THE SILENT PL	C.S. Lewis	1996	Scribner	http://images.amazon.com/images/P/0684823802.01.LZZZZZZZ.jpg

(a)

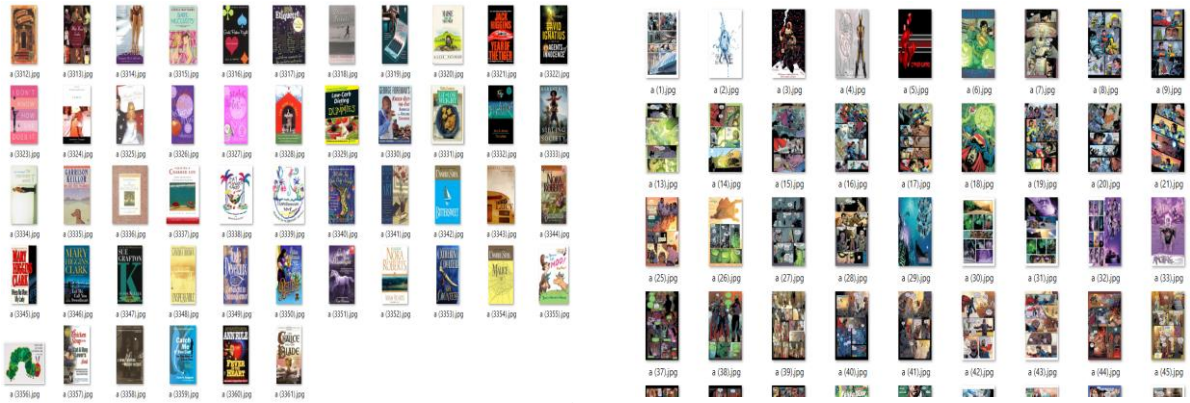
(b)

Figure 4.37: Initial dataset of (a) “Book Recommendation Dataset” (b) “Comic Book Classification”

Figure 4.37 shows the initial datasets for both files downloaded from Kaggle, with the "Book Recommendation Dataset" containing information about a book, including ISBN, book title, author (only the first author of the book is taken), year of publication, publisher, and URLs linking to the book cover image provided by Amazon, where the book labels will be used to test the model later as well as for use in the web-based digital library, while the images will be used to train the model.

This is a very large amount of data with up to 271,360 different books and their information. However, during the data cleaning process, I discovered that many books had unavailable images, and since this research only focuses on images, I selected and extracted just over 3,000 cover images of 3,000 different books to conduct this study. Converting the URL links to complete image files would be quite complex if done step by step, but I used the KuTools for Excel app to convert URLs to images and export a separate .zip file of images after completion, resulting in a dataset as shown in the Figure 4.38a.

For the "Comic Book Classification" data file, the data processing will be easier when just randomly taking comic images from different series to divide into the train, valid, and test folders and the files will look like the figure below



(a) (b)

Figure 4.38: Image dataset after processing of (a) “Book Recommendation Dataset” (b) “Comic Book Classification”

4.2.1.2. Training Model

After preparing the completed data files for model training, a VGG16 algorithm was chosen, and the model architecture was redesigned to be more suitable for the purpose and data files of this problem. The architecture includes layers, activation functions, loss functions, and optimization algorithms that were discussed and analyzed in section 3.4.1. Finally, the model structure used to train the model will be displayed as shown in the Table below (model structure in coding can be seen in Appendix 16).

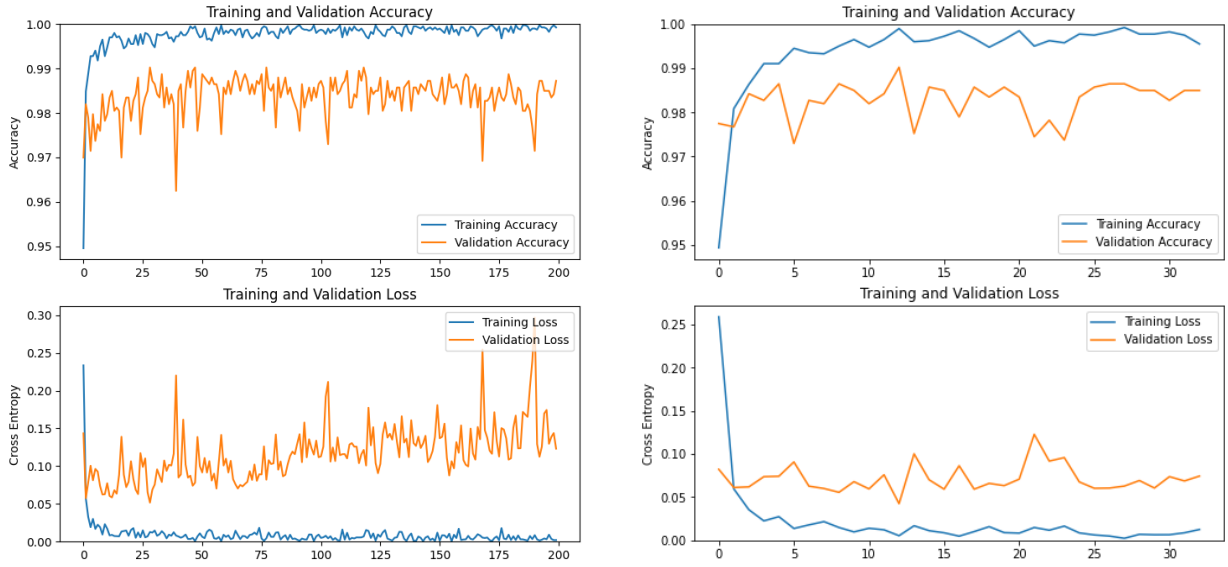
Layer	Type	Output Shape	Parameters
vgg16	Functional	(None, 7, 7, 512)	14714688
max_pooling2d_2	MaxPooling 2D	(None, 3, 3, 512)	0
flatten_2	Flatten	(None, 4608)	0
dense_4	Dense	(None, 2048)	94392332
batch_normalization_2	BatchNormalization	(None, 2048)	8192
dropout_2	Dropout	(None, 2048)	0
dense_5	Dense	(None, 2)	4098
Total params	24,166,210		
Trainable params	9,447,426		
Non-trainable params	14,718,784		

Table 4.2: The training model structure

Proposal model has a total of 24,166,210 parameters, of which 9,447,426 parameters are trained and 14,718,784 parameters are not trained. This means that there are some parameters that are fixed and cannot be updated or trained further as parameters of the pre-trained VGG16 model.

After determining and building the required model architecture, the model is compiled using the Adam optimizer with a learning rate of 0.001 and categorical cross-entropy as the loss function. Two callbacks are used during training: ModelCheckpoint and EarlyStopping. ModelCheckpoint saves the model with the highest validation accuracy during training, while EarlyStopping stops the training if the validation accuracy does not improve after 20 epochs. Although the model is trained for 200 epochs, the early stopping function was called at epoch 33 as the accuracy did not improve further (see Appendix 1), which helps to reduce the time for training the model without the need to run all 200 epochs.

Finally, the training history is saved in variables for future analysis, and the trained model is saved in a "model.h5" file for later use in testing and application to the web-based digital library. In addition, the accuracy and validation loss metrics of the trained model are examined to assess its performance, as shown in the following figure.



a) Model without Early Stopping b) Model with Early Stopping

Figure 4.39: The accuracy and loss of model

(a) without Early Stopping (b) with EarlyStopping

4.2.1.3. Data Processing

After completing the model training, the next step is to convert and test the model for real-world applications. Firstly, as the web-based digital library uses MongoDB, a portion of the data file containing book information from the "Book Recommendation Dataset" will be

converted to a convenient JSON format for accessing the demo model and for easy application to the developed localhost website. The information will be converted to JSON format as follows:

Figure 4.40: JSON format of "Book Recommendation Dataset"

Next, after obtaining a JSON file containing labels for the book images in the test file, a dataset will be created with feature vectors computed from a pre-trained model. The steps for this process are as follows:

1. Declare parameters for image processing, including width, height, number of channels, image size, and training speed (FAST_RUN).
2. Define a get_embedding function to compute the feature vector for any image.
3. Create a function for computing feature vectors for images in the test folder.
4. Load the pre-trained model from the model.h5 file.
5. Retrieve the last layer before softmax to use as the feature vector.
6. Use this model to compute the feature vector for each image in the dataset.
7. Initialize a new JSON file to write the feature vectors and other image information to the file.

Finally, a JSON file will be created to store the information of the image as well as the initialized feature vector as shown in the image below:

Figure 4.41: JSON file store image vector and information

Next, the kNN algorithm will be used to calculate the similarity between images. The following steps will be performed:

1. Define a function that loads an image, pre-processes it, and passes it through a pre-trained model to obtain the embedding.
2. Load the embeddings and other information from a JSON file (Figure 4.41) and convert it to a Pandas data frame.
3. Load a pre-trained deep learning model (model.h5) and extract the output of its third layer as the embeddings.
4. For each book type (in this case, there is only one type of book), select the embeddings and other information corresponding to that type from the Pandas data frame, convert the embeddings to a matrix, apply a KNN algorithm with n_neighbors=5 to the matrix, and save the kNN model to a file (knn.pkl) for later use.

Finally, the information for each book in the same JSON file of Figure 4.40 will be separated into individual JSON files for each book. In addition, the image file of each book will also be saved to a folder that includes the previously saved KNN file (Figure 4.42) to improve performance when applied to the website because it will be easier to access.

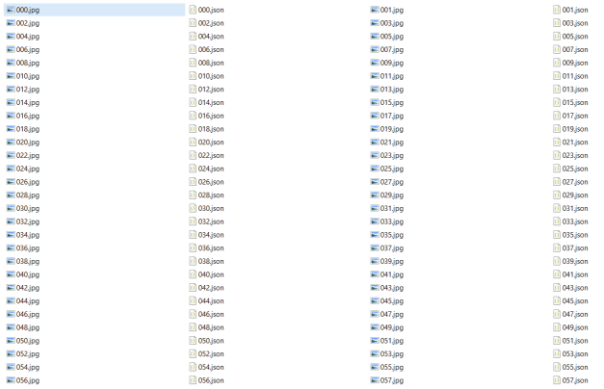


Figure 4.42: The folder after splitting into individual JSON files

4.2.2. Result

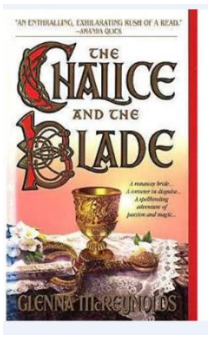
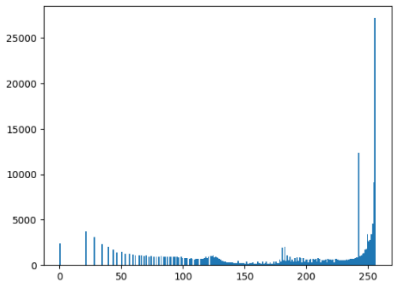
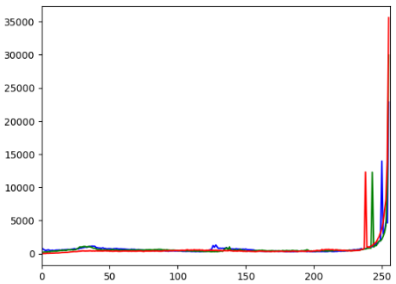
Next is to develop a simple Streamlit application that allows users to upload an image, extract features from the image using the pre-trained VGG16 model, and then use KNN to find the closest matching book cover images from the available book cover dataset.

After finding the closest matching book covers, the application will display the top 4 closest matching images along with some information about these books including the book title, author, field, and year. This process takes the input image and turns it into a corresponding feature vector by using the previously trained model.h5 with the image and projecting it onto the knn.pkl file. Finally, the closest vectors are obtained and the JSON file containing the book information of those vectors is read and returned to the system.

As mentioned in Section 2.2.1, Content-based image retrieval is a system used to retrieve images based on image content such as color, shape, structure, and other features. Because the proposal model was used to search for close images in the prepared datatset file and gave test results as shown in Figure 4.16. The method “Histogram Calculation” is used as a method to evaluate the similarity of results when applying proposal modal, resulting in histograms representing the distribution of color values in the image.

Through studying in OpenCV's article on histogram [29], there will be 2 histograms developed from the input image and the resulting images of the model in Figure 4.16.

- The first histogram is analyzed when converting images to grayscale to see the number of pixels with corresponding gray values, which provides important information about brightness, contrast, or the uniform gray distribution of the image.
- The second Histogram is used for color images to display information about the distribution of colors in each color channel (Red, Green, Blue). The histogram will show three lines representing the different color channels and the corresponding number of pixels with values from 0 to 255 for each color value.

	Image	**Histogram (Gray)**	**Histogram (R,G,B)**
Query			

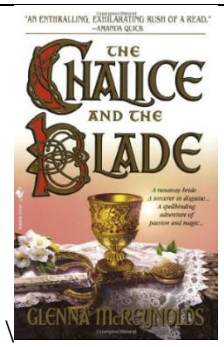
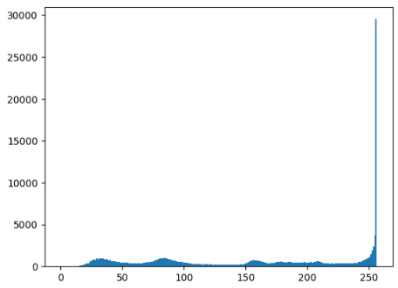
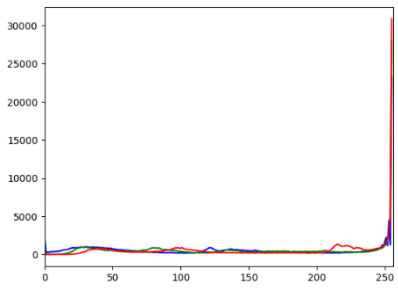
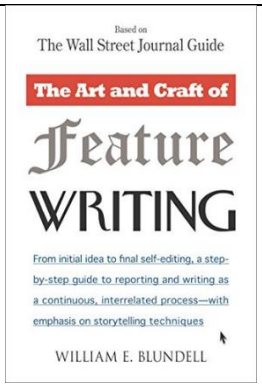
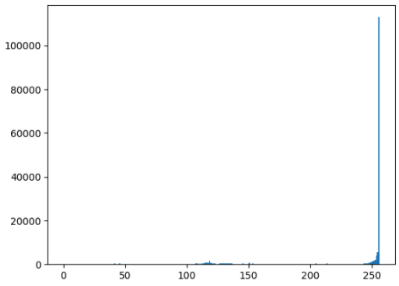
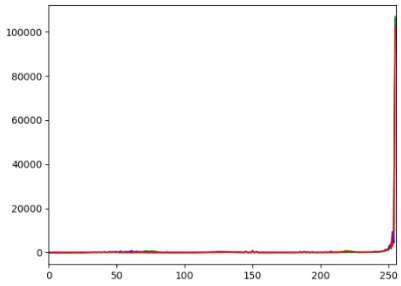
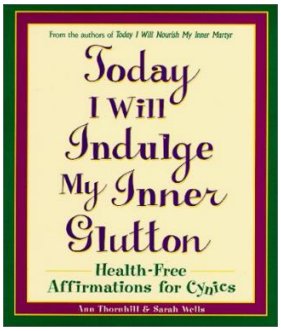
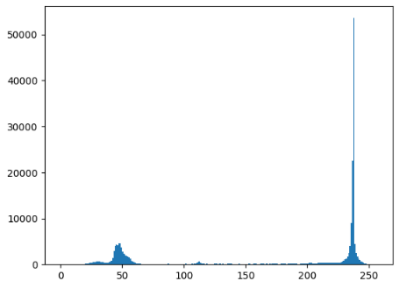
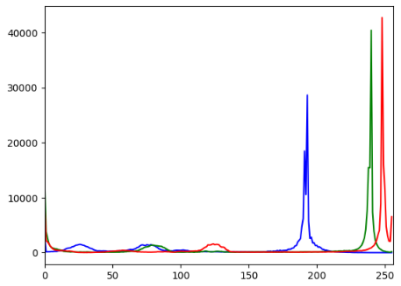
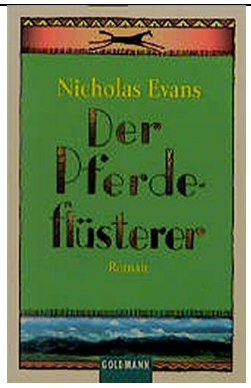
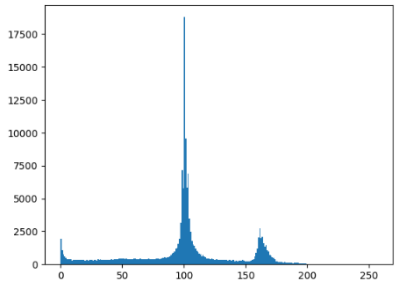
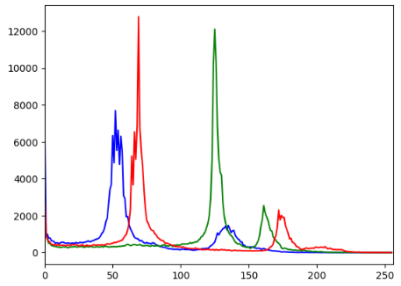
Result 1			
Result 2			
Result 3			
Result 4			

Table 4.3: Histogram Calculation

Based on the results in Table 4.3, the input image is analyzed as a gray image whose gray level distribution in the image is mostly around 255-pixel level, and from Result 1, 2, 3 also has gray level in the image which mostly distributed at 255. For histograms analyzed for color images, the degree of color distribution is similar to that compared with histograms for gray images.

Chapter 5 : DISCUSSION AND EVALUATION

5.1. Discussion

Some of the initial job objectives have been met with a complete demo version such as a model used to search for book cover images that are similarly developed from VGG16, a digital library that provides full information about books, methods of searching, and borrowing books by day, a system for librarians to control and adjust data in the system. In addition, the interaction between the digital library and the image search model to provide necessary information for users is also an advantage of the deployed system.

The model used and developed for the image search function is VGG16 of Neural Network which is not yet the most optimal method but has a very good performance and is useful for what digital library needs. In addition, the digital library developed still has many things that need to be improved such as creating a complete order form and managing it. The system is still at the level of implementing the basic functions of a library and checking whether the image search method is suitable for it or not, so the content will be further developed when applying new technologies in both web programming and machine learning in the future.

5.2. Comparison

5.2.1. Finding Similar Image Model

Different from searching for information based on text, product information or a book, searching based on book cover images has not been widely used but will be a helpful method for increasing and diversing user experience. Compared with other digital library applications, the advantage of applying the "finding similar image" function in this thesis is that it allows users to search for a book without the relevant information of the book such as title, author, field, publisher, etc. Instead, users only need to have an image of the book cover they want to find, which increases the user experience, saves time searching and interacting with the system. The disadvantage of this built-in model and function is that there is no automatic cropping feature for the input image, identifying the book in an image has a lot of redundant information, which forces users to use a input image with only the book cover image due to limited data and model.

5.2.2. Digital Library Website

As you mentioned in Chapter 2, Open Library and IU Library are good examples to look at and develop the system in this project. In addition, Shopee e-commerce is one of the few websites that apply an image search function. So, let's take a look at the developed system and how those 3 websites differ.

With Open Library, besides providing users with the ability to view book information, it has a feature that no library has which is allowing users to read books online. However, this book can only be viewed online by the hour and there is no physical library that provides book borrowing services by day.

Regarding IU Library, this is a website used to manage and help users interact more easily with physical libraries when providing information about books, displaying book reservation information, and allowing book borrowing, similar functions to the system developed from this project.

Unlike the above 2 websites, Shopee is used as an example for an image search function. In addition to using images to search for the most similar content, this system also applies the function of cutting out the main content in an image to increase search accuracy. In addition, a large database of many images of a product helps it have better output results than the relatively small data of this project's system.

These two tables below describe the comparison.

Digital Library			
	Open Library	**IU Library**	**My system**
Searching by information	Yes	Yes	Yes (All books can be found by information of the book)
Read Book Online	Yes	No	No (The dataset is only covering image and general information, not all pages of the book)
Show book available or not	Yes	Yes	Yes (Slot of the book can possible to update the number of days and make it unavailable on these dates)
Support multiple language	Yes	No	No (Will be updated in the future)
Data Tracker	Unknow	Yes	Yes (There is Admin Dashboard systems for managing the data in the database)

Table 5.1: Digital Library's Function Comparison

Finding Similar Image		
	Shopee	**My application**
Crop content in image	Yes	No (This must apply another model like Yolo and there is no dataset for training this model)

Suggested information of similar products	Yes	Yes (Provide relevant information of suggested photos)

Table 5.2: Finding Similar Image Function Comparison

5.3. Evaluation

In this project, proposing books by finding similar images has been created based on an image dataset of book covers along with their information. The method used for this search function is transfer learning from the VGG16 model developed from the Convolutional Neural Network, which has been successfully applied with a fairly good performance and is suitable for a digital library website. Suggesting book types for users so that they can search, filter, and borrow books from a digital library system will be more helpful for both physical libraries and users. In addition, there is also a Dashboard used to manage assets and data such as books and user accounts that have been built to help librarians more.

However, building a machine learning model is still only from a small dataset so it needs to research more optimal methods as well as expand the dataset to be more diverse to aim for the best performance for websites. Building more functions to manage other types of resources of the IU library will also be an essential task that needs to be done for upgrading the system after this project.

Chapter 6 : CONCLUSION AND FUTURE WORK

6.1. Conclusion

The digital library system has a function of searching for similar images developed from a neural network that has most of the basic functions that a digital library website needs such as: suggesting information through image search, searching for book information, borrowing books, viewing information on borrowed dates, managing user accounts, and the number of books. This project is an opportunity to better understand the platform for building a machine-learning model, especially in the field of Image Classification. In addition, the digital library system enhances knowledge about programming a website and how to implement a digital library system in practice.

6.2. Future work

Although the newly developed system is suitable for a basic digital library system, there are still many shortcomings that need to be improved.

The first is about upgrading the machine learning model with a newer, more optimized method, analyzing the image into more layers to remove the redundant content in the image and focus on the information needed to find image.

The second thing is a very nice function of Shopee, which is to cut the main content from the input image. Not every user input image will be a book cover, so this results in less accuracy when searching. The application of a Yolo model for object recognition is necessary and helps users much more.

What follows is a larger data file for both similarity and object recognition models. This will make the system performance faster, give more accurate results, and recommend the right books that the user needs. In addition, the book data file for training the Yolo model is not yet available, so it is necessary to crawl from many sources and carefully prepare the bounding boxes.

Next are the things to do for the digital library system. First, it is necessary to build additional functions to lend books from users, such as notifying users and saving to the Admin Dashboard for easier control. In addition, editing data also needs to be added to the Dashboard instead of directly manipulating the Cloud Database, easily causing errors for librarians.

REFERENCES

1. Mehta, D., & Wang, X. (2020). COVID-19 and digital library services–a case study of a university library. Digital library perspectives, 36(4), 351-363.
2. Chowdhury, G. G., & Chowdhury, S. (1999). Digital library research: major issues and trends. Journal of documentation, 55(4), 409-448.
3. Witten, I. H., Bainbridge, D., & Nichols, D. M. (2009). *How to build a digital library*. Morgan Kaufmann.
4. Shriwas, M. K., & Raut, V. R. (2015, March). Content based image retrieval: a past, present and new feature descriptor. In *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]* (pp. 1-7). IEEE.
5. da Silva Torres, R., & Falcao, A. X. (2006). Content-based image retrieval: theory and applications. RITA, 13(2), 161-185.
6. Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1-6). Ieee.
7. OpenLibrary.org. (n.d.). Welcome to Open Library | Open Library. Original Content Copyright; 2007-2015. https://openlibrary.org/ (Last accessed 05/21/2023)
8. *IU Library* |. (n.d.). https://library.hcmiu.edu.vn/ (Last accessed 05/21/2023)
9. *Shopee* |. (n.d.). https://shopee.vn/ (Last accessed 05/21/2023)
10. Comic Books Images. (2017, November 21). Kaggle. https://www.kaggle.com/datasets/cenkbircanoglu/comic-books-classification (Last accessed 05/21/2023)
11. Book Recommendation Dataset. (2020, November 29). Kaggle. https://www.kaggle.com/datasets/arashnic/book-recommendation-dataset (Last accessed 05/21/2023)
12. A. Aylin Tokuç. (2022, November 9). k-Nearest Neighbors and High Dimensional Data. https://www.baeldung.com/cs/k-nearest-neighbors (Last accessed 05/21/2023)
13. CodeLearn. (n.d.). Thuật Toán K-Nearest Neighbors (KNN) Siêu Cơ Bản. https://codelearn.io/sharing/thuat-toan-k-nearest-neighbors-knn (Last accessed 05/21/2023)
14. Lee, W. Y., Park, S. M., & Sim, K. B. (2018). Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm. Optik, 172, 359-367.

15. GeeksforGeeks. (2019). Activation Functions. GeeksforGeeks. https://www.geeksforgeeks.org/activation-functions/ (Last accessed 05/21/2023)
16. Hassan, M. U. (2021, February 24). VGG16 – Convolutional Network for Classification and Detection. Neurohive. https://neurohive.io/en/popular-networks/vgg16/ (Last accessed 05/21/2023)
17. Phung, V. H., & Rhee, E. J. (2019). A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. Applied Sciences, 9(21), 4500.
18. Donges, N. (2019). What Is Transfer Learning? Exploring the Popular Deep Learning Approach. Built In. https://builtin.com/data-science/transfer-learning(Last accessed 05/21/2023)
19. GeeksforGeeks. (2023). VGG 16 CNN model. GeeksforGeeks. https://www.geeksforgeeks.org/vgg-16-cnn-model/(Last accessed 05/21/2023)
20. Rohit. (n.d.). GitHub - 1297rohit/VGG16-In-Keras: Implementation of VGG16 architecture in Keras. GitHub. https://github.com/1297rohit/VGG16-In-Keras(Last accessed 05/21/2023)
21. Streamlit Docs. (2023, April 27). https://docs.streamlit.io/ (Last accessed 05/21/2023)
22. GeeksforGeeks. (2022). HTML Introduction. GeeksforGeeks. https://www.geeksforgeeks.org/html-introduction/ (Last accessed 05/21/2023)
23. A brief history of CSS until 2016. (n.d.). https://www.w3.org/Style/CSS20/history.html (Last accessed 05/21/2023)
24. ReactJS Introduction - javatpoint. (n.d.). www.javatpoint.com. https://www.javatpoint.com/react-introduction (Last accessed 05/21/2023)
25. Introduction to React. (n.d.). https://www.w3schools.com/react/react_intro.asp (Last accessed 05/21/2023)
26. Introduction to Node.js. (n.d.). Introduction to Node.js. https://nodejs.dev/en/learn/ (Last accessed 05/21/2023)
27. auth0.com. (n.d.). JWT.IO - JSON Web Tokens Introduction. JSON Web Tokens - jwt.io. https://jwt.io/introduction/ (Last accessed 05/21/2023)
28. MongoDB. (n.d.). *MongoDB: The Developer Data Platform*. https://www.mongodb.com/ (Last accessed 05/21/2023)
29. *ImageNet*. (n.d.). https://www.image-net.org/about.php (Last accessed 05/21/2023)

30. *OpenCV: Histograms - 1 : Find, Plot, Analyze !!!* (n.d.). https://docs.opencv.org/3.4/d1/db7/tutorial_py_histogram_begins.html (Last accessed 05/21/2023)
31. Latif, A., Rasheed, A., Sajid, U., Ahmed, J., Ali, N., Ratyal, N. I., ... & Khalil, T. (2019). Content-based image retrieval and feature extraction: a comprehensive review. *Mathematical problems in engineering*, *2019*.
32. Alzu'bi, A., Amira, A., & Ramzan, N. (2015). Semantic content-based image retrieval: A comprehensive study. Journal of Visual Communication and Image Representation, 32, 20-54.
33. Razzak, M. I., Naz, S., & Zaib, A. (2018). Deep Convolutional Neural Networks for CBIR: A Survey. Journal of Computer Science and Technology, 33(1), 11-30.

APPENDIX

Name	UC-10: Login
Actor	Admin/User
Preconditions	(none)
Trigger	Admin/User wants to log in to the website.
Postconditions	Admin/User go to Website Homepage.
Main Scenario	1. Admin/User enters the username and password. -- Verify if the username and the password are correct, if it goes to step 2. 2. The system shows the homepage with their login.
Extensions	1a. If the username and password are incorrect, alert the Admin/User and the use case ends.

Appendix 1: Description for “Login” use case

Name	UC-11: Login Admin Dashboard
Actor	Admin
Preconditions	(none)
Trigger	Admin/User wants to log in to the Admin Dashboard.
Postconditions	Admin/User goes to Dashboard Homepage.
Main Scenario	1. Admin enters the username and password. -- Verify if the username and the password are correct, then it goes to step 2. 2. The system shows the Dashboard Homepage with their login.
Extensions	1a. If the username and password are incorrect, alert the Admin and the use case ends.

Appendix 2: Description for “Login Admin Dashboard” use case

Name	UC-12: View Book List
Actor	Admin
Preconditions	Admin/User logged in the Admin Dashboard.
Trigger	Admin wants to check/edit the book list.
Postconditions	A list of books is shown.
Main Scenario	1. Click “Book” on the Sidebar. 2. The system shows the list of books in the database.
Extensions	(none)

Appendix 3: Description for “View Book List” use case

Name	UC-13: Delete Book
Actor	Admin
Preconditions	Admin logged in and views list of books.
Trigger	Admin wants to delete a book.
Postconditions	A selected book is deleted and updated in the database.
Main Scenario	1. Admin chooses a book that they want to delete. 2. Click the “Delete” button on the book they want to delete. 3. The system deletes that book and saves it to the database.
Extensions	(none)

Appendix 4: Description for “Delete Book” use case

Name	UC-14: Edit Book
Actor	Admin
Preconditions	Admin logged in “Cloud Database” and views the list of books.
Trigger	Admin wants to edit the book’s information.

Postconditions	Information of the selected book is edited and updated in the database.
Main Scenario	1. Admin chooses a book that they want to edit. 2. Admin updates new content for what book information they want to change. 3. Click “Update” and the system will save it to the database.
Extensions	(none)

Appendix 5: Description for “Edit Book” use case

Name	UC-15: Create New Book
Actor	Admin
Preconditions	Admin logged in and views the list of books.
Trigger	Admin wants to create a new book.
Postconditions	A new book is created and added to the database.
Main Scenario	1. Click the “Add New” button. 2. The system moves to an Add New Book page with a form for Admin to create a new book. 3. Admin enters content on all information of a user account that they want to create. 4. Click “Ok” and the system will create and save it to the database.
Extensions	(none)

Appendix 6: Description for “Create New Book” use case

Name	UC-16: View Slot List
Actor	Admin
Preconditions	Admin/User logged in the Admin Dashboard.
Trigger	Admin wants to check/edit the slot list.
Postconditions	A list of slots is shown.

Main Scenario	1. Click “Slot” on the Sidebar. 2. The system shows the list of slots in the database.
Extensions	(none)

Appendix 7: Description for “View Slot List” use case

Name	UC-17: Delete Slot
Actor	Admin
Preconditions	Admin logged in and views list of slots.
Trigger	Admin wants to delete a slot.
Postconditions	A selected slot is deleted and updated in the database.
Main Scenario	1. Admin chooses a slot that they want to delete. 2. Click the “Delete” button on the slot they want to delete. 3. The system deletes that slot and saves it to the database.
Extensions	(none)

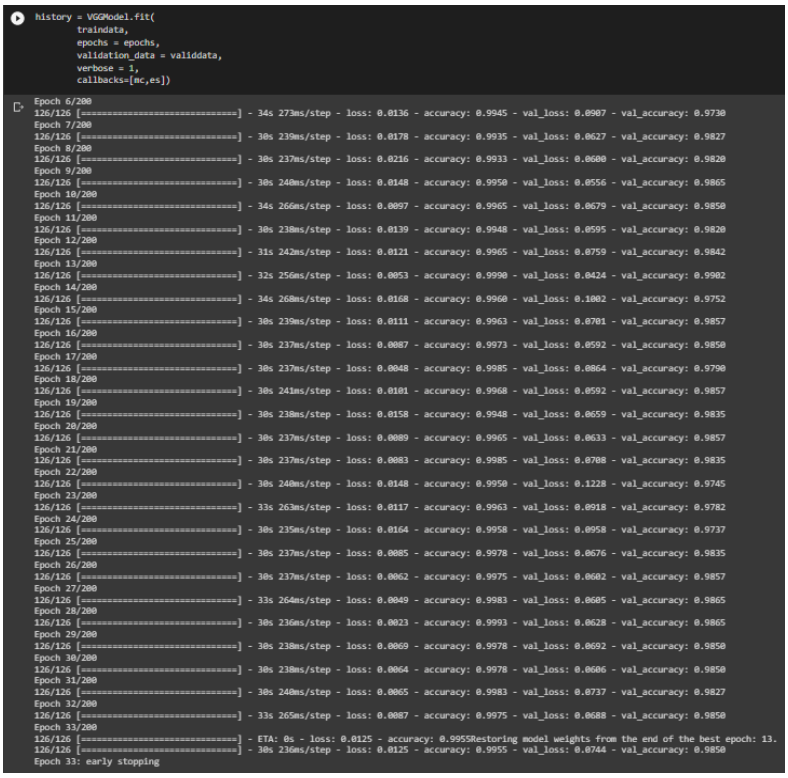
Appendix 8: Description for “Delete Slot” use case

Name	UC-18: Edit Slot
Actor	Admin
Preconditions	Admin logged in and views list of slots.
Trigger	Admin wants to edit the slot’s information.
Postconditions	Information of the selected slot is edited and updated in the database.
Main Scenario	1. Admin chooses a slot that they want to edit. 2. Admin updates new content for what slot information they want to change. 3. Click “Update” and the system will save it to the database.
Extensions	(none)

Appendix 9: Description for “Edit Slot” use case

Name	UC-19: Create New Slot
Actor	Admin
Preconditions	Admin logged in and views the list of slots.
Trigger	Admin wants to create a new slot.
Postconditions	A new slot is created and added to the database.
Main Scenario	1. Click the “Add New” button. 2. The system moves to an Add New Book page with a form for Admin to create a new slot. 3. Admin enters content on all information of a slot that they want to create. 4. Click “Ok” and the system will create and save it to the database.
Extensions	(none)

Appendix 10: Description for “Create New Slot” use case



Appendix 11: Model training process

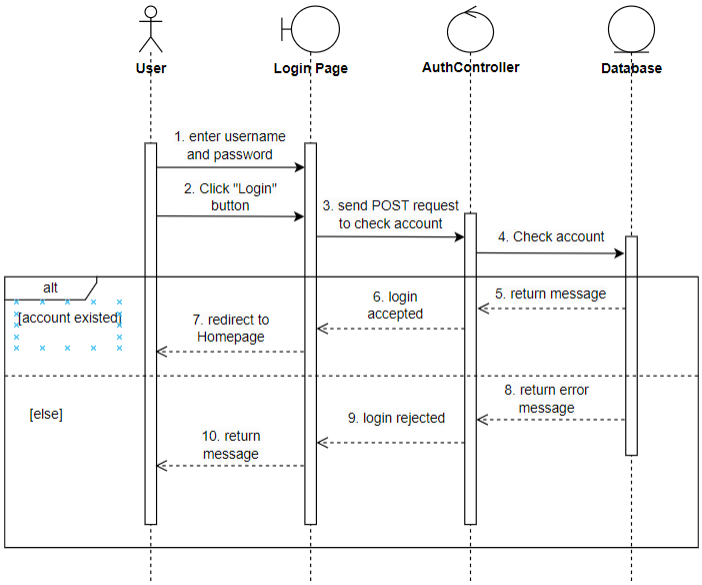
```
import Featured from "../../components/featured/Featured";
import Header from "../../components/header/Header";
import Navbar from "../../components/navbar/Navbar";
import PropertyList from "../../components/propertyList/PropertyList";
import FeaturedProperties from "../../components/featuredProperties/FeaturedProperties";
import "./home.css";
import MailList from "../../components/mailList/MailList";
import Footer from "../../components/footer/Footer";

const Home = () => {
  return (
    <div>
      <Navbar/>
      <Header/>
      <div className="homeContainer">
        <Featured/>
        <h1 className="homeTitle">Browse by property type</h1>
        <PropertyList/>
        <h1 className="homeTitle">Book recommend</h1>
        <FeaturedProperties />
        <MailList />
        <Footer />
      </div>
    </div>
  )
}

export default Home
```

Appendix 12: Example of the structure of a page

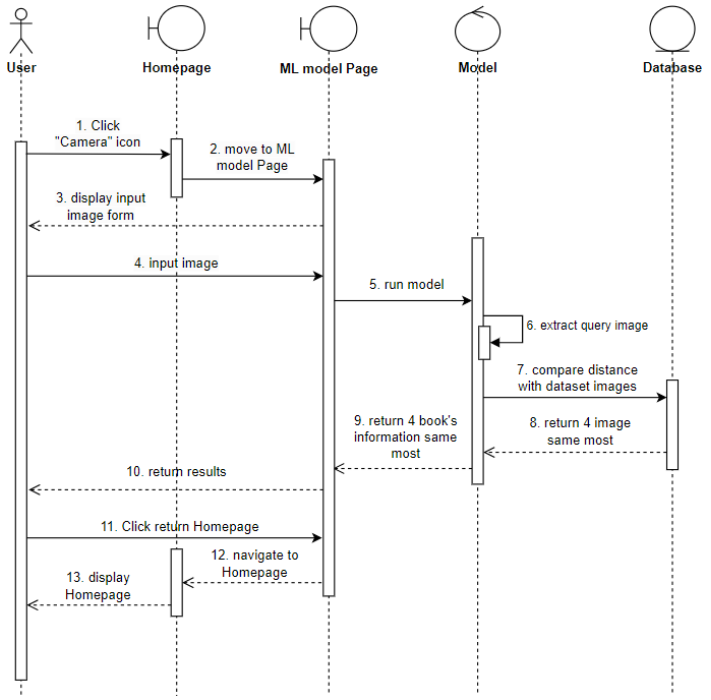
UC-1: Login



Appendix 13: Sequence diagram for “Login”

Appendix 3 shows that this use case is required when user want to log in to the Digital Library or want to borrow a book but have not logged in. The system will display the login page, User is required to enter the login username and password. Then, the system will use the input username and password to use the test API, if both the username and password are correct, the user will be redirected to the Homepage. If 1 of the 2 variables has an incorrect result, the system will refuse to log in and display a message to the user.

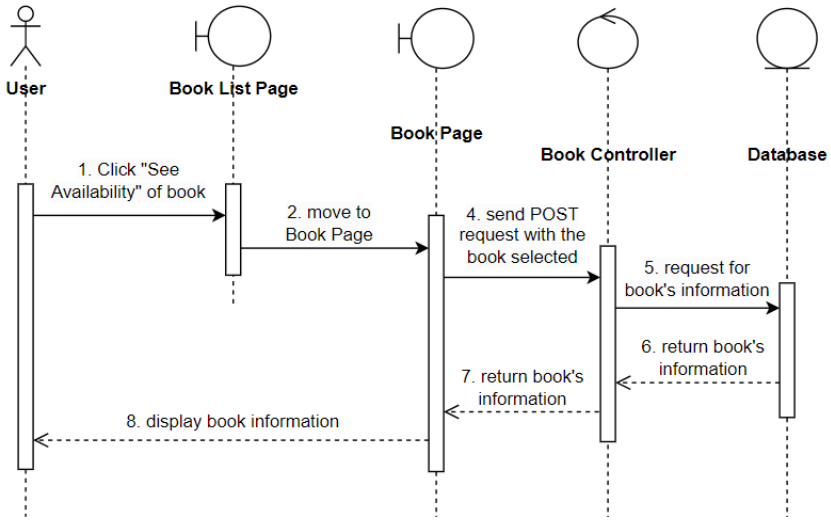
UC-4: Search Image



Appendix 14: Sequence diagram for "Search Image"

Appendix 4 shows that this use case is required when the user has a need to search for book information by image. First, user clicks on the "camera" icon and directly navigate to a web page running a machine learning model. On this page, users need to provide the book cover image they want to search, the system will run a pre-trained model to extract the features of the image to compare with all processed images in the data. After the comparison is complete, the system will return the information of 4 books with images that are closest to the input image provided by the user. Finally, the user can return to the Homepage to search for books based on the information provided by the model.

UC-5: View Book Information



Appendix 15: Sequence Diagram for "View Book Information"

Appendix 5 shows that this use case is required when the user has selected a book and wants to see its details. First, the user will select the "See availability" button to redirect to the information page of the book he has selected. The system will send a request to the database to return all the book information requested by the user. Finally, the user will see all the information of the book that the user wants to see.

```
[ ]  for (i,layer) in enumerate(VGGModel.layers):
         print(str(i) + " "+ layer.__class__.__name__, layer.trainable)
     print(VGGModel.summary())
```

```
0 Functional True
1 MaxPooling2D True
2 Flatten True
3 Dense True
4 BatchNormalization True
5 Dropout True
6 Dense True
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 max_pooling2d_2 (MaxPooling  (None, 3, 3, 512)        0
 2D)

 flatten_2 (Flatten)         (None, 4608)              0

 dense_4 (Dense)             (None, 2048)              9439232

 batch_normalization_2 (Batc  (None, 2048)             8192
 hNormalization)

 dropout_2 (Dropout)         (None, 2048)              0

 dense_5 (Dense)             (None, 2)                 4098

=================================================================
Total params: 24,166,210
Trainable params: 9,447,426
Non-trainable params: 14,718,784
_________________________________________________________________
None
```

Appendix 16: The training model structure in coding