VIETNAM NATIONAL UNIVERSITY OF HO CHI MINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A WEB APPLICATION FOR ADMINISTRATIVE DOCUMENT DIGITIZATION

By

VŨ NHẬT THANH

A THESIS SUBMITTED TO THE SCHOOL OF COMPUTER SCIENCE AND ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF COMPUTER SCIENCE

Ho Chi Minh City, Vietnam

07/2020

A WEB APPLICATION FOR ADMINISTRATIVE DOCUMENT DIGITIZATION

APPROVED BY ADVISOR	APPROVED BY COMMITTEE
	____________________________
	____________________________
	____________________________
	____________________________
____________________________	____________________________
Nguyen Van Sinh, Ph.D.	Thesis Committee

ACKNOWLEDGMENTS

This is a good chance to express my gratitude to everyone who supported me. I am very grateful for the guidance and advice that they gave me during this thesis project.

Firstly, I would like to express my sincere appreciation to Dr. Nguyen Van Sinh, my supervisor for his guidance, suggestion, and patience during the Thesis development. His constant encouragement and support helped me so much to achieve this goal.

Next, I would like to thank my parents who gave me a chance to study at this school. They are always by my side when all the difficult things come to me. They help me stand up to complete this road.

Last but not least, I want to give my thanks to all of my friends at the International University for all the support and good memories that they gave me when we studied together.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

API:	Application Programming Interface
ASCII:	American Standard Code for Information Interchange
CL:	Computer Language
CPU:	Central Processing Unit
CSS:	Cascading Style Sheets
CV:	Computer Vision
DB:	Database
DOC:	Document
DPI:	Dots Per Inch
GPU:	Graphics Processing Unit
GUI:	Graphical User Interface
HTTP:	Hypertext Transfer Protocol
JPG:	Joint Photographic Experts Group
JS:	JavaScript
JSON:	JavaScript Object Notation
MIT:	Massachusetts Institute of Technology
MVC:	Model-View-Controller
NPM:	Node Package Manager
OCR:	Optical Character Recognition
PDF:	Portable Document Format
PNG:	Portable Network Graphics
PSM:	Page Segmentation Method
RAM:	Random Access Memory
SQL:	Structured Query Language
TS:	TypeScript
URL:	Uniform Resource Locator
XML:	Extensible Markup Language

ABSTRACT

Document digitization is one of the emerging trends of digitization and no more a new concept in the information science field. The digitization of documents allows retrieving of information from a paper document. The document digitization process involves conversion processing of the obtained image to extract information. The digitized document can be directly applied to searching, sorting, and storage stage. As the digitization literature is surveyed, the process of extraction of digital string from scanned paper documents plays a very important role in the document digitization process.

In this thesis, we will research and create an implementation application for processing the administrative documents, which used for management storing and searching. This application will focus on processing and digitizing the document. The method consists of the following steps: Obtaining data by using a scanner to scan administrative documents. Processing obtained data to remove noise and picking up the information, the content of the document is necessary. The application picks up the data based on the structure of documents following the criteria of the Vietnam government.

CHAPTER 1

INTRODUCTION

1.1 Background

Digitization of documents is the method of changing data within the document into a digital format. In this format, information is organized into discrete units of data that can be separately addressed. This is the kind of data that a computer can understand and process. The document can be digitized by the following steps: a scanner captures a document and converts it to an image file. An optical character recognition engine analyzes the light and dark areas to identify every single letter or digit and converts it into an ASCII code. After that, these codes will go through a process of analyzing and grouping into many small parts that can be stored for later use.

1.2 Problem statement

With the development of information technology, people are becoming more and more familiar with the utility that digital technology brings us. It leads to the fact that today's traditional documents cannot process to meet the demands of working time and space. Some surveys show that a typical employee spent half of working time to organize, store, and keep documents in the cabinet. These tasks waste a lot of time, space and there is a risk of document loss.

To carry out these problems, an application in computer-based needs to be created to process all traditional documents into a digital document system. In this system, the documents are organized in a data structure to serve the professional task of administrative staff. Therefore it can help to keep information on the organization, easy to use for every people.

1.3 Scope and objectives

The objective of this thesis is to build a web application for Administrative Document Digitization. The thesis provides a tool that helps organizations to manage administrative documents by scanning and digitizing all of them, which can help to handle processing, maintaining, and finding documents easily. Also, it provides an ability for users to export the document in a printable format for use.

The input data are high quality images of Vietnamese administrative documents including NGHỊ QUYẾT, THÔNG BÁO, QUYẾT ĐỊNH, KẾ HOẠCH and CÔNG VĂN. Their structure has the following form:

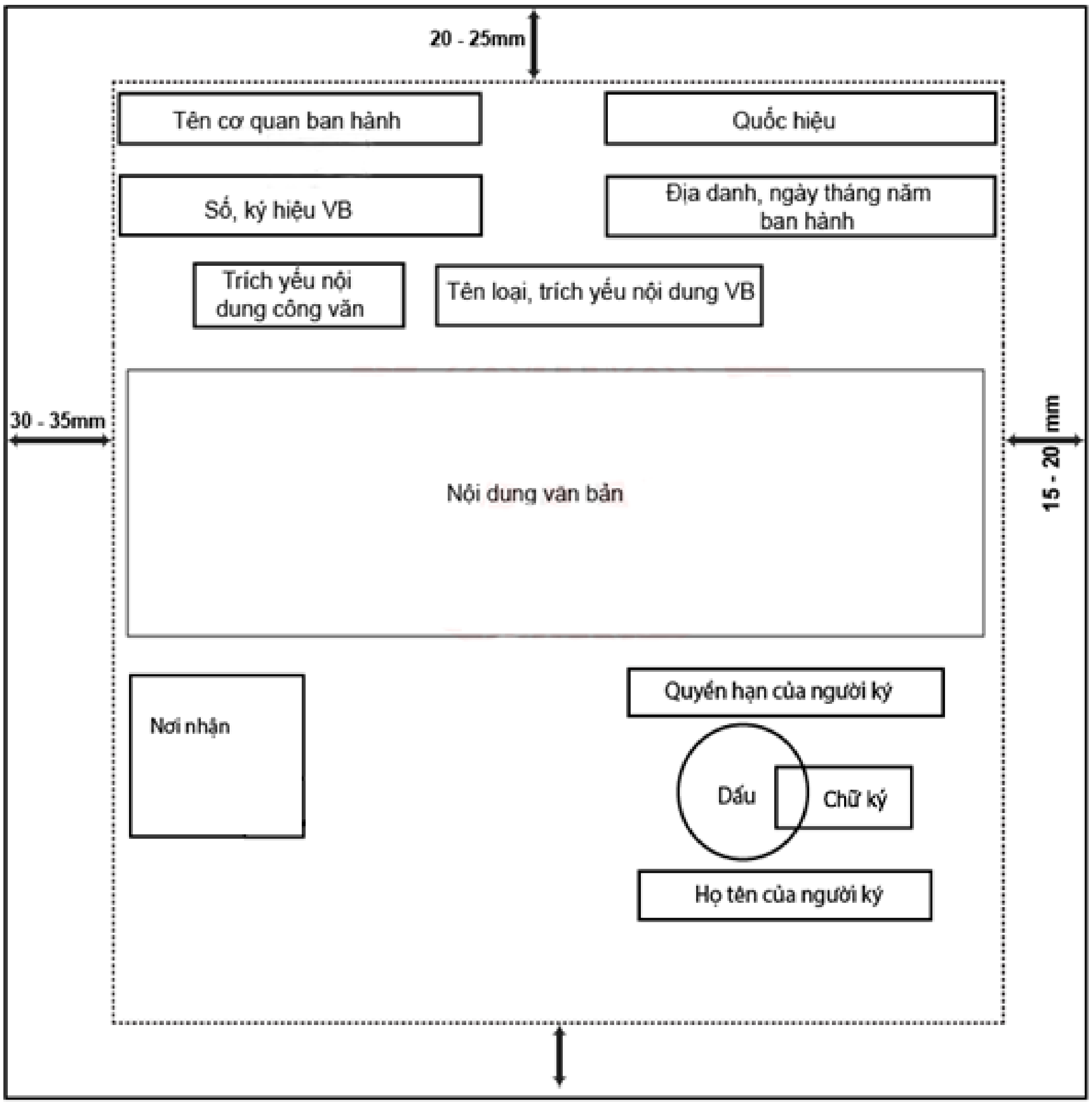


Figure 1.1: A structure of document

1.4 Assumption and Solution

Document digitization project covers many problems. To reduce the amount of work, a good choice is to learn how to use and combine the existing library and framework related to this project.

1.5 Structure of the thesis

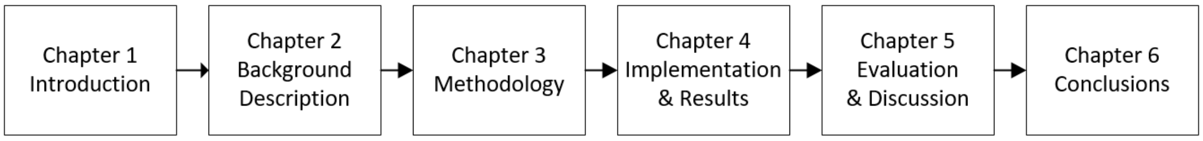


Figure 1.2: Structure of the Thesis

Chapter 1 introduces about thesis including tools and techniques that we base on to make this project, why we need this application, and what problem it can solve in real life.

Chapter 2 describes related technologies that the application has to use and gives detail information about some existing Document Digitization applications. Besides, it also shows the advantages and disadvantages of the application.

Chapter 3 focuses on the method and its description in detail that helps to explain what the user can do in this application and the functions it can support.

Chapter 4 shows all the functions that have been implemented in the project. It includes some pictures to display the application results.

Chapter 5 includes the evaluation and discussion of the final result we get from the previous chapter.

Chapter 6 is the summary of all processes in the thesis. What we have learned to the end of the project, which experiences while the project has been developed. The work we should do in the future to improve the application more effectively.

CHAPTER 2

BACKGROUND DESCRIPTION

2.1 Related technologies

2.1.1 NodeJS and Express [1]

NodeJS is an open-source server environment that can run on various platforms such as Windows, Linux, macOS... This environment includes everything we need to execute a program written in Javascript.

The existence of NodeJS makes something that only works on the browser becomes a standalone application thanks to Chrome's V8 Javascript runtime engine [2], which can convert Javascript into machine code.

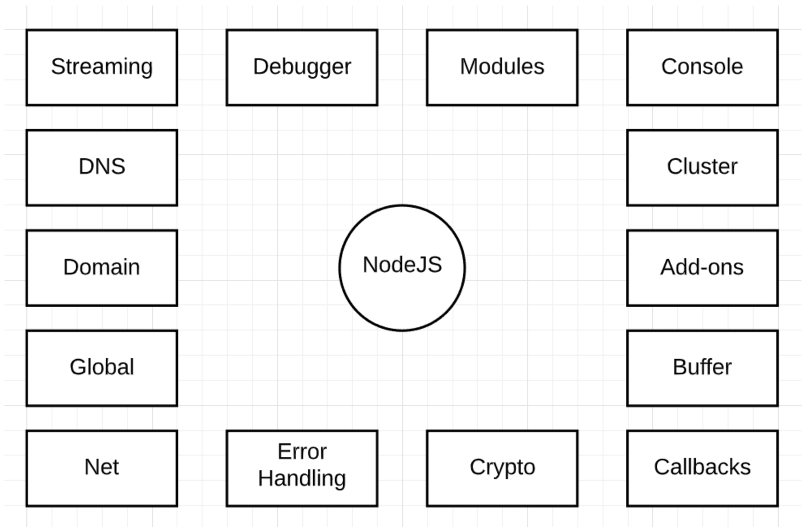


Figure 2.1: The concept of NodeJS

NodeJS has some important features that help it become the first choice of software architects:

- Highly performance
- Asynchronous and Event-Driven
- Single thread with Highly Scalable
- No Buffering
- License

Express is a flexible web application framework of NodeJS which gives robust features for use of the web as well as mobile applications. Express can be used to set up middlewares to respond to HTTP Requests and defines a routing table that is used to perform different actions based on HTTP Method and URL. It also allows us to dynamically render HTML Pages based on passing arguments to templates.

2.1.2 ReactJS [3]

ReactJS is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. React was created by Jordan Walke, a software engineer working for Facebook and it's first deployed on Facebook's newsfeed in 2011 and on Instagram.com in 2012.

React allows developers to create large web applications that can change data without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template. It can be used with a combination of other JavaScript libraries or frameworks, such as AngularJS in MVC.

2.1.3 MongoDB [4]

MongoDB came into light around the mid-2000s and it is an open-source NoSQL database used for high volume data storage.

It is designed in an object-oriented manner, the tables in MongoDB are the very flexible structure, allowing the data stored on the table does not need to follow a certain structure at all (this is suitable for making big data). MongoDB stores data in the direction of the document, the data is stored in the JSON-style document so the query will be very fast.

Advantages of MongoDB:

- Flexible schema: Each collection will have different sizes and documents
- Clear object structure: Although the structure of the data is flexible, its object is clearly defined
- Using internal memory: the query will be very fast

- Easy to expand
- No joins: Contributes to a very fast query speed on MongoDB

MongoDB is suitable for realtime applications.

2.1.4 Docx library [5]

Docx is a library that has modules to create, read, and write DOC files. These are referred to as ‘WordML’, ‘Office Open XML’, and ‘Open XML’ by Microsoft. They also validate as well-formed XML.

DocxJS can easily generate a DOC file with JS/TS that works for node and on the browser.

2.1.5 OpenCV[6]

OpenCV (Open Source Computer Vision) is considered one of the leading open-source libraries for image processing.

OpenCV officially launched in 1999 and it is free for both academic and commercial use. It supports multiple platforms including Windows, Linux, macOS, iOS, and Android with many programming languages like C/C++, Python, Java, and Javascript.

OpenCV is written in C/C++ and integrates OpenCL, so it has good performance that can be used with real-time related applications.

The library has more than 2500 optimized algorithms, most of them have contributed to developing many fields such as:

- Photo recognition
- Image processing
- Recover photos / videos
- Virtual reality
- Other applications

In this thesis, we will use some of its powerful processing functions to identify the data area in the image as well as improve the quality of the material to achieve the best results, including: find contours, dilate, and threshold, and Hough circle algorithm.

2.1.6 Tesseract OCR [7]

Tesseract is an open-source OCR engine. Its development has been sponsored by Google. This engine is available for Windows, Linux, and Mac OS X

Tesseract is not supplied with a GUI and can only be executed from the command-line interface. The first version of Tesseract was only able to convert English-language from image to text. But when version 4 has been released, it now can support over 116 languages.

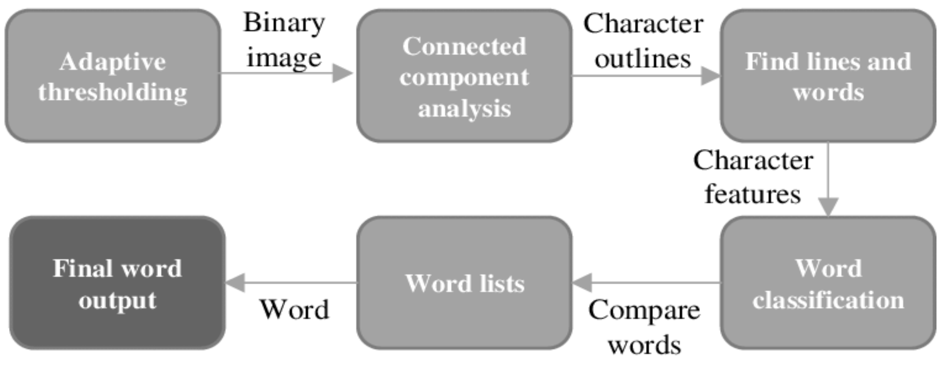


Figure 2.2: Tesseract workflow

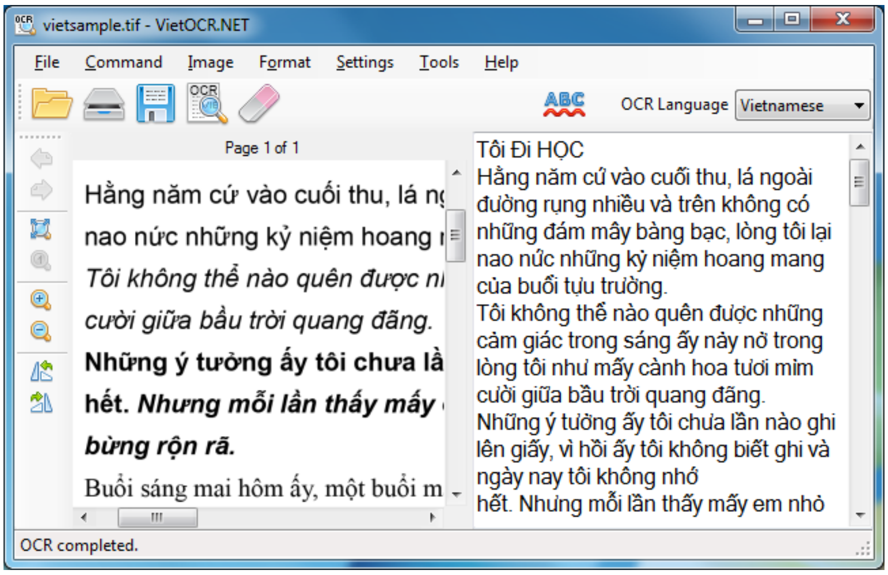
To detect and convert the imaged text to string type, Tesseract will try to find and organize text lines into blobs, and the lines and regions are analyzed for fixed proportional text. Base on the kind of character spacing, text lines are split into words. Then, an attempt is made to recognize each word in turn. Each word is passed to an adaptive classifier as training data. This classifier then enhances the accuracy to recognize the text lower down the page. When the adaptive classifier has qualified to contribute to the top of the page, a second pass is run over the page to recognize again the words that were not recognized well enough from the beginning.

Although the newest version of Tesseract has a better performance after adding a new training model with a lot of data and fonts. It’s still not good enough to handle the strange font and handwritten text.

2.2 Existing document digitization applications

2.2.1 VietOCR [8]

VietOCR is a GUI frontend for the Tesseract OCR engine. This program can recognize text from images of common formats. It can also function as a console application, which can be executed from the command line.



OCR Language Vietnamese

Hằng năm cứ vào cuối thu, lá ng
nao nức những kỷ niệm hoang r
Tôi không thể nào quên được nh
cười giữa bầu trời quang đãng.
Những ý tưởng ấy tôi chưa lầ
hết. *Nhưng mỗi lần thấy mấy*
bừng rộn rã.
Buổi sáng mai hôm ấy, một buổi m

Tôi Đi HỌC
Hằng năm cứ vào cuối thu, lá ngoài đường rụng nhiều và trên không có những đám mây bàng bạc, lòng tôi lại nao nức những kỷ niệm hoang mang của buổi tựu trường.
Tôi không thể nào quên được những cảm giác trong sáng ấy nảy nở trong lòng tôi như mấy cành hoa tươi mỉm cười giữa bầu trời quang đãng.
Những ý tưởng ấy tôi chưa lần nào ghi lên giấy, vì hồi ấy tôi không biết ghi và ngày nay tôi không nhớ
hết. Nhưng mỗi lần thấy mấy em nhỏ

OCR completed.

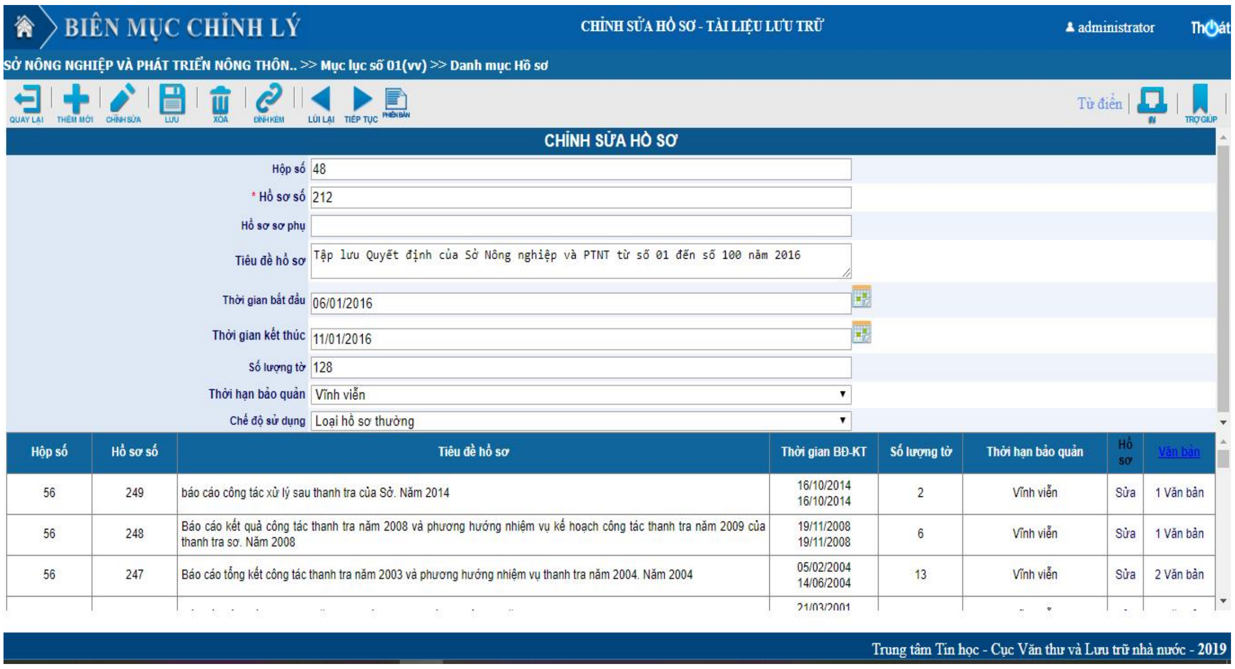
Figure 2.3: VietOCR Interface

VietOCR supports several languages, including Vietnamese. The Vietnamese are trained with Arial, Times New Roman, Courier New, and Verdana fonts. Therefore, images having similar font glyphs can be recognized with a better success rate. OCRing images that have font glyphs look different from the supported fonts generally will require training Tesseract to create another language data pack specifically for those typefaces. This application also provides a tool to merge several images into a single PDF file for convenient OCR operations or to split a PDF file into smaller ones.

- Advantage:
    - Support multiple languages
    - High performance
- Disadvantage:
    - Components in the document cannot be recognized
    - No data storage and management after the conversion.

2.2.2 Electronic Document Management Software [9]

This software is written on Microsoft ASP.NET technology and it is built with a 3-layer architecture (Interface layer, processing layer and database layer), its system consists of functional modules and integrated into a unified system. The software can be exploited on LAN. WAN, or the Internet.



CHỈNH SỬA HỒ SƠ

Hộp số	48
* Hồ sơ số	212
Hồ sơ sơ phụ	
Tiêu đề hồ sơ	Tập lưu Quyết định của Sở Nông nghiệp và PTNT từ số 01 đến số 100 năm 2016
Thời gian bắt đầu	06/01/2016
Thời gian kết thúc	11/01/2016
Số lượng tờ	128
Thời hạn bảo quản	Vĩnh viễn
Chế độ sử dụng	Loại hồ sơ thường

Hộp số	Hồ sơ số	Tiêu đề hồ sơ	Thời gian BĐ-KT	Số lượng tờ	Thời hạn bảo quản	Hồ sơ	Văn bản
56	249	báo cáo công tác xử lý sau thanh tra của Sở. Năm 2014	16/10/2014 16/10/2014	2	Vĩnh viễn	Sửa	1 Văn bản
56	248	Báo cáo kết quả công tác thanh tra năm 2008 và phương hướng nhiệm vụ kế hoạch công tác thanh tra năm 2009 của thanh tra sơ. Năm 2008	19/11/2008 19/11/2008	6	Vĩnh viễn	Sửa	1 Văn bản
56	247	Báo cáo tổng kết công tác thanh tra năm 2003 và phương hướng nhiệm vụ thanh tra năm 2004. Năm 2004	05/02/2004 14/06/2004	13	Vĩnh viễn	Sửa	2 Văn bản
			21/03/2001				

Trung tâm Tin học - Cục Văn thư và Lưu trữ nhà nước - 2019

Figure 2.4: GUI of Electronic Document Management Software

Electronic Document Management Software provides multiple functions to interact with the document, such as document management, document splitting, user management… and especially it also has tools to extract text from the document.

The conversion tool helps the user automatically detect and convert many fields, those will be used as finding keywords for the future uses. Moreover, the user can manually convert the document whenever the result of the software is unsatisfactory.

After the conversion process, the text will be saved in a common data storage center and divided into partitions for each locality.

CHAPTER 3

METHODOLOGY

3.1 Overview

In this chapter, the method applied to the project will be shown. The first section will present the idea that will be used to handle the conversion process. The next section provides information on user requirements and system architecture. The database model, user interface, activity and sequence diagrams will be shown in the last section.

3.2 Proposed method for the conversion process

In this section, we research on a method to detect and extract the information from the components of the document.

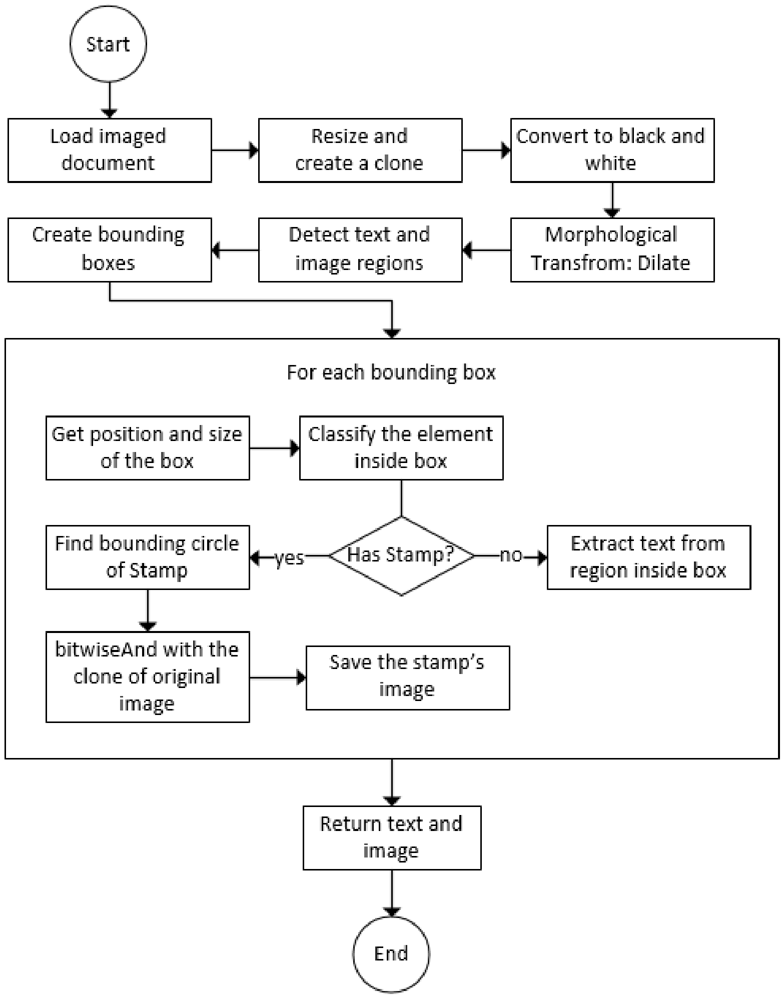

Start
Load imaged document
Resize and create a clone
Convert to black and white
Morphological Transfrom: Dilate
Detect text and image regions
Create bounding boxes
For each bounding box
Get position and size of the box
Classify the element inside box
Has Stamp?
yes
no
Find bounding circle of Stamp
Extract text from region inside box
bitwiseAnd with the clone of original image
Save the stamp's image
Return text and image
End

Figure 3.1: Conversion process workflow

After receiving and loading imaged document for the conversion process, we need to create a clone of the imaged document (this one will be used later for finding the stamp), then we convert the imaged document into a grayscale image, combined with threshold function to filter out noise elements[10]. Understandably, the threshold process is to change the value of the pixels, if the pixel value exceeds the threshold value, it is assigned the value 1 (white), otherwise, it will be set to 0 (black).

Next, to be able to identify each component of the document in the image, we need to fill in the spaces between the characters to form a uniform partition (Using the dilate method to increase the character thickness to a specific level).

Then, we identify 4 points of the rectangular area surrounding partitions by finding the contours [11].

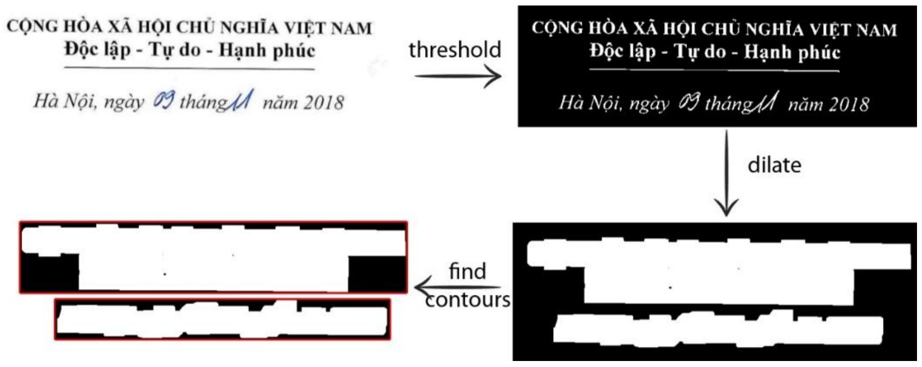

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc
Hà Nội, ngày 09 tháng 11 năm 2018
threshold
CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc
Hà Nội, ngày 09 tháng 11 năm 2018
dilate
find contours

Figure 3.2: Detecting component process

From the four inferred points x, y coordinates and partition size (width, height), from which we can guess the location of the partition corresponding to which components in the image (that can be organization, id, place and date, document type, abstract, content, recipient, position, stamp, or name). If the found component is the stamp, we continue to find the circle shape which is the border of the stamp, using the RGB filter to keep the red color only and save it as an PNG image, this image will not be used to extract the text.

After getting the location of each component, we crop the image of each component out of the clone (which is created at the beginning) and save it as a set of images with the name of the role of that component, the purpose is to easily handle each component individually.

Finally, we use the OCR engine to convert image of each found component into text and return the result.

The psuedo-code for the conversion process operates as follows:

```
//Conversion
Input: Image file (png/jpg)
Output: Image file (png)
        Array of Strings
Method:
    Step 1: Load image file
    Step 2: Resize image with fixed width and height
    Step 3: Create clone of the image
    Step 3: Convert image into pure Black and White using threshold function
    Step 4: Dilate character in image
    Step 6: Find contours in image and store the matrix in variable C
    Step 7: Initialize i = 0
    Step 8: While i < C.length:
        Create bounding rectangle with C[i]
        Get x, y, width, height of the bounding rectangle
        Classify the component base on x, y, width, height
            If component is stamp:
                Find mask of bounding circle of stamp
                BitwiseAnd between mask and image's clone to get stamp image
            Else:
                Crop the area at (x,y,x+width,y+height) of the image's clone
                Extract text from the area and push into array texts[]
            i = i + 1
    Step 9: Return stamp's image and texts[]
```

3.3 System architecture

Model-View-Controller is one of the most popular architectures for applications. The MVC model was conceived as a solution to the problem of organizing applications with graphical user interfaces [12].

We will use Express Framework to build our server system base on this model

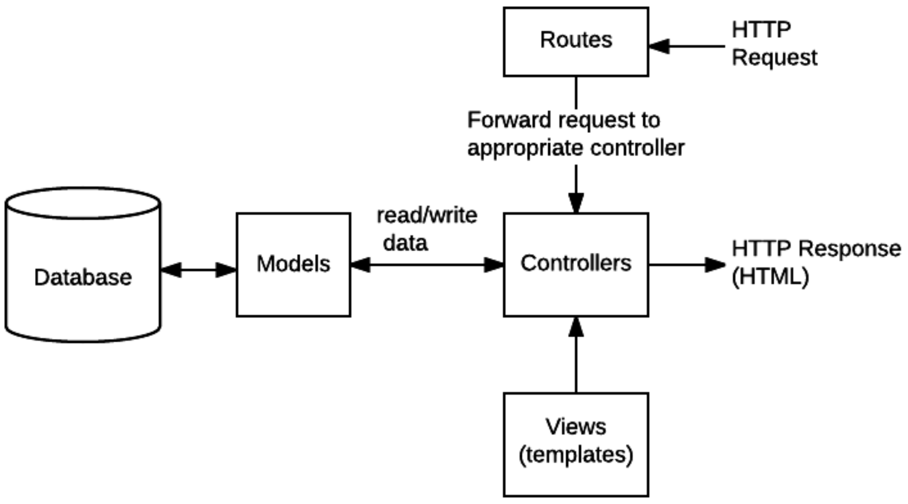


Figure 3.3: MVC model in Express

- **Models**: the part of our application that will deal with the database or any data-related functionality.
- **Views**: everything the user will see. Basically, the pages that we're going to send to the client (in this thesis, the view components will be created using ReactJS)
- **Controllers**: the logic of our site, and the glue between models and views. We call our models to get the data, then we put that data on our views to be sent to the users
- **Routes**: have main function to navigate to a specific Controller which follows a request from the user

3.4 User requirements analysis

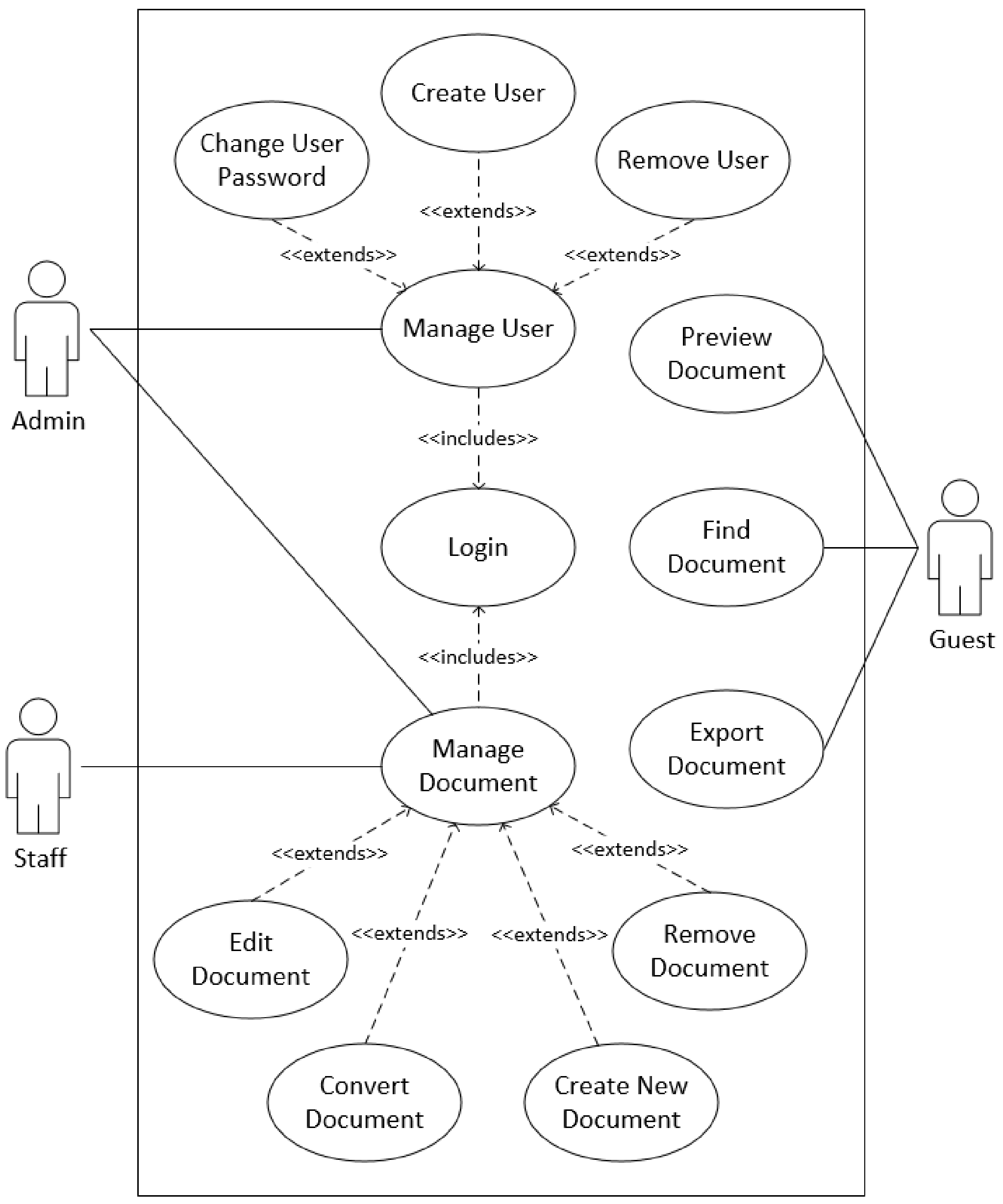


Figure 3.4: Use case diagram

Name	**UC-1: Create New Document**
Summary	Create a new digital document
Rationale	As a user, I want to create a new document that doesn’t exist in both paper form and digital form

Users	Admin, Staff
Events	1. Click the Create button 2. In the Create Page, enter all the required information for the document 3. Click the Save button
Precondition	None
Post conditions	None

Table 3.1: Use case description for UC-1: Create New Document

Name	**UC-2: Convert Document**
Summary	Create a digital document from an imaged document
Rationale	As a user, I want to convert a paper document into a digital document
Users	Admin, Staff
Events	1. Click the Create button 2. In the Create Page, click the Open Image button 3. Select imaged document 4. Select the component (all components in document, id, title...) to convert 5. Locate the document area in the image 6. Click the Convert button 7. After the conversion process, click the Save button
Precondition	Image file (jpg, png) of document
Post conditions	None

Table 3.2: Use case description for UC-2: Convert Document

Name	**UC-3: Edit Document**
Summary	Modify the information of a digital document
Rationale	As a user, I want to fix the errors getting from the creation process
Users	Admin, Staff
Events	1. Access Database page 2. Click the row of the document that is needed to modify 3. Click the Edit button 4. Modify the document 5. Click the Save button
Precondition	Documents and images have been loaded from the database
Post conditions	None

Table 3.3: Use case description for UC-3: Edit Document

Name	**UC-4: Remove Document**
Summary	Remove the document from the database
Rationale	As a user, I want to delete the document that no longer used
Users	Admin, Staff
Events	1. Access Database page 2. Click the row of the document that is needed to delete 3. Click the Edit to enable Remove button 4. Click the Remove button

Precondition	Documents and images have been loaded from the database
Post conditions	None

Table 3.4: Use case description for UC-4: Remove Document

Name	**UC-5: Find Document**
Summary	Find one or a list of documents in the database
Rationale	As a user, I want to find the document by providing identifying characteristics
Users	All users
Events	1. Access Database page 2. Enter the keyword in the filter 3. Click the table header to sort the column in ascending order or descending order
Precondition	Document has been loaded from the database
Post conditions	None

Table 3.5: Use case description for UC-5: Find Document

Name	**UC-6: Preview Document**
Summary	Open and read the document
Rationale	As a user, I want to watch the document in printed form or the original imaged document
Users	All users
Events	1. After finding out the document, click the document's row to open the modal

	2. Switch between digital form and image form by clicking the “See in image/digital form” button
Precondition	Documents and images have been loaded from the database
Post conditions	None

Table 3.6: Use case description for UC-6: Preview Document

Name	**UC-7: Export Document**
Summary	Export the document to a DOC file
Rationale	As a user, I want to export DOC file so that I can copy or print the document
Users	All users
Events	1. Access Database page 2. Click the row of the document that is needed to export 3. Click the Export button
Precondition	Documents and images have been loaded from the database
Post conditions	None

Table 3.7: Use case description for UC-7: Export Document

Name	**UC-8: Create User**
Summary	Add a new user to the database
Rationale	As a user, I want to create a new user account that can handle a specific function

Users	Admin
Events	1. Access User Management page 2. Enter the information of the user in the text fields 3. Click the Create button
Precondition	None
Post conditions	None

Table 3.8: Use case description for UC-8: Create User

Name	**UC-9: Change User Password**
Summary	Change a user password with another password
Rationale	As a user, I want to change the password of the existing user
Users	Admin
Events	1. Access User Management page 2. Click the Edit button in the row of the user that is needed to change password 3. Enter the new password 4. Click the Save button
Precondition	User list has been loaded from the database
Post conditions	None

Table 3.9: Use case description for UC-9: Change User Password

Name	**UC-10: Remove User**
Summary	Remove a user from the database
Rationale	As a user, I want to remove a user account that is no longer used
Users	Admin
Events	1. Access User Management page 2. Click the Remove button in the row of the user that is needed to eliminate
Precondition	User list has been loaded from the database
Post conditions	None

Table 3.10: Use case description for UC-10: Remove User

3.5 Front-end

The frontend will be designed using ReactJS, so we just have one HTML file (single page web application) created by 3 hash routes which point to Create Document component, User Management component, and Document Storage component. Some activity diagrams and sequence diagrams will be shown to describe how the user can interact with the system through these components.

a) **Create New Document**

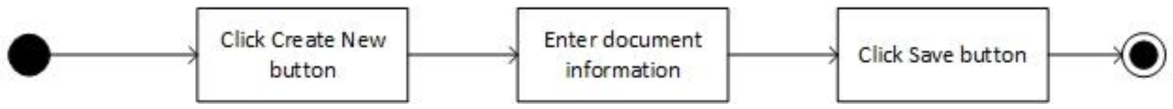


Figure 3.5: Activity diagram for Create New Document function

Description: The diagram figure above describes step by step how admin and staff can create a new document.

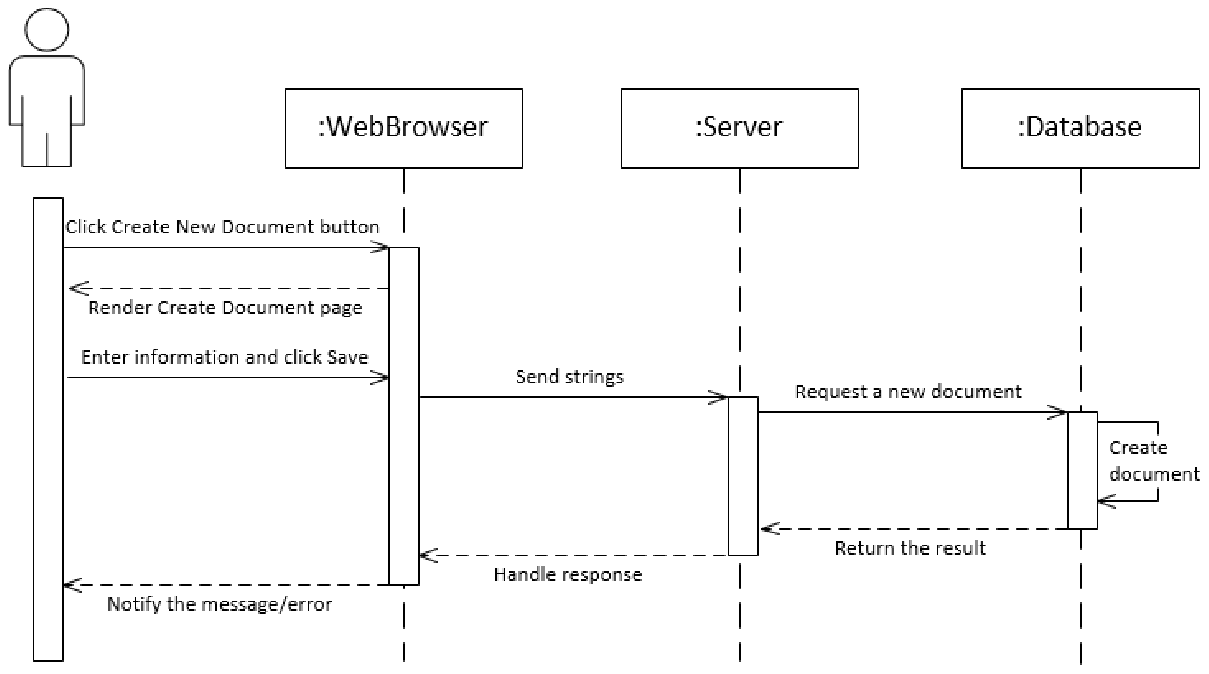


Figure 3.6: Sequence diagram for Create New Document function

Description: When the user goes to the Database page, he or she can access the Create New Document page by clicking the Create New Document button, so that the react component "Create Document page" can be rendered in the web browser. Then, the server will receive the providing information about the document from the user and request the MongoDB server to create a new one.

b) Convert Document

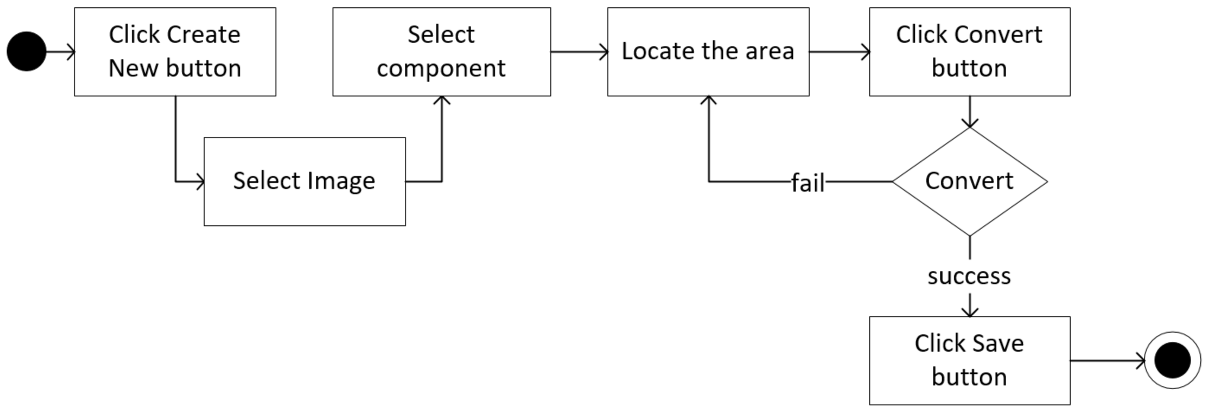


Figure 3.7: Activity diagram for Convert Document function

Description: The diagram figure above describes step by step how admin and staff can do the conversion process to create a new document.

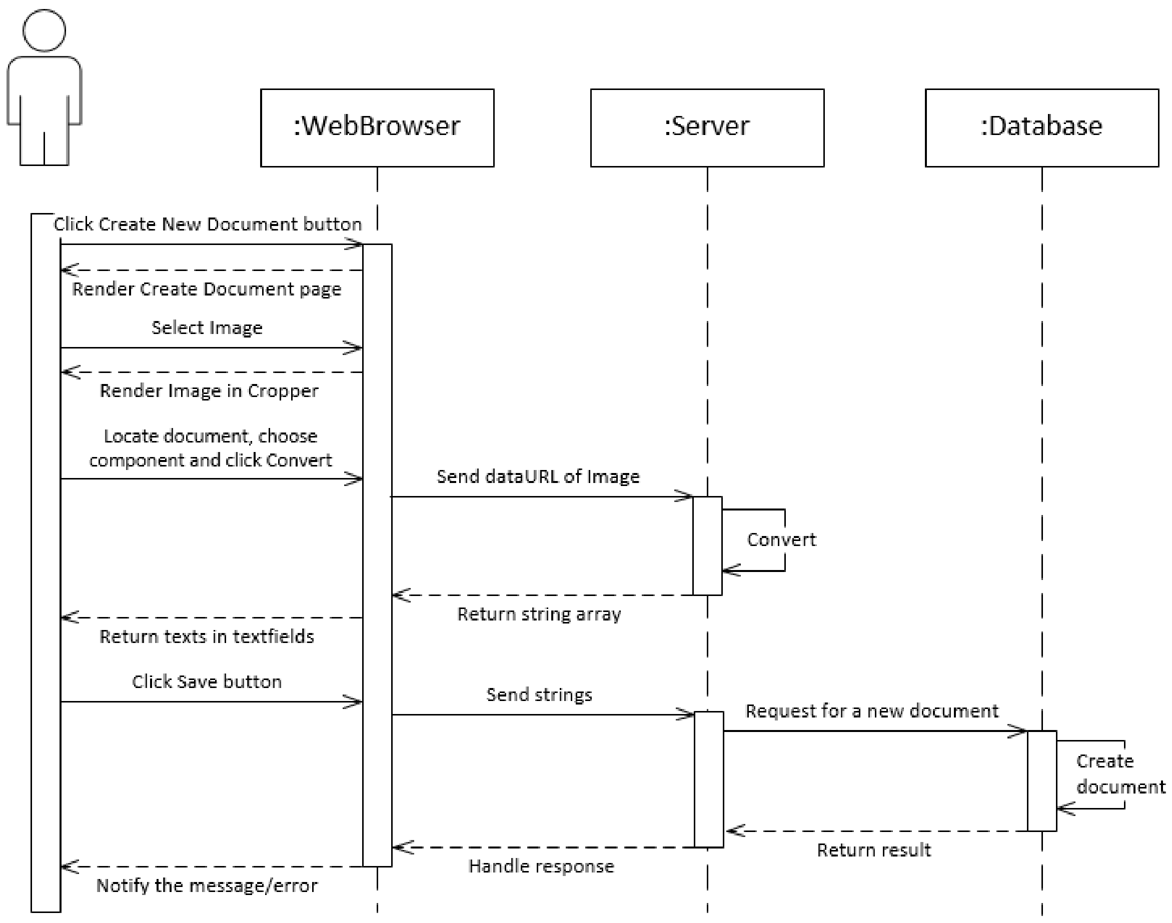


Figure 3.8: Sequence diagram for Convert Document function

Description: When the user accesses the Database page, he or she needs to click the Create New Document button to access Create New Document page, the react-component "Create Page" will be rendered. After that, the user chooses the imaged document from the local storage. Then, they need to select the component to be converted and adjust the cropper box to locate the area of the document in the image. Then, when the user clicks the Convert button, the image will be sent to the server to handle the conversion process and return the string array. Finally, the user checks the result and clicks the save button to ask the server for creating a new document in MongoDB server.

c) Find

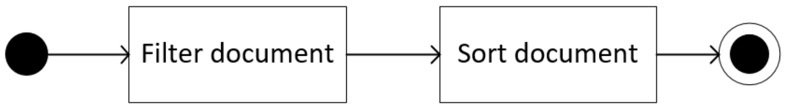


Figure 3.9: Activity diagram for Find function

Description: The diagram figure above describes step by step how the user can find the document.

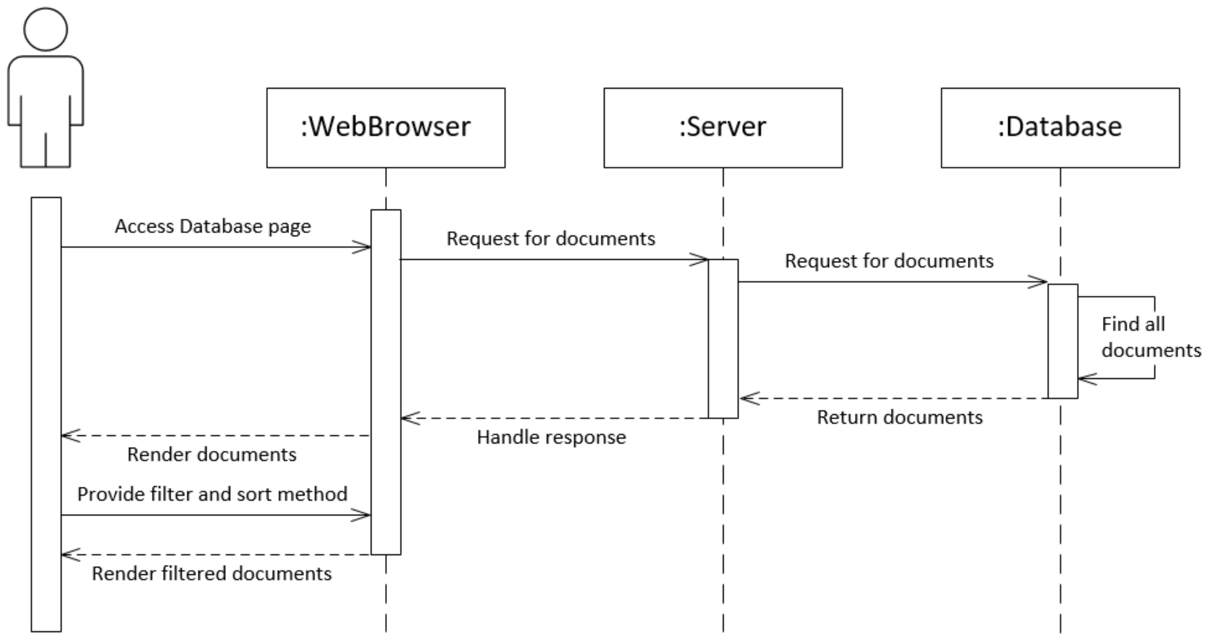


Figure 3.10: Sequence diagram for Find function

Description: After accessing the Database page, a request will be sent to the server to ask for getting the list of documents, which is used to render on the page. When the documents are rendered, the user provides document identification such as keywords and date time to help the filter eliminates irrelevant documents. Finally, the user has a list of documents they want.

d) Preview

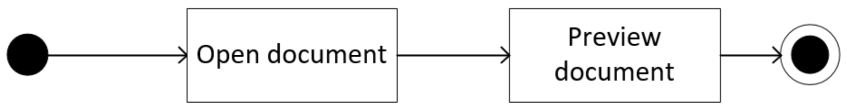


Figure 3.11: Activity diagram for Preview function

Description: The diagram figure above describes step by step how the user can preview the document.

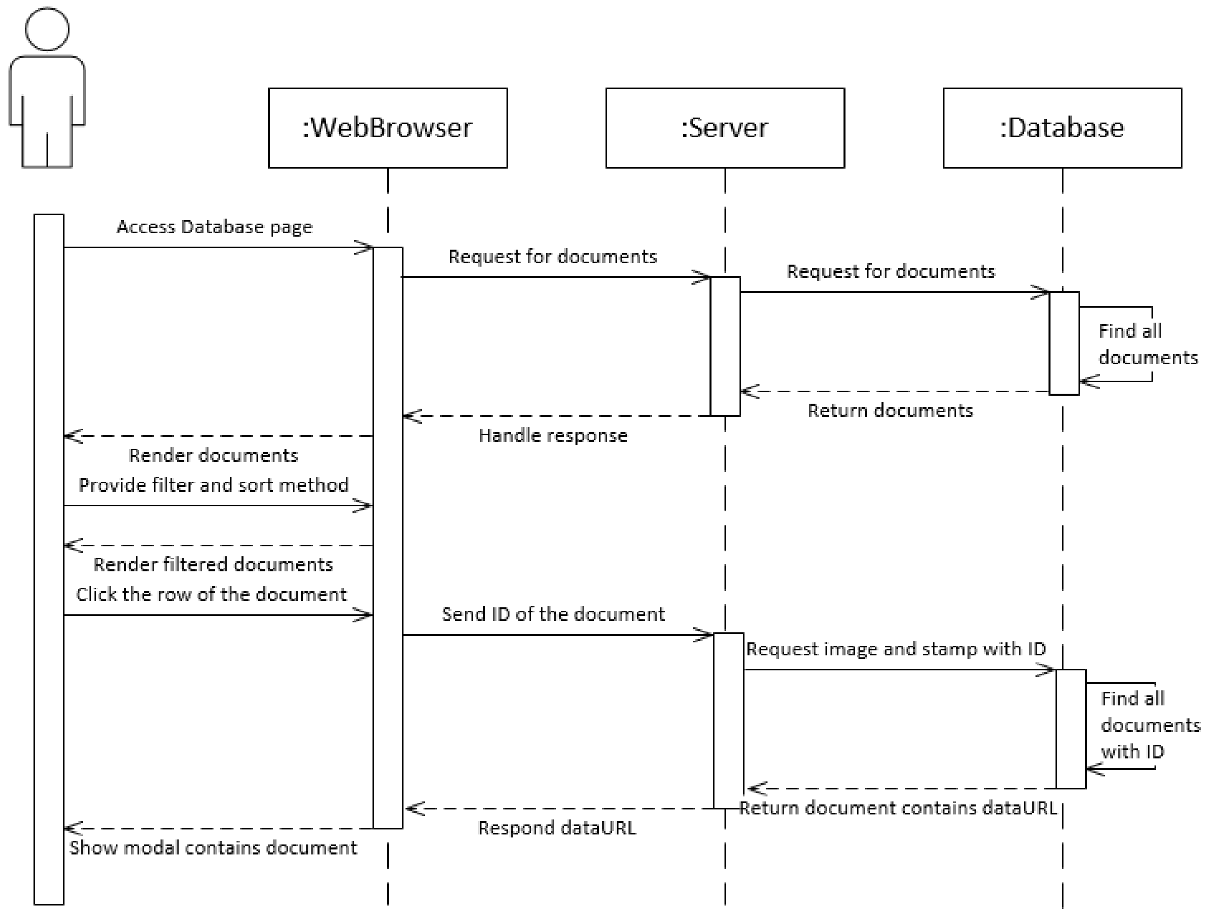

:WebBrowser
:Server
:Database
Access Database page
Request for documents
Request for documents
Find all documents
Return documents
Handle response
Render documents
Provide filter and sort method
Render filtered documents
Click the row of the document
Send ID of the document
Request image and stamp with ID
Find all documents with ID
Return document contains dataURL
Respond dataURL
Show modal contains document

Figure 3.12: Sequence diagram for Preview function

Description: After accessing the Database page, a request will be sent to the server to ask for getting the list of documents and showing it on the page. After that, the user clicks on the row of the document they want to read, so the ID of the document can be sent to the server. The server will seek all image data relate to the ID (image and stamp), then respond to the client, and render all information in the modal.

e) Edit

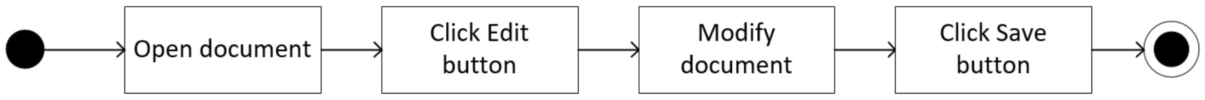


Figure 3.13: Activity diagram for Edit function

Description: The diagram figure above describes step by step how admin or staff can modify the document.

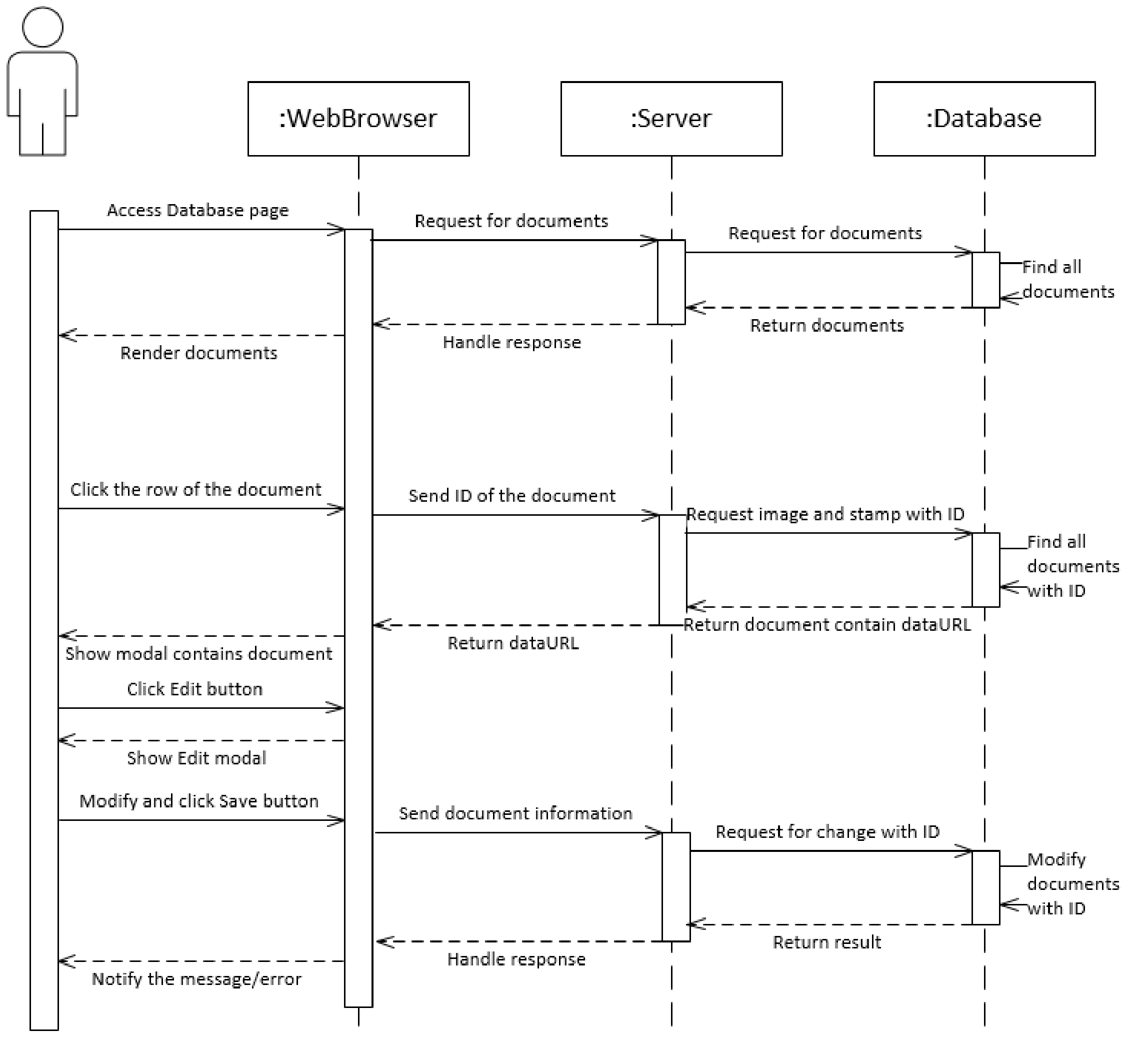


Figure 3.14: Sequence diagram for Edit function

Description: After doing the preview step. The user clicks on the Edit button and get into the edit modal, change the information that needed to modify, then click on the Save button, a request will be sent with the ID asking for changing the document in MongoDB.

f) Remove

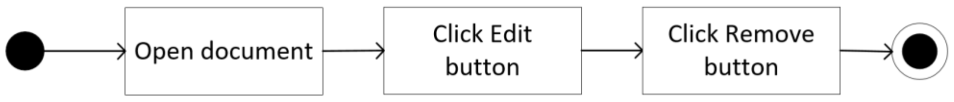


Figure 3.15: Activity diagram for Remove function

Description: The diagram figure above describes step by step how admin or staff can remove the document.

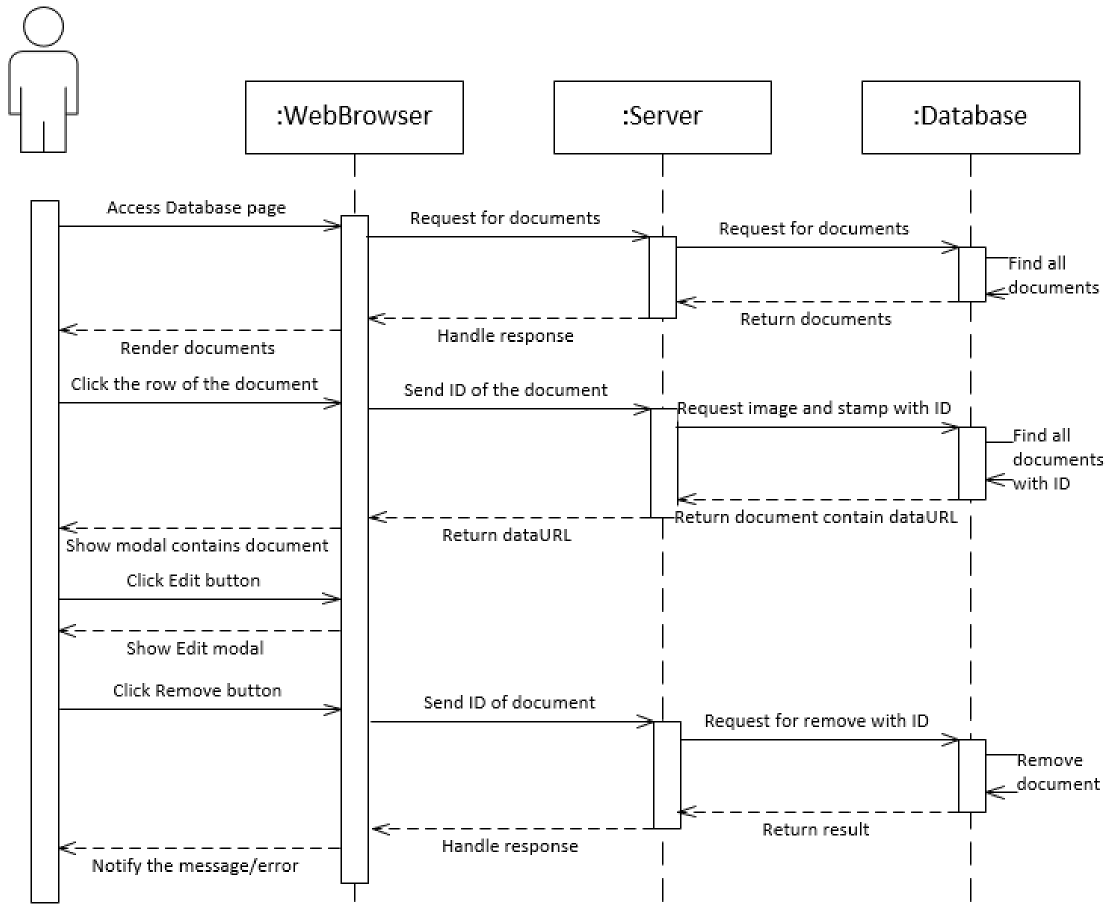


Figure 3.16: Sequence diagram for Remove function

Description: After doing the preview step. The user clicks the Edit button to get into the edit modal, which contains the Remove button, if the user clicks on it, a request will be sent with the ID asking for removing the document in MongoDB.

g) Export

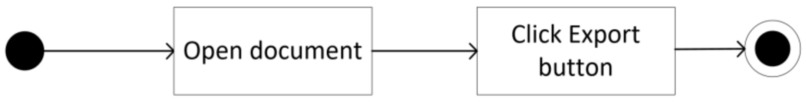


Figure 3.17: Activity diagram for Export function

Description: The diagram figure above describes step by step how the user can export the document.

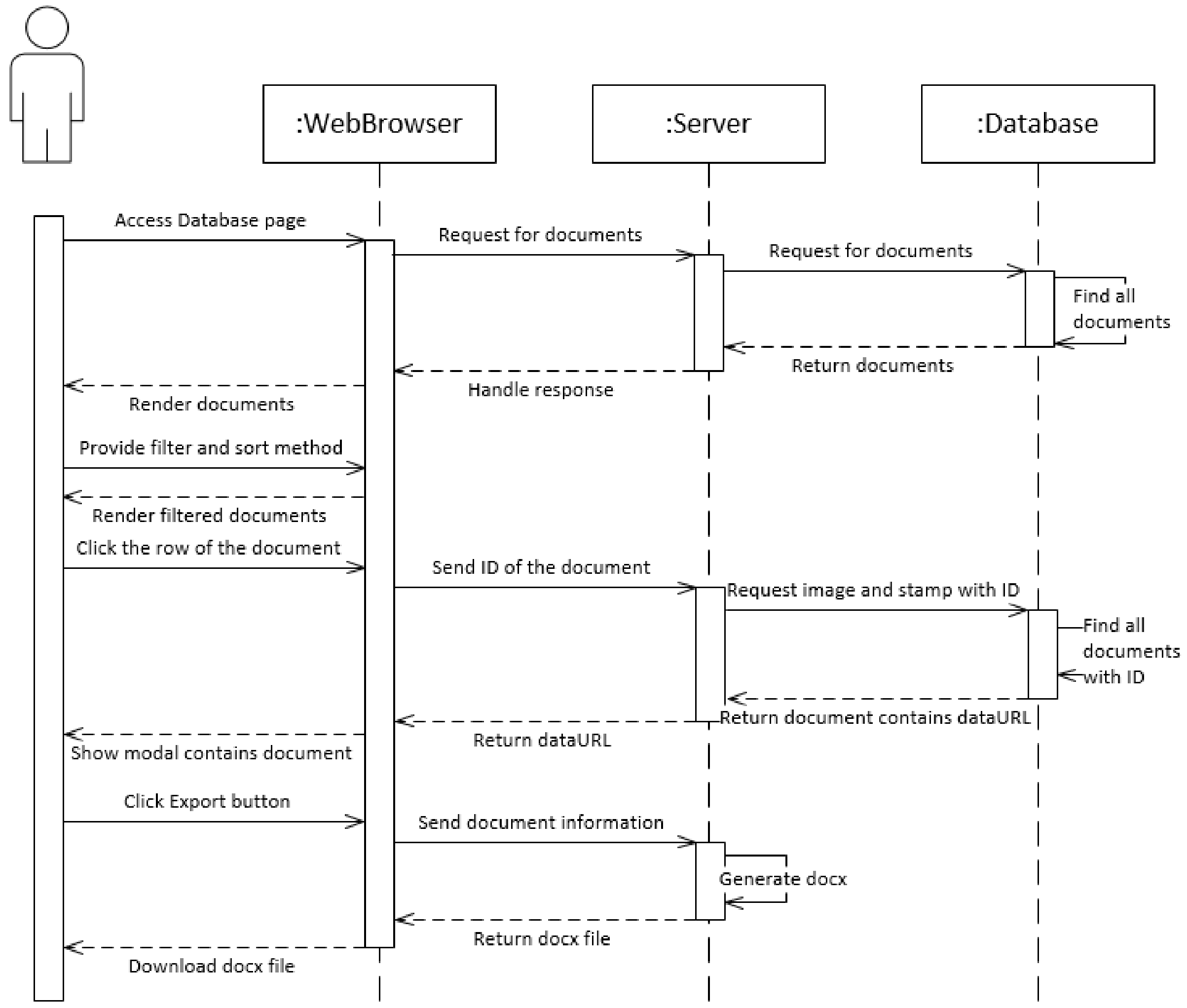


Figure 3.18: Sequence diagram for Export function

Description: After finding and opening the target document, the user can click on the Export button inside the modal, which can trigger the server to pack the document information and send back to the user under the DOC file.

h) Create User

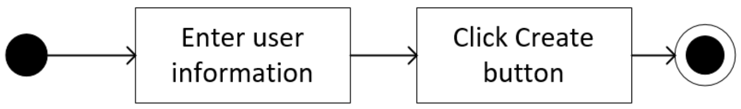


Figure 3.19: Activity diagram for Create User function

Description: The diagram figure above describes step by step how an admin can create a new user account.

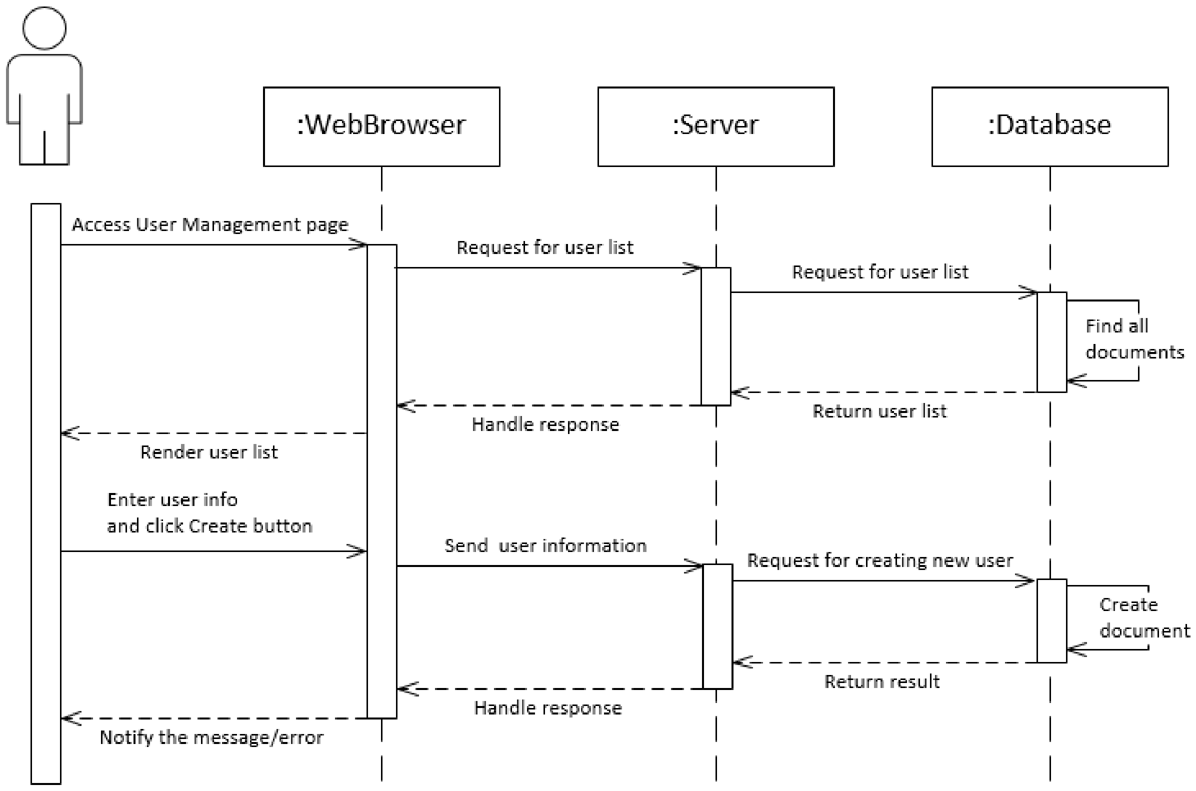


Figure 3.20: Sequence diagram for Create User function

Description: After accessing the User Management page, a request will be sent to the server to asks for getting the list of users and show it on the page. The admin can create a new user by providing username, password, and user permission, then click on the Create button to sent a request to the server asking to create a new user document in MongoDB.

i) Change User Password

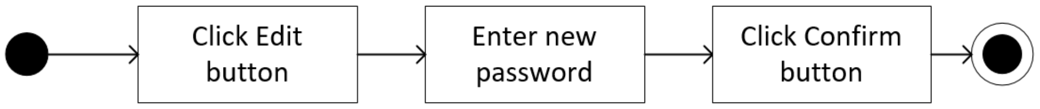


Figure 3.21: Activity diagram for Change User Password function

Description: The diagram figure above describes step by step how an admin can change user password.

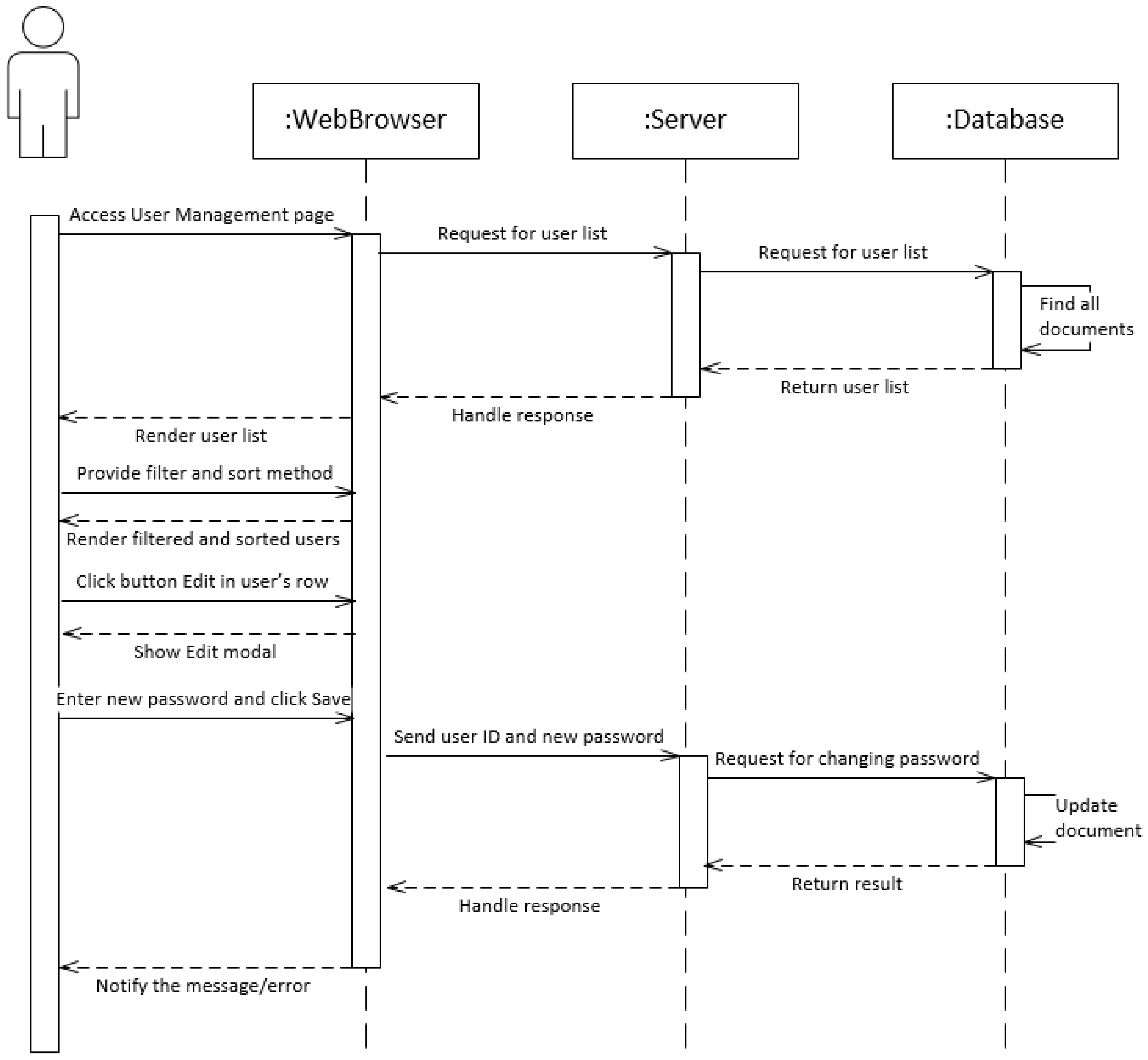


Figure 3.22: Sequence diagram for Change User Password function

Description: After accessing the User Management page, a request will be sent to the server to asks for getting the list of users and show it on the page. The admin can change the user's password by clicking the Edit button inside the user's row, typing a new password into the input attribute inside the modal and click Submit to ask the server for applying a new password to the user.

j) Remove User

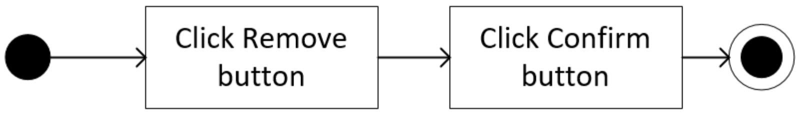


Figure 3.23: Activity diagram for Remove User function

Description: The diagram figure above describes step by step how an admin can remove a user account.

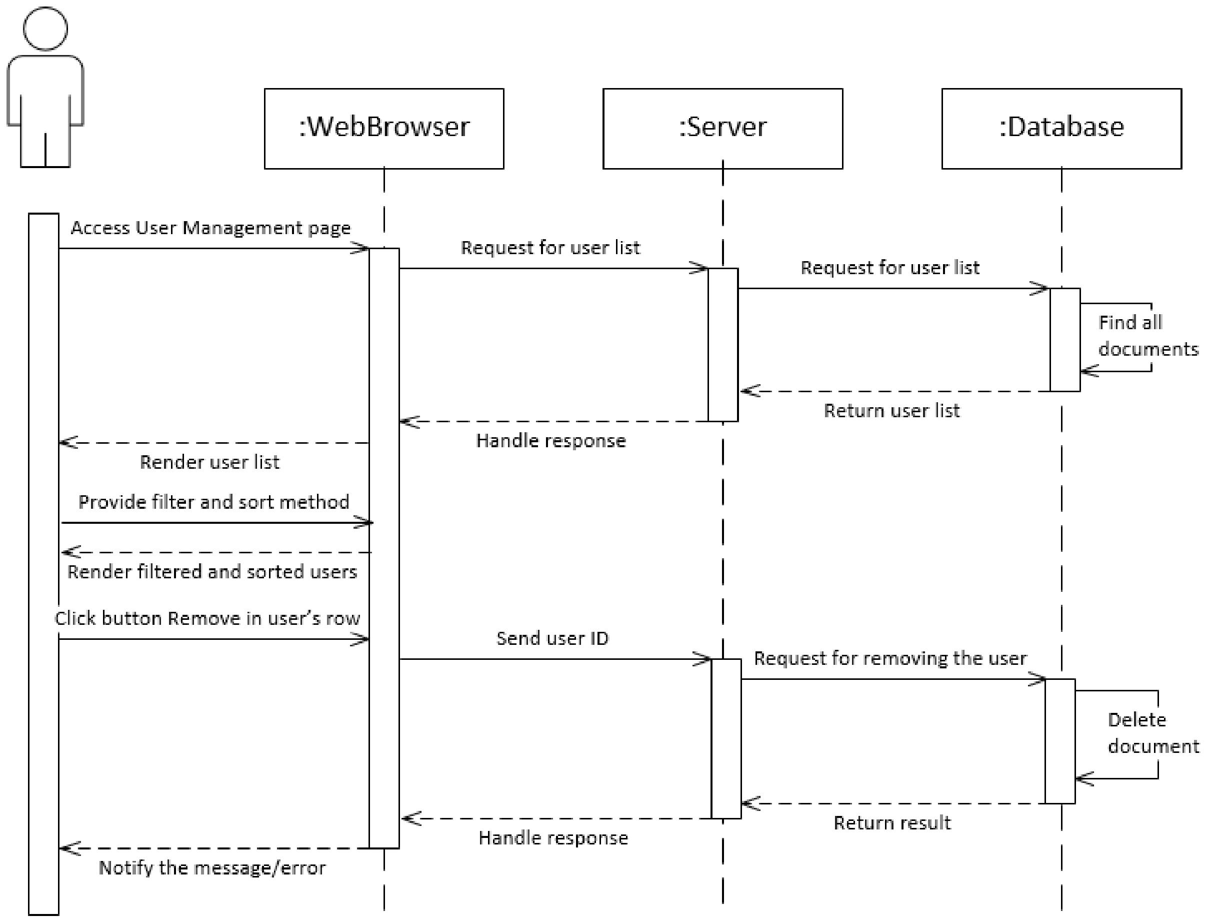


Figure 3.24: Sequence diagram for Remove User function

Description: After accessing the User Management page, a request will be sent to the server to ask for getting the list of users and showing it on the page. The admin removes the user by clicking the Remove button inside the user's row, a request will be sent to the server to remove the user out of the database.

3.6 System design

3.6.1 Database Schema

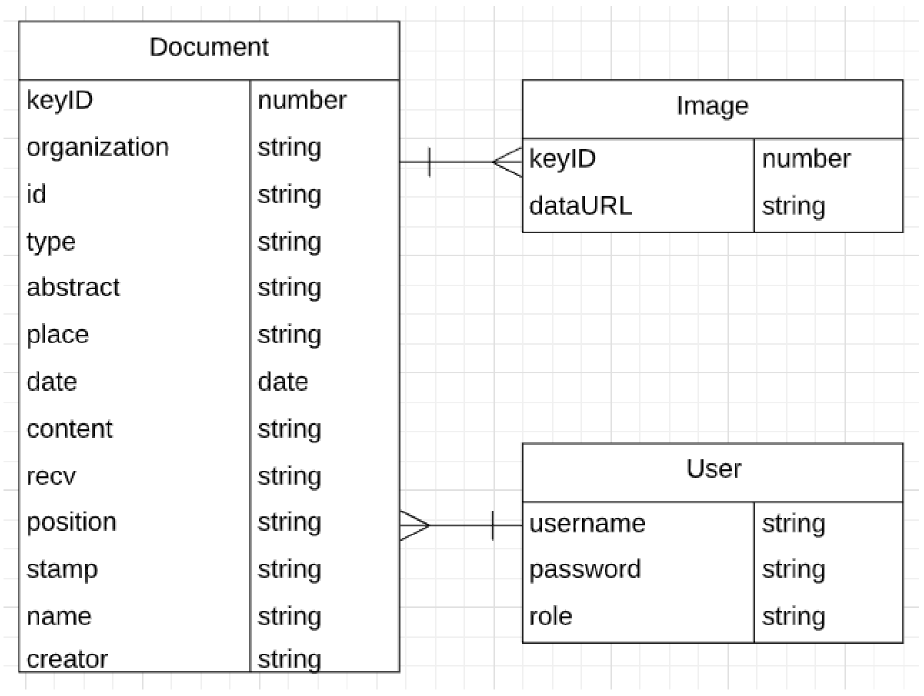


Figure 3.25: Database Schema

- **Document Schema:**
    - **keyID:** the ID of the document collection which is unique and required
    - **organization:** the name of the organization that writes the document
    - **id:** the id of the document
    - **type:** the type of the document
    - **abstract:** the abstract of the document
    - **place:** the place where the document was written
    - **date:** the time the document was written
    - **content:** the content of the document
    - **recv:** the recipient of the document
    - **position:** the position of the writer
    - **stamp:** the dataURL of the image of the stamp
    - **name:** the name of the writer
    - **creator:** the username of the account that create this document

- **Image Schema:**
  - **keyID:** the ID of the image collection which is unique and required
  - **dataURL:** the dataURL of original imaged documents
- **User Schema:**
  - **username:** the login name of the user which is unique and required
  - **password:** the login password of the user which is required
  - **role:** the role reflects the ability to exploit the features of the system which is required

3.6.2 Class Diagram

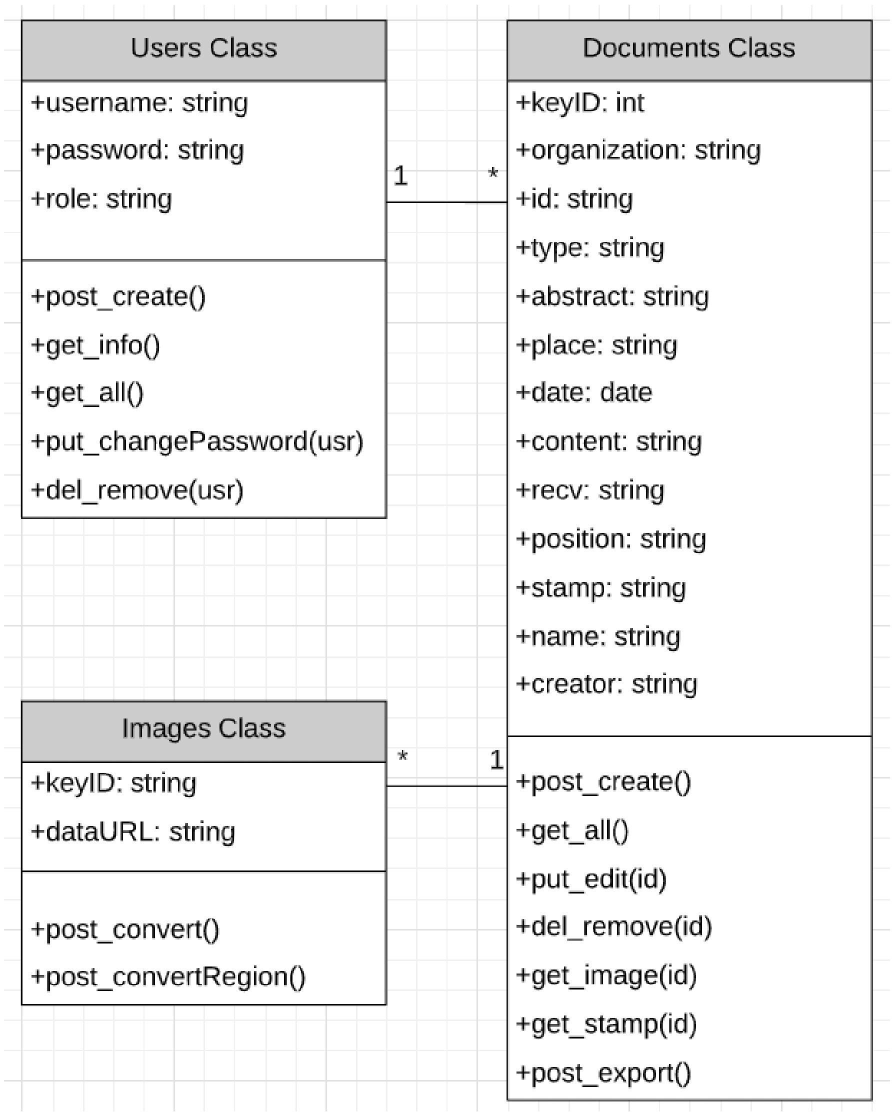


Figure 3.26: Class Diagram

- **Documents Class:**
    - **Attributes:** follow the attributes of Document Schema
    - **Method:**
        - **post_create():** create a new document in the database
        - **get_all():** return a JSON file contains a list of all documents in the database
        - **put_edit(id):** update the document which has keyID = id in the database
        - **del_remove(id):** remove the document which has keyID = id in the database
        - **get_image(id):** return dataURL of the original image of the document which has keyID = id in the database
        - **get_image(id):** return dataURL of stamp of the document which has keyID = id in the database
        - **post_export():** create and return the DOC file
- **Images Class:**
    - **Attributes:** follow the attributes of Image Schema
    - **Method:**
        - **post_convert():** receive an imaged document and extract document information using the method of section 3.2 and OCR engine
        - **post_convertRegion():** receive a piece of imaged document and convert it to text using OCR engine only
- **User Class:**
    - **Attributes:** follow the attributes of User Schema
    - **Method:**
        - **post_create():** create a new user account in the database
        - **get_info():** return username and role of the current login user in the database
        - **get_all():** return a JSON file contains a list of all account in the database

- **put_changePassword(usr):** update the new password of the user which has username = usr in the database
- **del_remove(usr):** remove the user which has username = usr in the database

3.6.3 MVC Model

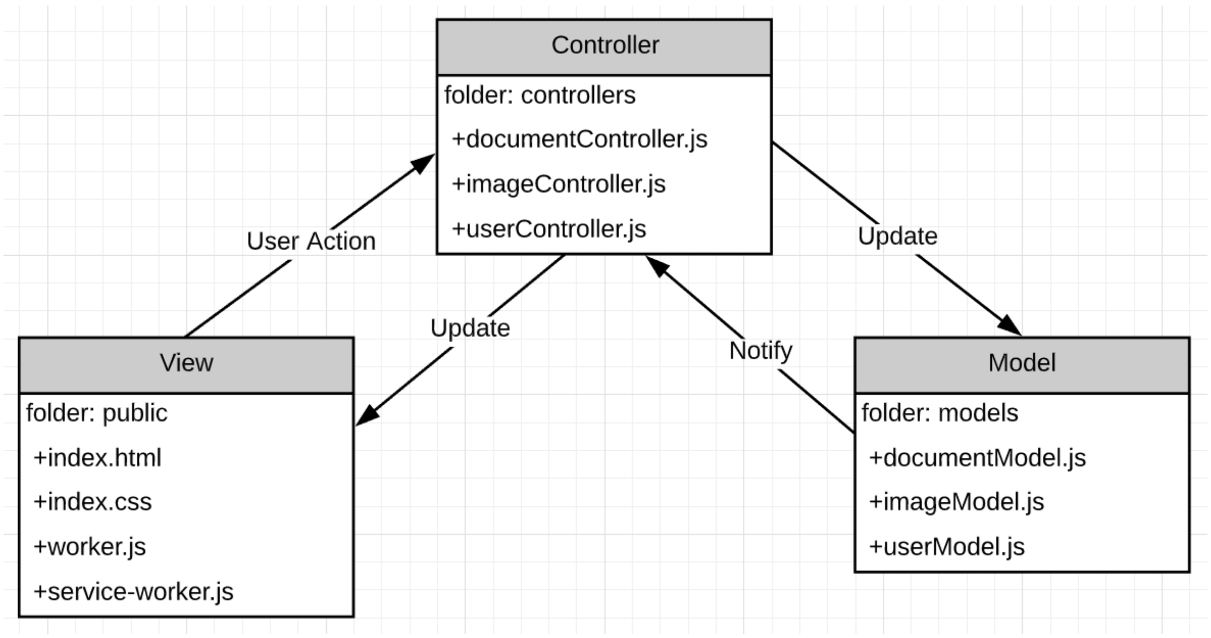


Figure 3.27: MVC Model

- View: ReactJS is used to create components which are rendered in the webpage
- Controller: NodeJS with Express Framework will control all the logic in the system and execute most of the features. Handling the request and response of servers to clients and otherwise
- Model: MongoDB is used to store and manage data. Returning the data whenever it gets a request from the server

3.6.4 Graphical user interfaces

The image below shows a draft user interface I designed for the program:

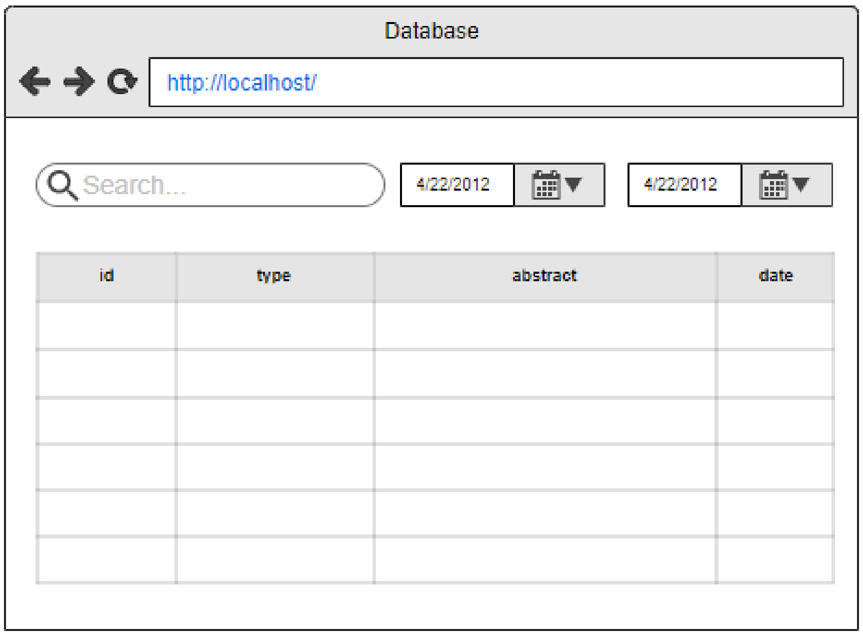


Figure 3.28: User Interface of the Database page

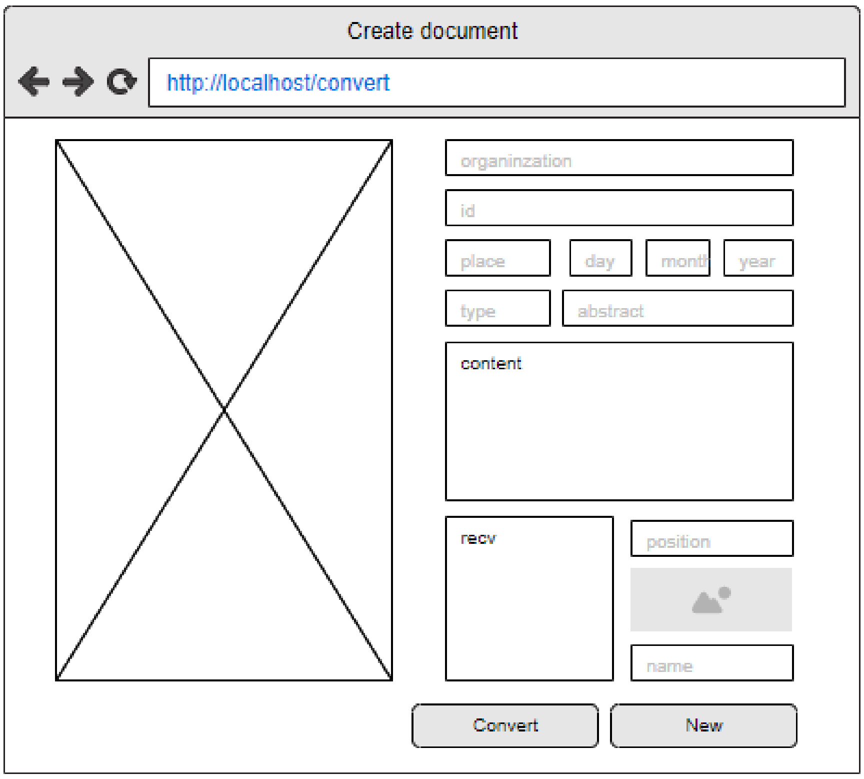


Figure 3.29: User Interface of the Create Document page

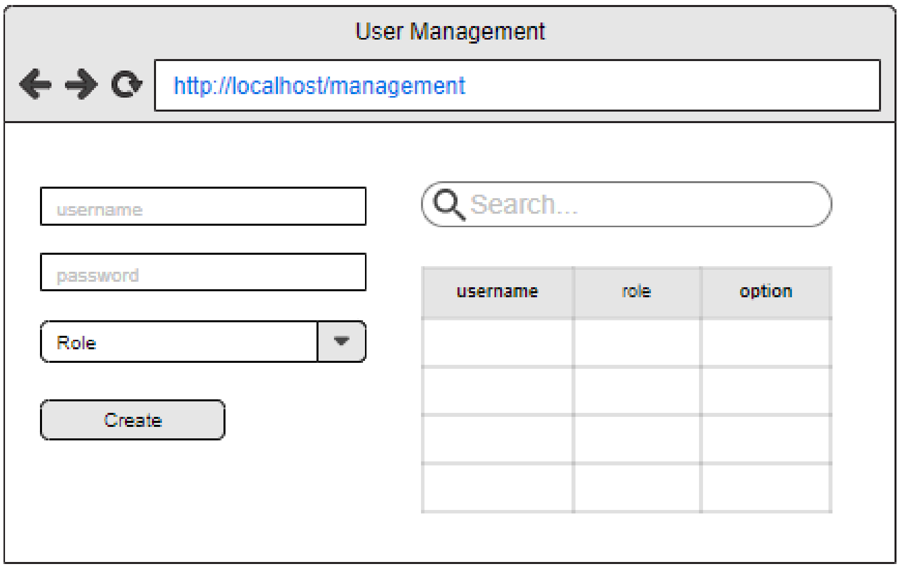


Figure 3.30: User Interface of the User Management page

These are the design of the three main pages. During the development process, we also need to create some support components such as modal and notification to make the interface clear and concise.

CHAPTER 4

IMPLEMENTATION AND RESULTS

4.1 Implementation

The system needs to have NodeJS installed. Then, we arrange folders and files following this structure:

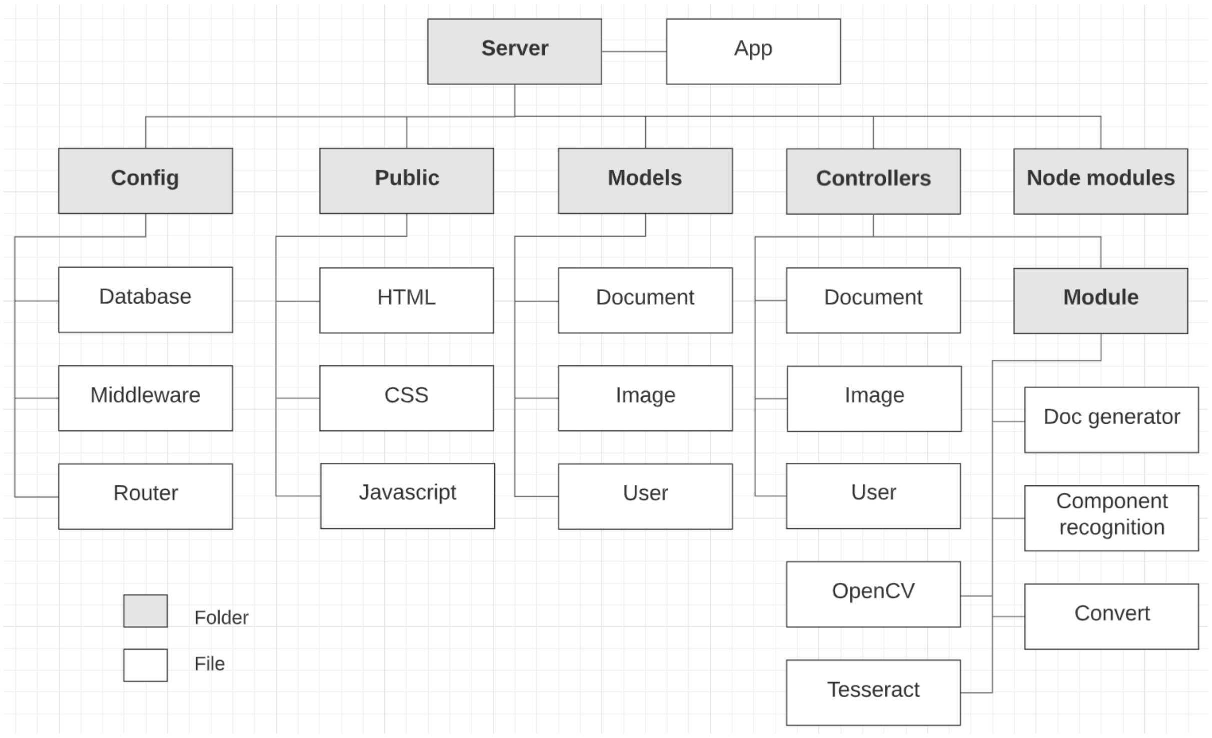
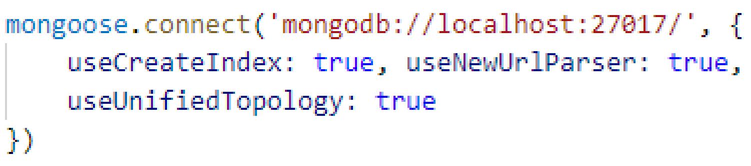


Figure 4.1: Project structure

Description:

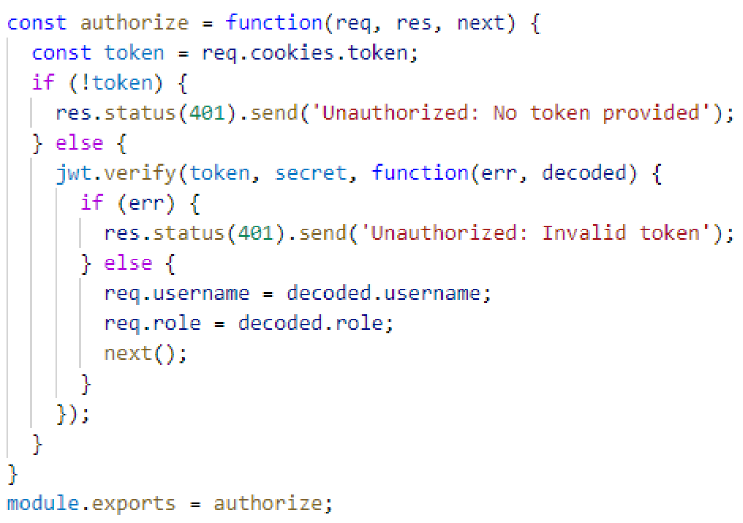
- **Config:** The folder contains files that are used to configure the server
  - Database: This is a javascript file that contains code to connect the Node server with the mongoDB database. In this case, the server has already installed in the localhost



```
mongoose.connect('mongodb://localhost:27017/', {
    useCreateIndex: true, useNewUrlParser: true,
    useUnifiedTopology: true
})
```

Figure 4.2: Connecting to server

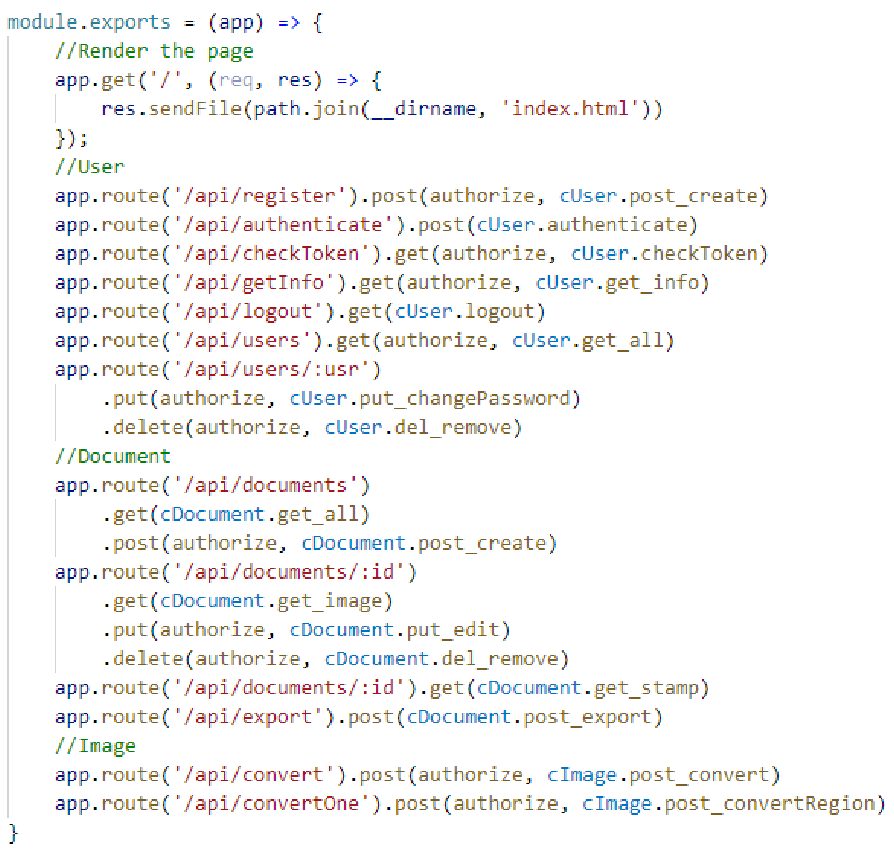
- Middleware: This javascript file is used to protect selected routes, and ensure that a user is authenticated (login) before allowing their requests to go through



```
const authorize = function(req, res, next) {
  const token = req.cookies.token;
  if (!token) {
    res.status(401).send('Unauthorized: No token provided');
  } else {
    jwt.verify(token, secret, function(err, decoded) {
      if (err) {
        res.status(401).send('Unauthorized: Invalid token');
      } else {
        req.username = decoded.username;
        req.role = decoded.role;
        next();
      }
    });
  }
}
module.exports = authorize;
```

Figure 4.3: Checking the authentication

- Routes: have main function to navigate to a specific Controller which follows a request from the user



```
module.exports = (app) => {
    //Render the page
    app.get('/', (req, res) => {
        res.sendFile(path.join(__dirname, 'index.html'))
    });
    //User
    app.route('/api/register').post(authorize, cUser.post_create)
    app.route('/api/authenticate').post(cUser.authenticate)
    app.route('/api/checkToken').get(authorize, cUser.checkToken)
    app.route('/api/getInfo').get(authorize, cUser.get_info)
    app.route('/api/logout').get(cUser.logout)
    app.route('/api/users').get(authorize, cUser.get_all)
    app.route('/api/users/:usr')
        .put(authorize, cUser.put_changePassword)
        .delete(authorize, cUser.del_remove)
    //Document
    app.route('/api/documents')
        .get(cDocument.get_all)
        .post(authorize, cDocument.post_create)
    app.route('/api/documents/:id')
        .get(cDocument.get_image)
        .put(authorize, cDocument.put_edit)
        .delete(authorize, cDocument.del_remove)
    app.route('/api/documents/:id').get(cDocument.get_stamp)
    app.route('/api/export').post(cDocument.post_export)
    //Image
    app.route('/api/convert').post(authorize, cImage.post_convert)
    app.route('/api/convertOne').post(authorize, cImage.post_convertRegion)
}
```

Figure 4.4: Handling the RESTful API

- **Public:** The folder contains HTML, CSS, and Javascript files that were generated using ReactJS (The result from the front-end development)
- **Models:** The folder contains javascript files that are used to create mongoDB schemas which define the shape and content in the document collection, image collection, and user collection following section 3.6.1
- **Controllers:** The folder contains javascript files that have functions to handle requests from the client (such as handling the conversion request, create new user request, edit document request…). These functions will be called inside the Router file
- **Module:** This folder contains Tesseract trained data, OpenCV.js, and these modules:
  - **components-recognition.js:** the file has javascript code that runs the method from section 3.2
  - **convert.js:** the file has javascript code to configurate TesseractOCR engine to extract text from image.
  - **doc-generator.js:** the file has javascript code to create DOC file using DocxJS library
- **Node module:** This folder contains all the libraries needed for the server to work. In the first installation, the user needs to type the command "npm install" in the command-line interface to fully update and install the support libraries into this folder
- **App**: The run file of the server. This is where all the main function executes all the direction of the server

a) Module component-recognition.js:

In this module, we rewrite the psuedo-code from the method that has been shown in section 3.2 into javascript language that can run on the server:

```
const cv = require(‘opencv.js’)
const fs = require(‘fs’)
let oImage = cv.imread(IMAGE_PATH)
                  //Load the image file uploaded by the client
oImage = cv.resize(oImage, (WIDTH, HEIGHT))  //Fix original image size
```

```
    let image = cv.cvtColor(oImage, cv.RGBA2GRAY)   //RGBA to Grayscale

    image = cv.threshold(image, cv.THRESH_BINARY_INV)
                     //Grayscale to black and white (binary) only

    image = cv.dilate(image, (cv.MORPH_RECT, (15, 10)))
                     //Dilate the text with rectangle-shaped(15x10) kernel

    let contours = cv.findContours(image)

    let i = 0

    let ERR = 50            //Tolerance = ±50 pixel

    let listOfFoundElems = []

    let ElemsProp = [['organization',x1,y1,w1,h1], ['id',x2,y2,w2,h2]...]
        //x,y,w,h base on component's position and size

    while i < contours:

        let rect = cv.boundingRect(contours[i]) //Get bounding rectangle

        for elem in ElemsProp:

                if(elem[1]-ERR < rect.x < elem[1]+ERR &&
                   elem[2]-ERR < rect.y < elem[2]+ERR &&
                  elem[3]-ERR < rect.width < elem[3]+ERR &&
                  elem[4]-ERR < rect.height < elem[4]+ERR):

                      fs.writeFile(elem[0]+'.png', oImage.roi(rect.x, rect.y,
                      rect.width, rect.height))

                      listOfFoundElems.push(e[0])

                      break

        i += 1

    if fs.existsSync('stamp.png'):

        let oStamp = cv.imread('stamp.png')

        oStamp = cv.inRange(oStamp, (170,255,255,255), (255,255,255,255))
                                  //Get R in RGB color only with 170<R<255

        let stamp = cv.cvtColor(oStamp, cv.RGBA2GRAY)

        stamp = cv.GaussianBlur(stamp)

        let minR = 70, maxR = 200 //Max, min radius of the circle stamp

        let circle = cv.HoughCircles(stamp,cv.HOUGH_GRADIANT,minR,maxR)

        let mask = cv.circle(circle.center,circle.radius,(255,255,255,255))

        mask = cv.bitwise_not(cv.threshold(mask, cv.THRESH_BINARY_INV))

        stamp = cv.bitwise_and(oStamp, stamp, mask)

        fs.writeFile('stamp.png', stamp.roi(circle.x,circle.y, circle.radius*2,
        circle.radius*2)

return listOfFoundElems
```

b) **Module convert.js:**

In this module, we will use Tesseract to extract text from each component's images of the document that we have got from the image process above.

For each individual component, we will use different page segmentation methods (PSM) that Tesseract provides [13]. For example, we will apply PSM 3 (Fully automatic page segmentation) in the configuration to get a text from the Content part, but using PSM 7 (Treat image as a single text line) or PSM 11(Find as much text as possible in no particular order) in the configuration to get a text from the ID part. This will help Tesseract to return a better result.

After converting the images, we will store all of the string in a dictionary type that is easier to manage.

```
const tess = require('node-tesseract-ocr')
let config = { lang: 'vie', oem: 1, psm: 3 }
let texts = {}
for elemName in listOfFoundElems
      texts[elemName] = tess.recognize(element.png, config)
   return texts
```

c) **Module doc-generator.js:**

```
const {Packer,Document,Table,Col,Row,Cell,TextRun} = require('docx')
let document = new Document()
let firstTable = new Table()
let lastTable = new Table()
let firstRow = new Row()
let secondRow = new Row() Let lastRow = new Row()
firstRow.push(new Cell(new
TextRun(texts['Organization'])))
firstRow.push(new Cell(new TextRun('CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT
NAM'),  new TextRun('Độc lập – Tự do – Hạnh phúc'))) secondRow.push(new
Cell(new TextRun(texts['id'])))
secondRow.push(new Cell(new TextRun(texts['place'] + ', ngày ' +
texts['day'] + ' tháng ' + texts['month'] + ' năm ' + texts['year'])))
lastRow.push(new Cell(new TextRun(texts['recv'])))
lastRow.push(new Cell(new TextRun(texts[‘position’]), new
TextRun(‘(Đã ký)’), new TextRun(texts[‘name’])))
firstTable.push(firstRow, secondRow)  lastTable.push(lastRow)
document.push(firstTable, new TextRun(texts[‘type’]), new
TextRun(texts[‘abstract’]), new TextRun(texts[‘content’]), lastTable)
let file = Packer.toBase64String(document)
return file
```

The real code of this module is also needs some child classes to arrange and set the style for text such as WidthType, BorderStyle, WidthType, Paragraph… But the concept is quite simple as you can see above. This function will be used to describe the system of how to create the docx file with the right document form.

4.2 Results

Login component

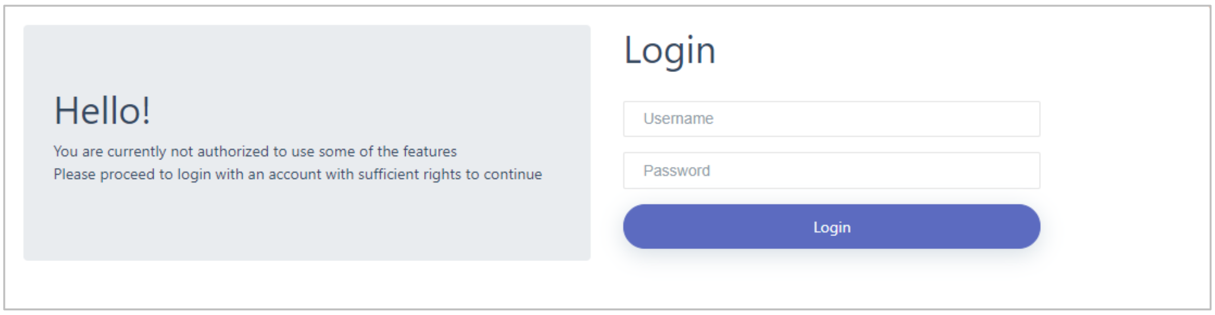

Hello!
You are currently not authorized to use some of the features
Please proceed to login with an account with sufficient rights to continue
Login
Username
Password
Login

Figure 4.5: Login component

Create New Document component

The image below is the user interface when the user runs the application with the browser. In the initial state, we have a blank text area with no imaged document.

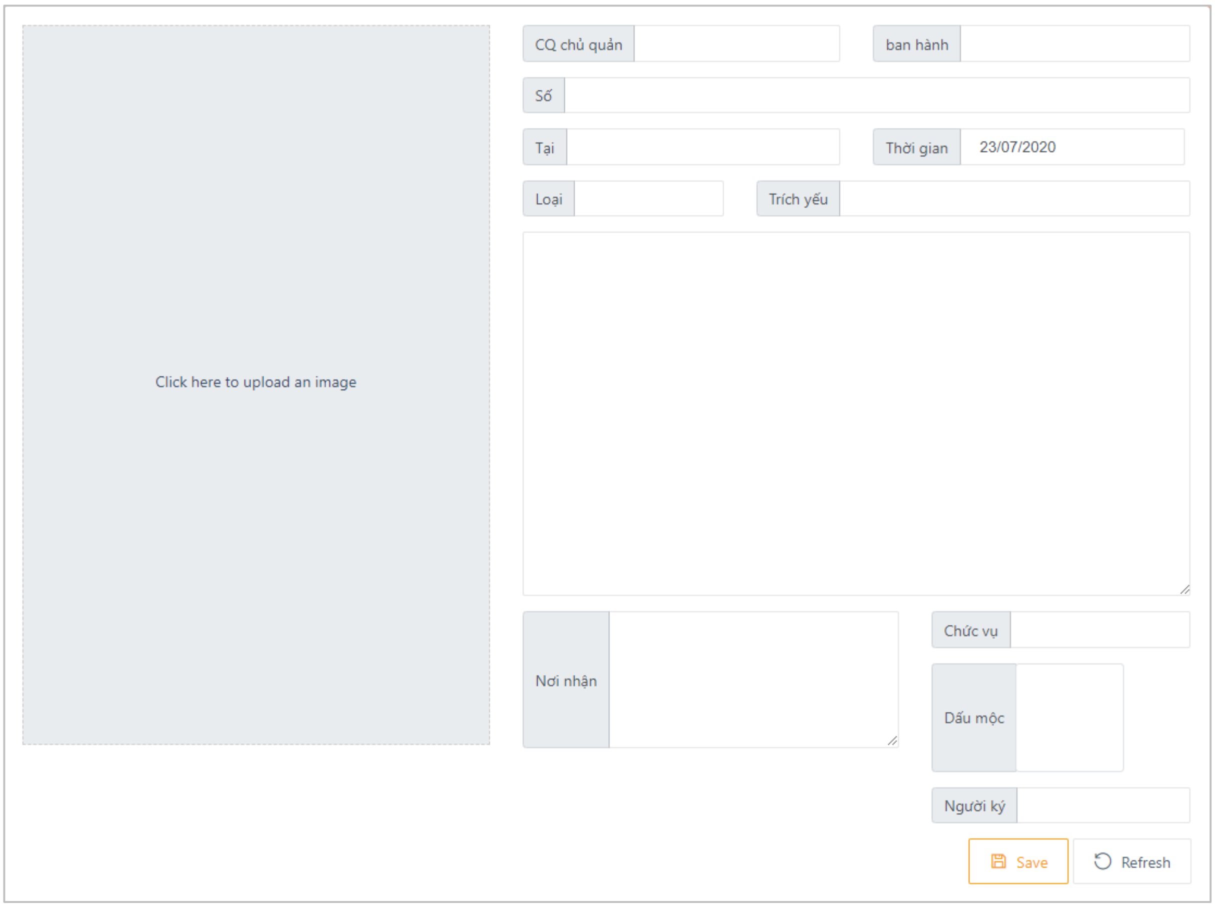
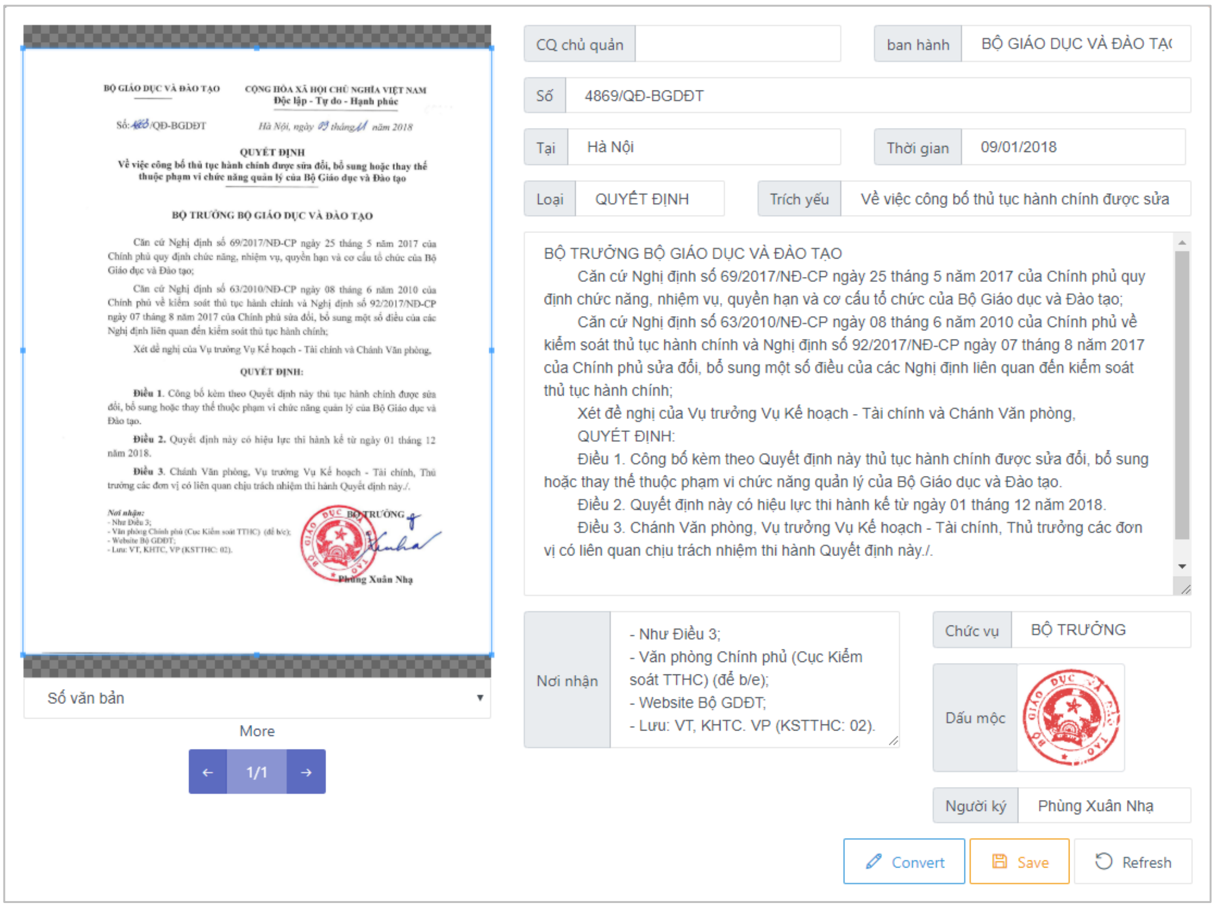


Figure 4.6: Create New Document component

After the conversion process:



BỘ GIÁO DỤC VÀ ĐÀO TẠO

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Số: 4869/QĐ-BGDĐT

Hà Nội, ngày 09 tháng 01 năm 2018

QUYẾT ĐỊNH
Về việc công bố thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo

BỘ TRƯỞNG BỘ GIÁO DỤC VÀ ĐÀO TẠO

Căn cứ Nghị định số 69/2017/NĐ-CP ngày 25 tháng 5 năm 2017 của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Giáo dục và Đào tạo;

Căn cứ Nghị định số 63/2010/NĐ-CP ngày 08 tháng 6 năm 2010 của Chính phủ về kiểm soát thủ tục hành chính và Nghị định số 92/2017/NĐ-CP ngày 07 tháng 8 năm 2017 của Chính phủ sửa đổi, bổ sung một số điều của các Nghị định liên quan đến kiểm soát thủ tục hành chính;

Xét đề nghị của Vụ trưởng Vụ Kế hoạch - Tài chính và Chánh Văn phòng,

QUYẾT ĐỊNH:

Điều 1. Công bố kèm theo Quyết định này thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo.

Điều 2. Quyết định này có hiệu lực thi hành kể từ ngày 01 tháng 12 năm 2018.

Điều 3. Chánh Văn phòng, Vụ trưởng Vụ Kế hoạch - Tài chính, Thủ trưởng các đơn vị có liên quan chịu trách nhiệm thi hành Quyết định này./.

Nơi nhận:
- Như Điều 3;
- Văn phòng Chính phủ (Cục Kiểm soát TTHC) (để b/c);
- Website Bộ GDĐT;
- Lưu: VT, KHTC, VP (KSTTHC: 02).

BỘ TRƯỞNG

[illegible]

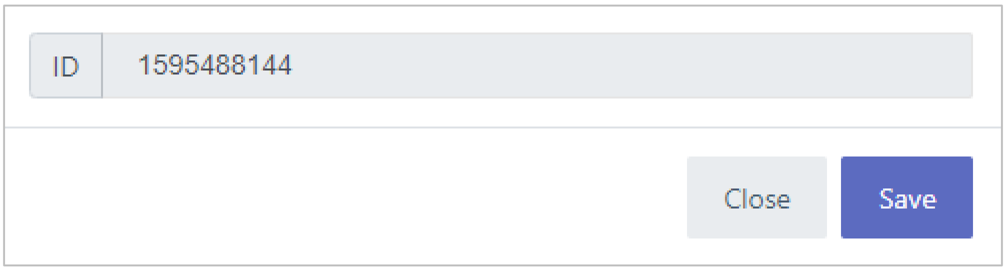
Phùng Xuân Nhạ

CQ chủ quản	
ban hành	BỘ GIÁO DỤC VÀ ĐÀO TẠ(
Số	4869/QĐ-BGDĐT
Tại	Hà Nội
Thời gian	09/01/2018
Loại	QUYẾT ĐỊNH
Trích yếu	Về việc công bố thủ tục hành chính được sửa
Nơi nhận	- Như Điều 3; - Văn phòng Chính phủ (Cục Kiểm soát TTHC) (để b/e); - Website Bộ GDĐT; - Lưu: VT, KHTC. VP (KSTTHC: 02).
Chức vụ	BỘ TRƯỞNG
Dấu mộc	
Người ký	Phùng Xuân Nhạ

BỘ TRƯỞNG BỘ GIÁO DỤC VÀ ĐÀO TẠO
Căn cứ Nghị định số 69/2017/NĐ-CP ngày 25 tháng 5 năm 2017 của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Giáo dục và Đào tạo;
Căn cứ Nghị định số 63/2010/NĐ-CP ngày 08 tháng 6 năm 2010 của Chính phủ về kiểm soát thủ tục hành chính và Nghị định số 92/2017/NĐ-CP ngày 07 tháng 8 năm 2017 của Chính phủ sửa đổi, bổ sung một số điều của các Nghị định liên quan đến kiểm soát thủ tục hành chính;
Xét đề nghị của Vụ trưởng Vụ Kế hoạch - Tài chính và Chánh Văn phòng,
QUYẾT ĐỊNH:
Điều 1. Công bố kèm theo Quyết định này thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo.
Điều 2. Quyết định này có hiệu lực thi hành kể từ ngày 01 tháng 12 năm 2018.
Điều 3. Chánh Văn phòng, Vụ trưởng Vụ Kế hoạch - Tài chính, Thủ trưởng các đơn vị có liên quan chịu trách nhiệm thi hành Quyết định này./.

Figure 4.7: Create New Document component after the Conversion process

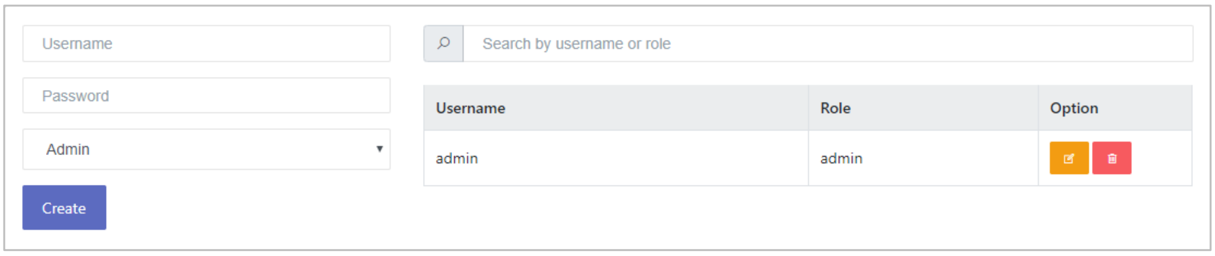
Confirm after clicking the saving button:



ID 1595488144

Figure 4.8: Document saving confirm dialog

User management component

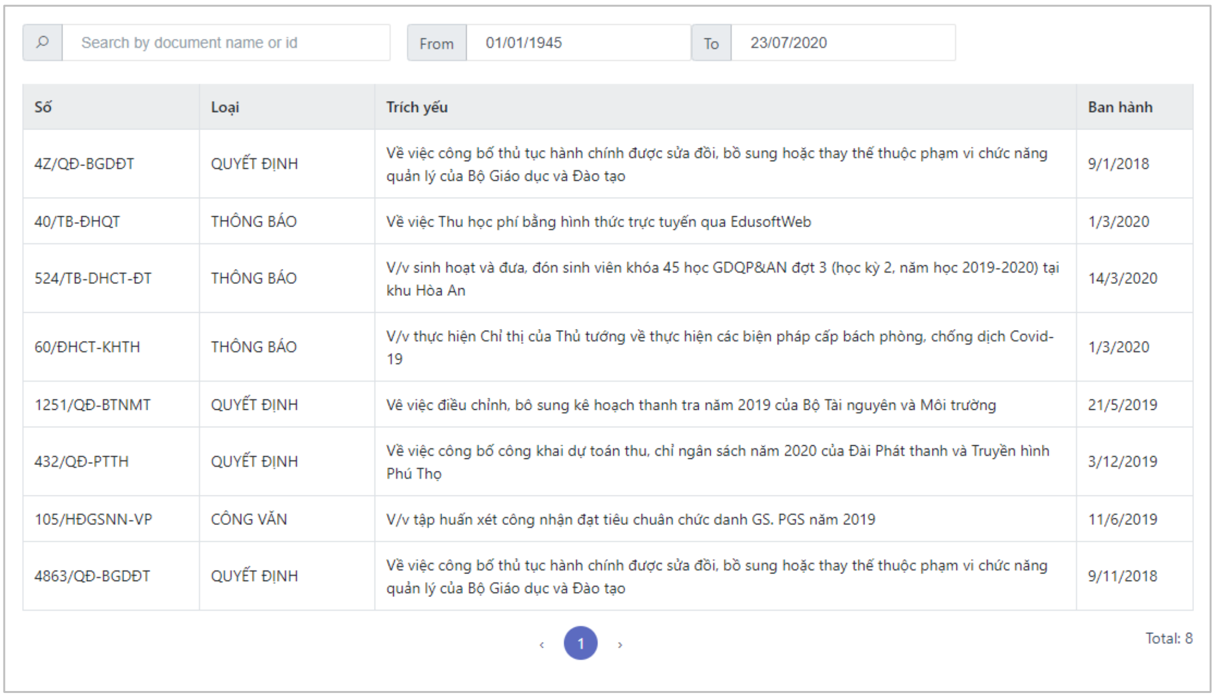


Username	Role	Option
admin	admin	

Figure 4.9: User Management component

In this component, the user can create a new account by filling the account's information in the form on the left side and submit it. Otherwise, the user can change the password, or remove the account by clicking on the option buttons on the right side.

Database component

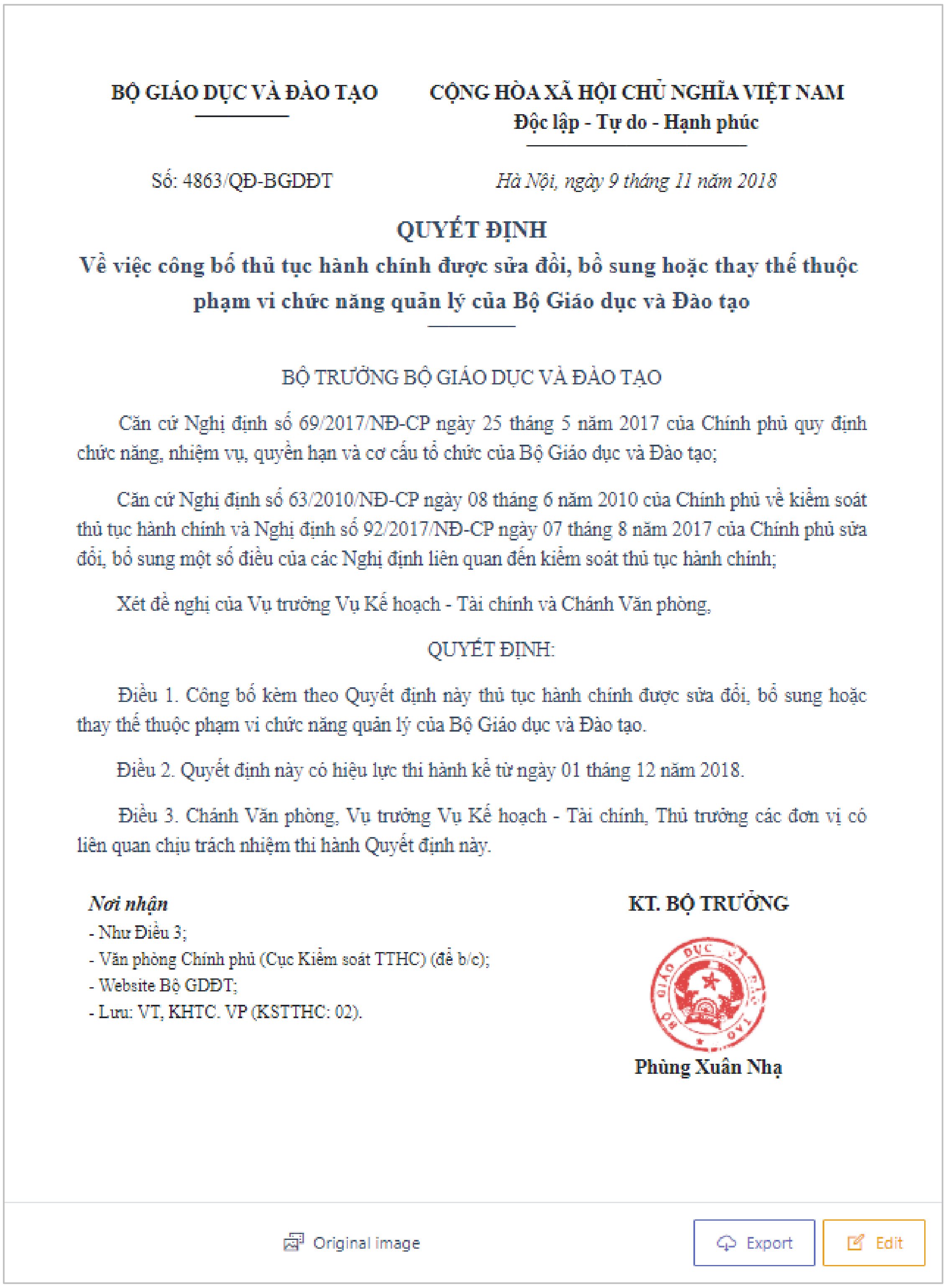


Số	Loại	Trích yếu	Ban hành
4Z/QĐ-BGDĐT	QUYẾT ĐỊNH	Về việc công bố thủ tục hành chính được sửa đồi, bồ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo	9/1/2018
40/TB-ĐHQT	THÔNG BÁO	Về việc Thu học phí bằng hình thức trực tuyến qua EdusoftWeb	1/3/2020
524/TB-DHCT-ĐT	THÔNG BÁO	V/v sinh hoạt và đưa, đón sinh viên khóa 45 học GDQP&AN đợt 3 (học kỳ 2, năm học 2019-2020) tại khu Hòa An	14/3/2020
60/ĐHCT-KHTH	THÔNG BÁO	V/v thực hiện Chỉ thị của Thủ tướng về thực hiện các biện pháp cấp bách phòng, chống dịch Covid-19	1/3/2020
1251/QĐ-BTNMT	QUYẾT ĐỊNH	Vê việc điều chỉnh, bô sung kê hoạch thanh tra năm 2019 của Bộ Tài nguyên và Môi trường	21/5/2019
432/QĐ-PTTH	QUYẾT ĐỊNH	Về việc công bố công khai dự toán thu, chỉ ngân sách năm 2020 của Đài Phát thanh và Truyền hình Phú Thọ	3/12/2019
105/HĐGSNN-VP	CÔNG VĂN	V/v tập huấn xét công nhận đạt tiêu chuân chức danh GS. PGS năm 2019	11/6/2019
4863/QĐ-BGDĐT	QUYẾT ĐỊNH	Về việc công bố thủ tục hành chính được sửa đồi, bồ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo	9/11/2018

Figure 4.10: Database component

This component has some input to help the user find the document. By clicking on the row of the document we want to read, a modal will popup as the figure below:

Preview document modal


BỘ GIÁO DỤC VÀ ĐÀO TẠO

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Số: 4863/QĐ-BGDĐT

Hà Nội, ngày 9 tháng 11 năm 2018

QUYẾT ĐỊNH
Về việc công bố thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo

BỘ TRƯỞNG BỘ GIÁO DỤC VÀ ĐÀO TẠO

Căn cứ Nghị định số 69/2017/NĐ-CP ngày 25 tháng 5 năm 2017 của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Giáo dục và Đào tạo;

Căn cứ Nghị định số 63/2010/NĐ-CP ngày 08 tháng 6 năm 2010 của Chính phủ về kiểm soát thủ tục hành chính và Nghị định số 92/2017/NĐ-CP ngày 07 tháng 8 năm 2017 của Chính phủ sửa đổi, bổ sung một số điều của các Nghị định liên quan đến kiểm soát thủ tục hành chính;

Xét đề nghị của Vụ trưởng Vụ Kế hoạch - Tài chính và Chánh Văn phòng,

QUYẾT ĐỊNH:

Điều 1. Công bố kèm theo Quyết định này thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo.

Điều 2. Quyết định này có hiệu lực thi hành kể từ ngày 01 tháng 12 năm 2018.

Điều 3. Chánh Văn phòng, Vụ trưởng Vụ Kế hoạch - Tài chính, Thủ trưởng các đơn vị có liên quan chịu trách nhiệm thi hành Quyết định này.

Nơi nhận
- Như Điều 3;
- Văn phòng Chính phủ (Cục Kiểm soát TTHC) (để b/c);
- Website Bộ GDĐT;
- Lưu: VT, KHTC. VP (KSTTHC: 02).

KT. BỘ TRƯỞNG

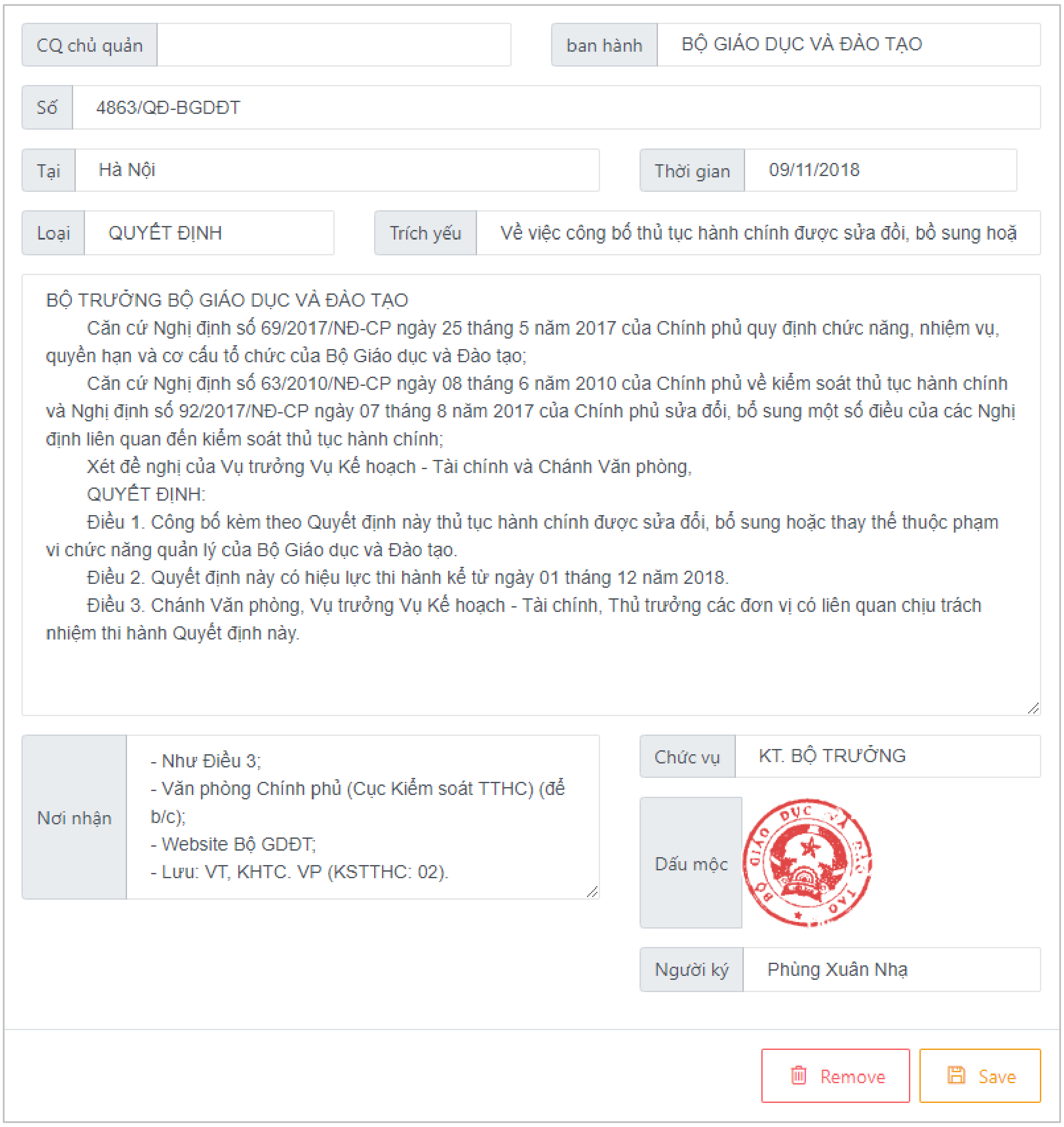
BỘ GIÁO DỤC VÀ ĐÀO TẠO

Phùng Xuân Nhạ

Figure 4.11: Preview modal

In this modal, the user has some buttons which can help them to see the original image of the document, export the document, or edit the document.

Edit Document modal



CQ chủ quản	
ban hành	BỘ GIÁO DỤC VÀ ĐÀO TẠO
Số	4863/QĐ-BGDĐT
Tại	Hà Nội
Thời gian	09/11/2018
Loại	QUYẾT ĐỊNH
Trích yếu	Về việc công bố thủ tục hành chính được sửa đổi, bổ sung hoặ

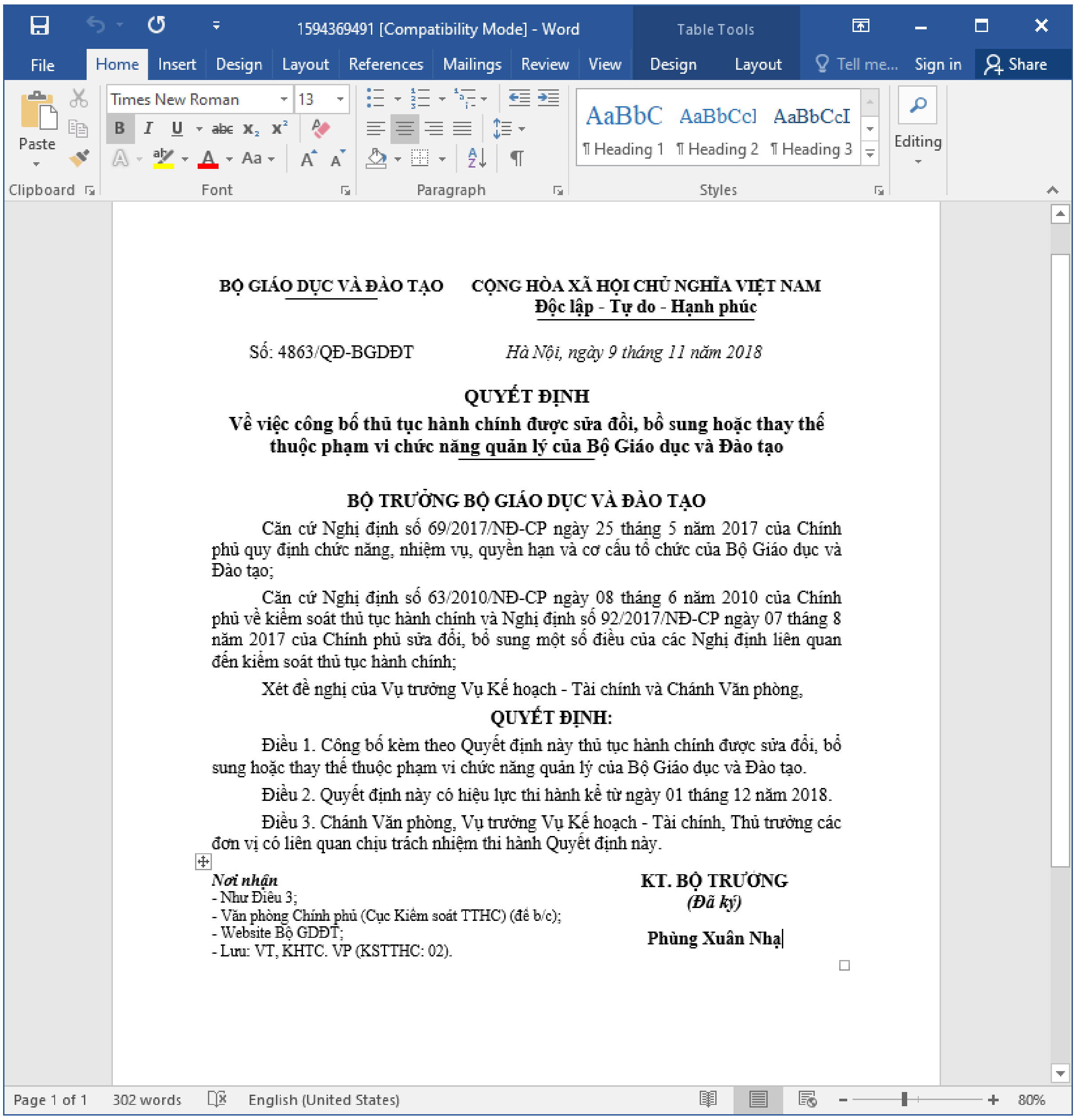
BỘ TRƯỞNG BỘ GIÁO DỤC VÀ ĐÀO TẠO
Căn cứ Nghị định số 69/2017/NĐ-CP ngày 25 tháng 5 năm 2017 của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Giáo dục và Đào tạo;
Căn cứ Nghị định số 63/2010/NĐ-CP ngày 08 tháng 6 năm 2010 của Chính phủ về kiểm soát thủ tục hành chính và Nghị định số 92/2017/NĐ-CP ngày 07 tháng 8 năm 2017 của Chính phủ sửa đổi, bổ sung một số điều của các Nghị định liên quan đến kiểm soát thủ tục hành chính;
Xét đề nghị của Vụ trưởng Vụ Kế hoạch - Tài chính và Chánh Văn phòng,
QUYẾT ĐỊNH:
Điều 1. Công bố kèm theo Quyết định này thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo.
Điều 2. Quyết định này có hiệu lực thi hành kể từ ngày 01 tháng 12 năm 2018.
Điều 3. Chánh Văn phòng, Vụ trưởng Vụ Kế hoạch - Tài chính, Thủ trưởng các đơn vị có liên quan chịu trách nhiệm thi hành Quyết định này.

Nơi nhận	- Như Điều 3; - Văn phòng Chính phủ (Cục Kiểm soát TTHC) (để b/c); - Website Bộ GDĐT; - Lưu: VT, KHTC. VP (KSTTHC: 02).
Chức vụ	KT. BỘ TRƯỞNG
Dấu mộc	BỘ GIÁO DỤC VÀ ĐÀO TẠO
Người ký	Phùng Xuân Nhạ

Figure 4.12: Edit modal

In this modal, the user can modify and click the Save button or remove it from the database by clicking on the Remove button.

An exported file is displayed on Microsoft Word



BỘ GIÁO DỤC VÀ ĐÀO TẠO **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**
Độc lập - Tự do - Hạnh phúc

Số: 4863/QĐ-BGDĐT *Hà Nội, ngày 9 tháng 11 năm 2018*

QUYẾT ĐỊNH
Về việc công bố thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo

BỘ TRƯỞNG BỘ GIÁO DỤC VÀ ĐÀO TẠO

Căn cứ Nghị định số 69/2017/NĐ-CP ngày 25 tháng 5 năm 2017 của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Giáo dục và Đào tạo;

Căn cứ Nghị định số 63/2010/NĐ-CP ngày 08 tháng 6 năm 2010 của Chính phủ về kiểm soát thủ tục hành chính và Nghị định số 92/2017/NĐ-CP ngày 07 tháng 8 năm 2017 của Chính phủ sửa đổi, bổ sung một số điều của các Nghị định liên quan đến kiểm soát thủ tục hành chính;

Xét đề nghị của Vụ trưởng Vụ Kế hoạch - Tài chính và Chánh Văn phòng,

QUYẾT ĐỊNH:

Điều 1. Công bố kèm theo Quyết định này thủ tục hành chính được sửa đổi, bổ sung hoặc thay thế thuộc phạm vi chức năng quản lý của Bộ Giáo dục và Đào tạo.

Điều 2. Quyết định này có hiệu lực thi hành kể từ ngày 01 tháng 12 năm 2018.

Điều 3. Chánh Văn phòng, Vụ trưởng Vụ Kế hoạch - Tài chính, Thủ trưởng các đơn vị có liên quan chịu trách nhiệm thi hành Quyết định này.

Nơi nhận	**KT. BỘ TRƯỞNG**
- Như Điều 3;	***(Đã ký)***
- Văn phòng Chính phủ (Cục Kiểm soát TTHC) (để b/c);	
- Website Bộ GDĐT;	**Phùng Xuân Nhạ**
- Lưu: VT, KHTC. VP (KSTTHC: 02).	

Figure 4.13: An exported DOC file

This is a sample of an exported file that the user can get from the exporting process.

CHAPTER 5

EVALUATION AND DISCUSSION

5.1 Evaluation

To evaluate the effectiveness of the application, several tests have been performed. All these tests have been individually analyzed in the scope of optical character recognition and image processing. For the comparison, the overall results on many different document qualities have been presented in table 5.1. Each row in the table represents one experiment performed on a particular document.

The table is composed of these following columns:

- Document – imaged document that has been used as a sample for testing
- Quality - refer to the level of accuracy in which different imaging systems capture (high for a scanned image that has DPI over 300, low for an image taken from camera or smartphone)
- Type – the type of document that has been converted, which can be an image that has been cleaned via image-processing or an original image
- Character accuracy – the accuracy of the OCR engine recognition on the character level, calculated as follows:

$$A_c = 1 - \frac{e}{c}$$

e: the number of character errors
c: the number of all characters in the document

- Components found – the number of document components that the application finds the right place (12 components in total)
- Time – the amount of time to finish the process (including OCR and image denoising)

System specifications for testing:

- CPU: Intel® Core™ i7-8700
- GPU: Nvidia GTX 1070 Ti
- RAM: 8GB 2666MHz

The ID of sample:

- Sample1: 4863/QĐ-BGDĐT
- Sample2: 105/HĐGSNN-VP
- Sample3: 40/TB-ĐHQT

Document	Quality	Type	Character accuracy(%)	Components found	Time(s)
sample1	high	original	95.862	10	3.104
sample1	high	cleaned	98.905	12	3.467
sample2	high	original	97.402	11	4.494
sample2	high	cleaned	98.880	12	4.839
sample3	high	original	95.133	11	4.195
sample3	high	cleaned	96.808	12	3.954
sample1	low	original	88.739	11	3.024
sample1	low	cleaned	89.143	11	3.101
sample2	low	original	86.243	10	2.726
sample2	low	cleaned	87.424	11	3.080
sample3	low	original	79.224	9	2.879
sample3	low	cleaned	82.494	10	3.276

Table 5.11: Accuracy and performance comparison

In general, the “cleaned” type provides more accurate results in terms of character level accuracy, it also gets more correct components position. But the time it takes to finish the process is longer (~300ms).

The accuracy of the high-quality image getting from the scanner overwhelms the low one (>11.62%). So, the better the input, the higher the efficiency.

We can also see that the accuracy with the clean document from scanning is over 95 percent while the additional time is just a little bit. So the trade-off between efficiency and performance is worth it.

For each test, the system will need to provide an average of 500MB of RAM to support OpenCV and Tesseract in image processing and text conversion.

5.2 Discussion

The results of the tests have shown us that the algorithms running in the application are able to meet the requirements of the thesis.

It is possible to digitize documents automatically only by providing a sufficiently powerful system and good quality input. That will greatly assist in reducing administrative workload, reducing manpower, and wasting time.

This web application will be suitable for companies or organizations, where so many documents are stored that they need a new method to handle and manage. Besides, the algorithm used in this application has the potential to develop and apply to many other text digitization models in the future.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The objectives of this thesis are researching and implementing an application to convert the paper document into the digitalized document. In this project, we have finished some functions such as image processing using OpenCV, convert the image using Tesseract, fix and preview documents in printed form, saving digitalized documents using mongoDB, as well as exporting the DOC file.

During the time of researching and creating this project, a lot of practical experiences are gained. Firstly, we were able to design a graphical user interface, what we need to do, and what we should not, to make the user approach the program easily. Secondly, it was a good chance to read more about new technology and try some new interesting library, thanks to them, we don’t have to suffer from futile works. Especially, we learned how to schedule and manage the time so that we can catch up on the deadline, which is very useful for the next time when we work in the industry.

6.2 Future work

At the end, we finally created a program that can do some basic functions to convert and get digitalized documents. But still, this project is not finished yet, there are many problems that we need to handle to make the program better. In the future, the method will need to be improved to enhance the accuracy, dealing with handwriting recognization and signature, then optimize the code to make it run faster. Finally, the mobile version is considered to develop if the web application gets good feedback.

REFERENCES

[1] “Express/Node introduction,” Jun 1, 2020. Accessed on: May 5, 2020 [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction

[2] Mayank Tripathi, “Understanding How the Chrome V8 Engine Translates JavaScript into Machine Code”. Dec 20, 2017. Accessed on: Jun 14, 2020. [Online]. Available: https://medium.com/free-code-camp/understanding-the-core-of-nodejs-the-powerful-chrome-v8-engine-79e7eb8af964,

[3] Nitin Pandit, “What and Why React.js”. [Online]. Available: https://www.c-sharpcorner.com/article/what-and-why-reactjs, Mar 05, 2020

[4] Christudas B, “Install, Configure, and Run MongoDB”, in Practical Microservices Architectural Patterns. Apress, Berkeley, CA, 2019

[5] DocxJS – Getting Started.[Online]. Accessed on: May 24, 2020. Available: https://docx.js.org/

[6] Chung B.W, “Getting Started with Processing and OpenCV”, in: Pro Processing for Images and Computer Vision with OpenCV. Apress, Berkeley, CA, 2017

[7]: Ray Smith, “An Overview of the Tesseract OCR engine” Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society (2007), pp. 629-633

[8] VietOCR – Usage. Accessed on: July 10, 2020. [Online]. Available: http://vietocr.sourceforge.net/usage.html

[9] “Electronic Document Management Software”. Accessed on: May 17, 2020 [Online]. Available: https://luutru.gov.vn/phan-mem-quan-ly-tai-lieu-luu-tru-dien-tu-dap-ung-yeu-cau-quy-dinh-tai-thong-tu-so-02-2019-tt-bnv.htm

[10] Fan, L. Zhang, F. Fan, H. et al. Brief review of image denoising techniques. Vis. Comput. Ind. Biomed. Art 2, 7 (2019). https://doi.org/10.1186/s42492-019-0016-7

[11] OpenCV – Finding contours in your image. Accessed on: Jun 6, 2020. [Online]. Available:https://docs.opencv.org/2.4/doc/tutorials/imgproc/shapedescriptors/find_contours/find_contours.html

[12] J.Kolce, N.Jacques, "How to Build and Structure a Node.js MVC Application". Accessed on: May 23, 2020.[Online]. Available: https://www.sitepoint.com/node-js-mvc-application, March 23, 2020

[13] Koichi Kise. "Page Segmentation Techniques in Document Analysis," in Handbook of Document Image Processing and Recognition. Springer, 2014, ch. 2, pp. 135-175