TABLE OF CONTENTS

CHAPTER NO. **TITLE** **PAGE NO.**

CHAPTER 1. INTRODUCTION

In the current world, there is a growing demand for software systems to recognize characters written on a piece of paper in computer systems. These days there is a huge demand in “storing the information available in these paper documents such as newspapers and journals into a computer storage disk and then later reusing this information by searching process”. One simple way to store information of these paper documents in the computer system is to first scan the documents and then store them as IMAGES. But to use this information is very difficult since we have to read the individual contents and search the contents from these documents line-by-line and word-by-word which is really inefficient and is no better than reading these documents physically.

Optical character recognition usually abbreviated to OCR, involves a computer system designed to translate images of typewritten text (usually captured by a scanner) into machine editable text. OCR traces its roots back to telegraphy. On the eve of the First World War, physicist Emanuel Goldberg invented a machine that could read characters and convert them into telegraph code. In the 1920s, he went a step further and created the first electronic document retrieval system. In 1931 he was granted USA Patent number 1,838,389 for the invention. The patent was acquired by IBM.

Today, there’s a host of OCR service providers offering technology (often accessible via APIs) capable of recognising most characters and fonts to a high level of accuracy. Even though the technology continues to improve, there is always scope for errors. And that means costly human intervention to validate information and ensure it’s ready to be used by the wider organisation – not to mention the economic and environmental cost of persisting with paper.

Ultimately, the truly “paperless business” doesn’t (yet) exist and data extraction is still a useful tool that can augment e-document processing. The scope of our project is to provide an efficient and enhanced software tool for reading and recognizing the characters in research, academic, governmental and business journals.

CHAPTER 2. SOFTWARES USED WITH DESCRIPTION

The softwares that we have used –

1. **ATOM IDE -** The most basic way to create and run a Python program is to create an empty file with a .py extension and then point to that file from the command line with python filename.py. Alternatively, you can use IDLE which comes as a default application along with Python to execute your code. However, if you want to be productive, the first two options would not be the best ones. You will need to use something more reliable and productive. Here is where the Atom comes into the picture. Atom does not have features in the traditional sense, it creates packages that add to its hackable core. These packages provide features like auto-complete, code lines, and code highlighters.

   We have used the following packages –

   - Script - The Script package displays a document about the details of other packages such as commands, shortcuts, etc.
   - Atom-file-icons - This package will add icons preceding your files in the tree view
   - Minimap and Minimap-highlight-selected - Opening file with many lines of code will be displayed as a whole on the window towards the right side. The minimap-highlight-selected will highlight the function or variable which is selected as white patches on that mini window

2. **Dreamweaver –** Adobe Dreamweaver CC is a web design and an Integrated Development Environment (IDE) application that is used to develop and design websites. Dreamweaver includes a code editor that supports syntax highlighting, code completion, real-time syntax checking, and code introspection for generating code hints to assist the user in writing code.

   Dreamweaver, like other HTML editors, edits files locally then uploads them to the remote web server using FTP, SFTP, or WebDAV. Dreamweaver CS4 supports the Subversion (SVN) version control system.

   We have used dreamweaver to design our homepage layout. We have created a dedicated file for all the CSS styling of the entire site. You can do so via Tools > CSS > Attach Style Sheet.

CSS is the part that provides all the styling on a web page. It allows you to define colors, the dimensions of elements, font types and sizes, and a whole lot more. We want to use the markup to spruce up our page title. we have created a CSS Selector for the Page Title,Changed the Headline Font,centered the Headline and changed Its Size.

3. **Open CV -** OpenCV-Python is a library of Python bindings designed to solve computer vision problems.

   We have used cv2.cvtColor() method to convert an image from one color space to another. There are more than 150 color-space conversion methods available in OpenCV.

   We have used cv2.threshold() method to change the thresh hold of the gray image so that its more accurately readable.

4. **Pillow –** In today's digital world, we come across lots of digital images. In case, we are working with Python programming language, it provides lot of image processing libraries to add image processing capabilities to digital images.

   Pillow is built on top of PIL (Python Image Library). PIL is one of the important modules for image processing in Python. It supports wide variety of images such as "jpeg", "png", "bmp", "gif", "ppm", "tiff". You can do almost anything on digital images using pillow module apart from basic image processing functionality, including point operations, filtering images using built-in convolution kernels, and color space conversions.

   - Image Archive – The Python Imaging Library is best suited for image archival and batch processing applications. Python pillow package can be used for creating thumbnails, converting from one format to another and print images, etc.
   - For debugging purposes, there is a show () method to save the image to disk which calls the external display utility.
   - Image Processing – The Pillow library contains all the basic image processing functionality. We can do image resizing, rotation and transformation.
   - We have used Image.open(fp, mode='r') syntax to open our image.
     - fp − A filename (string), pathlib.Path object or a file object. The file object must implement read(), seek() and tell() methods and be opened in binary mode.
     - mode − It's an optional argument, if given, must be 'r'.
     - Return value − An Image object.
     - Error − If the file cannot be found, or the image cannot be opened and identified.

5. **Flask –** Flask is a web application framework written in Python. Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

   We have installed virtualenv , a virtual Python environment builder. It helps a user to create multiple Python environments side-by-side. Thereby, it can avoid compatibility issues between the different versions of the libraries.

   Flask constructor takes the name of current module (_name_) as argument.

   The route() function of the Flask class is a decorator, which tells the application which URL should call the associated function.

   - Syntax -> app.route(rule, options)
     - The rule parameter represents URL binding with the function.
     - The options is a list of parameters to be forwarded to the underlying Rule object.

   Modern web frameworks use the routing technique to help a user remember application URLs. It is useful to access the desired page directly without having to navigate from the home page thats why we have used the route() decorator in Flask.

   We have built a URL dynamically, by adding variable parts to the rule parameter. This variable part is marked as <variable-name>. It is passed as a keyword argument to the function with which the rule is associated. The URL rules of Flask are based on Werkzeug’s routing module. This ensures that the URLs formed are unique and based on precedents laid down by Apache.

6. **Numpy –** In machine learning, Python uses image data in the form of a NumPy array, i.e., [Height, Width, Channel] format. To enhance the performance of the predictive model to load and manipulate images. In Python, we can perform one task in different ways.

   NumPy Or numeric python is a popular library for array manipulation. Since images are just an array of pixels carrying various color codes. NumPy can be used to convert an array into image. It is used along with the Pillow library to perform image processing functions. NumPy uses the asarray() class to convert PIL images into NumPy arrays. The np.array function also produce the same result. The type function displays the class of an image.The process can be reversed using the Image.fromarray() function.

Now that we have converted our image into a Numpy array, we might come across a case where we need to do some manipulations on an image before using it into the desired model. After performing the manipulations, it is important to save the image before performing further steps. The format argument saves the file in different formats, such as PNG, GIF, or PEG.

NumPy uses the asarray() class to convert PIL images into NumPy arrays. The np.array function also produce the same result. The type function displays the class of an image.

The process can be reversed using the Image.fromarray() function.

Our approach:

- Create a numpy array.
- Reshape the above array to suitable dimensions.
- Create an image object from the above array using PIL library.
- Save the image object in a suitable file format.

NumPy uses the asarray() class to convert PIL images into NumPy arrays. The np.array function also produce the same result. The type function displays the class of an image.

The process can be reversed using the Image.fromarray() function.

7. **Pytesseract –** Pytesseract or Python-tesseract is an Optical Character Recognition (OCR) tool for Python. It will read and recognize the text in images, license plates etc. Python-tesseract is actually a wrapper class or a package for Google's Tesseract-OCR Engine. It is also useful and regarded as a stand-alone invocation script to tesseract, as it can easily read all image types supported by the Pillow imaging library that we have implemented in our project.

   - Preprocessing for Tesseract
     - To avoid all the ways your tesseract output accuracy can drop, you need to make sure the image is appropriately pre-processed.
     - This includes rescaling, binarization, noise removal, deskewing, etc.

In our project we have first cinverted our image from BGR to Gray scale and later changed the threshold for most accurate results.

8. **OS –** The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The os and os.path modules include many functions to interact with the file system. In our project we have used these methods -

   - os.popen(): This method opens a pipe to or from command. The return value can be read or written depending on whether mode is 'r' or 'w'.Parameters mode & bufsize are not necessary parameters, if not provided, default 'r' is taken for mode.
   - os.truncate() method truncates the file corresponding to path so that it is at most length bytes in size. This function can support file descriptor too.
   - os.seek() method sets the current position of file descriptor fd to the given position pos which is modified by how.
   - os.write() method in Python is used to write a bytestring to the given file descriptor.

   A file descriptor is small integer value that corresponds to a file that has been opened by the current process. It is used to perform various lower level I/O operations like read, write, send etc.

   Note: os.write() method is intended for low-level operation and should be applied to a file descriptor as returned by os.open() or os.pipe() method.

CHAPTER 3. METHODS IMPLEMENTED

The main methods/functions implemented in our Website are:

1) **Allowed_file (filename) –**

   This function is used to check the extension of the file that is being uploaded on the server. The argument passed here is the full filename along with the extension. It splits the name after the '.' and check whether the extension exists in the allowed list of extensions.

2) **App_route (/reset) and def test() –**

   This app route and function is used to reset the current list of words that we have been passed through the OCR Character Recognition system. After resetting the list and the text file it return the original page in essence resetting the whole website.

3) **App_route (/return_files) and def return_files_tut() –**

   This app route and function is used to operate the download button wherein the text that has been read by the OCR Character Recognition system can be downloaded as a text file (.txt). It uses the send_file function of flask within which the parameters passed are the file that is to be downloaded. The cache timeout has been set to -1 so that the download page does not cache and is refreshed every time.

4) **App_route (/) and def upload_page() –**

   This is the main page of our website it uses the render_template function of Flask to display the upload.html page (which is our main page). Here the first thing that we check is whether the file that we want to upload exists or if the user has not selected any files the appropriate message is displayed. If everything works the file is saved in the static/uploads folder in the project folder after which the image is passed to ocr_core function for processing. The returned string from the ocr_core function is stored in a variable named extracted_text after which it is stored in a list and then the whole page is rendered again along the extracted text.

5) def ocr_core(filename) –

This is the main function of our program. Here the pre-processing and the OCR character recognition of our image takes place. It takes the argument ‘filename’ which is essentially the path of the image. After that the image is opened using the pillow library and converted into an array using the Numpy library. After that the image is converted into black and white and then the thresholding of the image takes place which means that all the pixels having a value of less than 127 are converted to 0 (perfectly black) and all the pixels whose value is greater than 127 are converted to 255 (perfectly white) in essence removing all the grey pixels. And then finally, this image is passed to the tesseract library which reads the image and returns a string.

CHAPTER 4. SCREENSHOTS

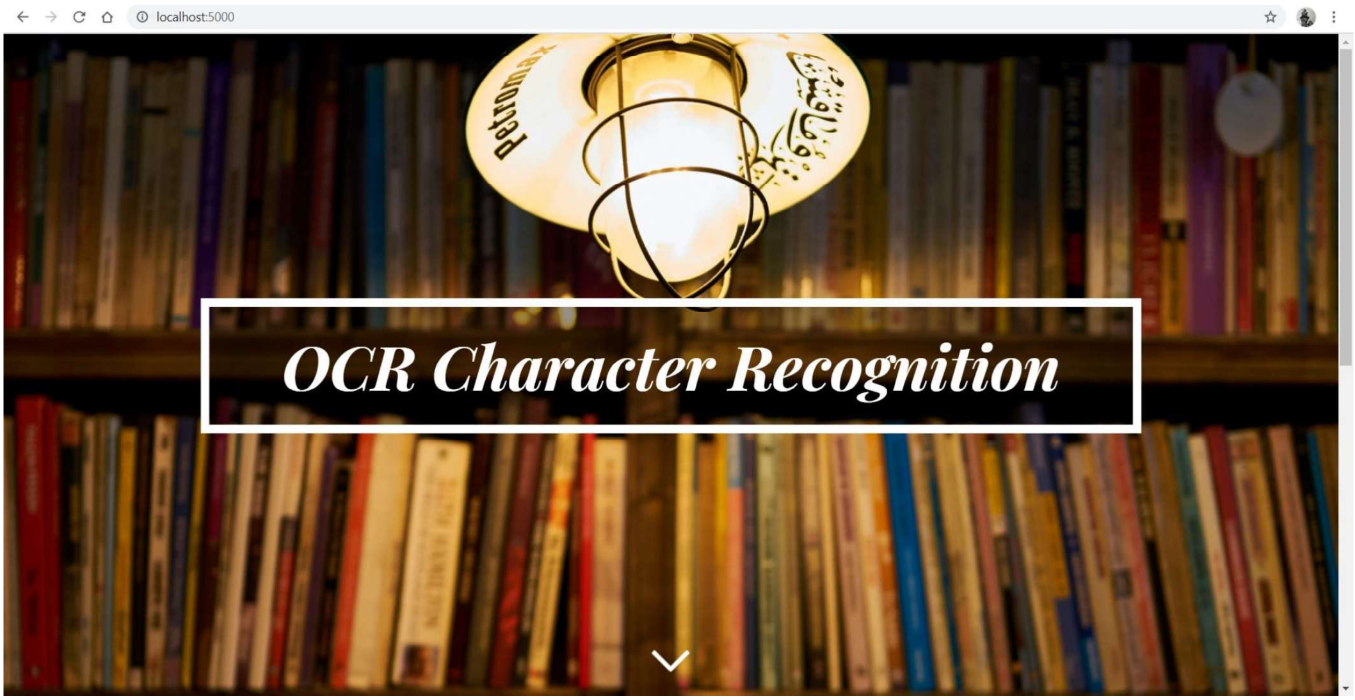
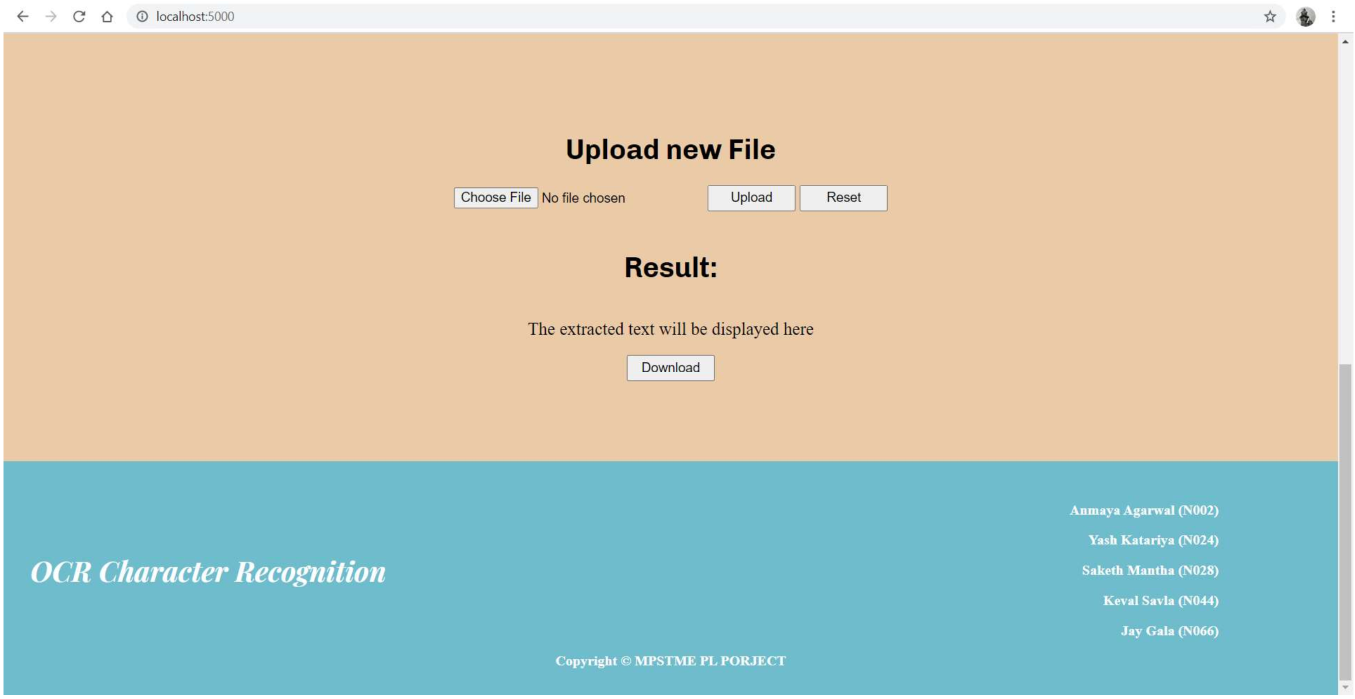


Figure 4.1 Welcome Page



Upload new File

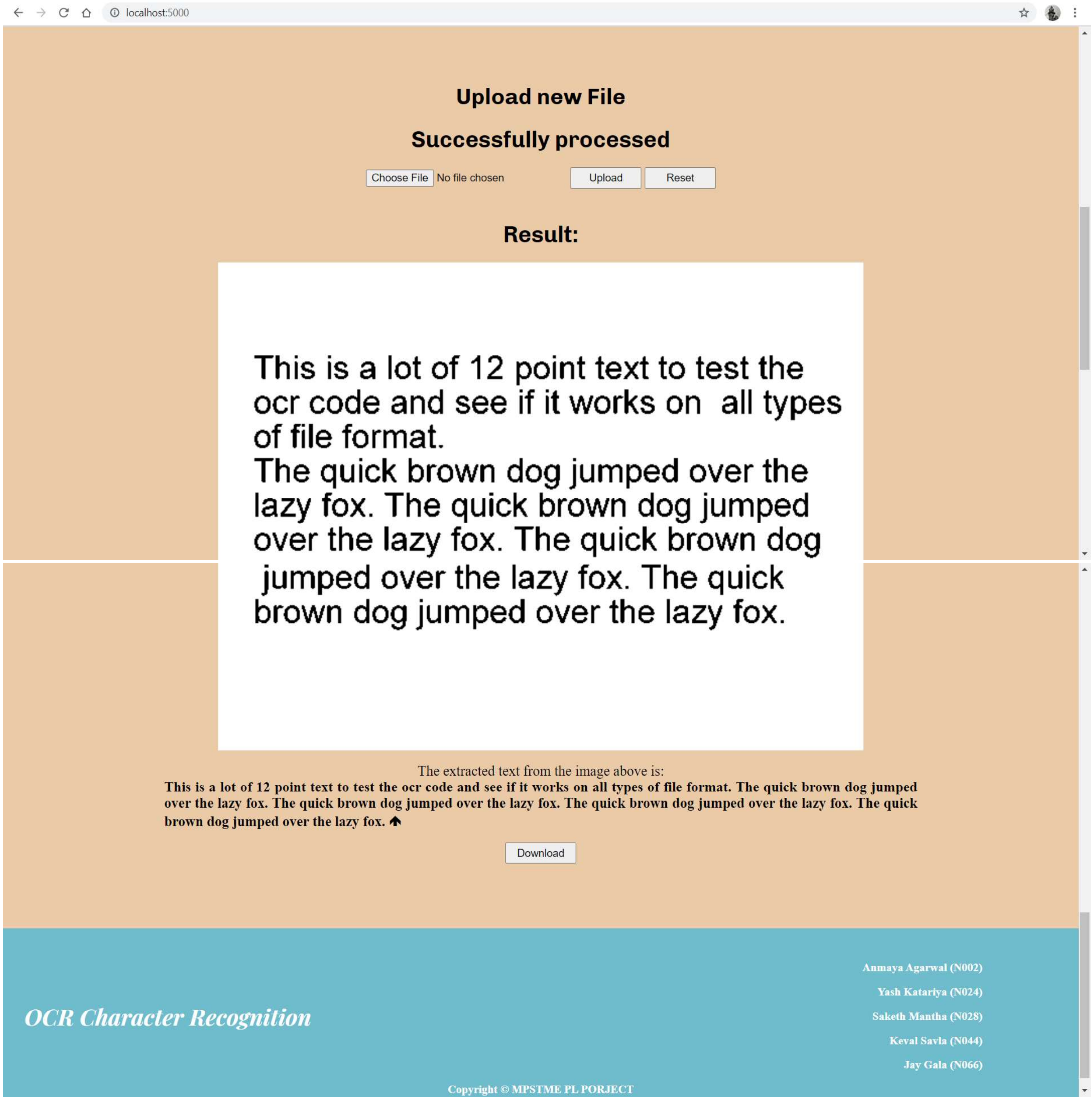
Result:

The extracted text will be displayed here

Anmaya Agarwal (N002)
Yash Katariya (N024)
Saketh Mantha (N028)
Keval Savla (N044)
Jay Gala (N066)

Copyright © MPSTME PL PORJECT

Figure 4.2 Home Page



Upload new File

Successfully processed

Result:

This is a lot of 12 point text to test the ocr code and see if it works on all types of file format.
The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox.

The extracted text from the image above is:

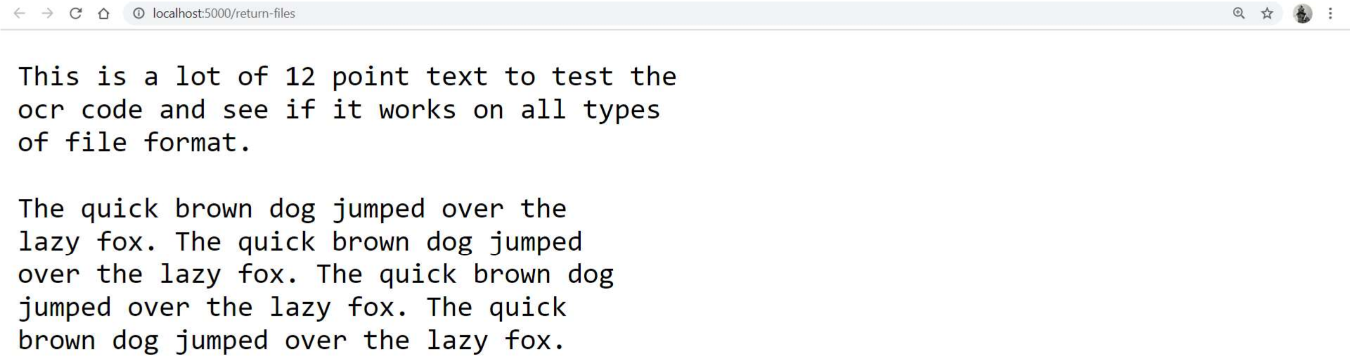
This is a lot of 12 point text to test the ocr code and see if it works on all types of file format. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. 🡅

OCR Character Recognition

Anmaya Agarwal (N002)
Yash Katariya (N024)
Saketh Mantha (N028)
Keval Savla (N044)
Jay Gala (N066)

Copyright © MPSTME PL PORJECT

Figure 4.3 Working Example



```
This is a lot of 12 point text to test the
ocr code and see if it works on all types
of file format.

The quick brown dog jumped over the
lazy fox. The quick brown dog jumped
over the lazy fox. The quick brown dog
jumped over the lazy fox. The quick
brown dog jumped over the lazy fox.
```

Figure 4.4 Download Page

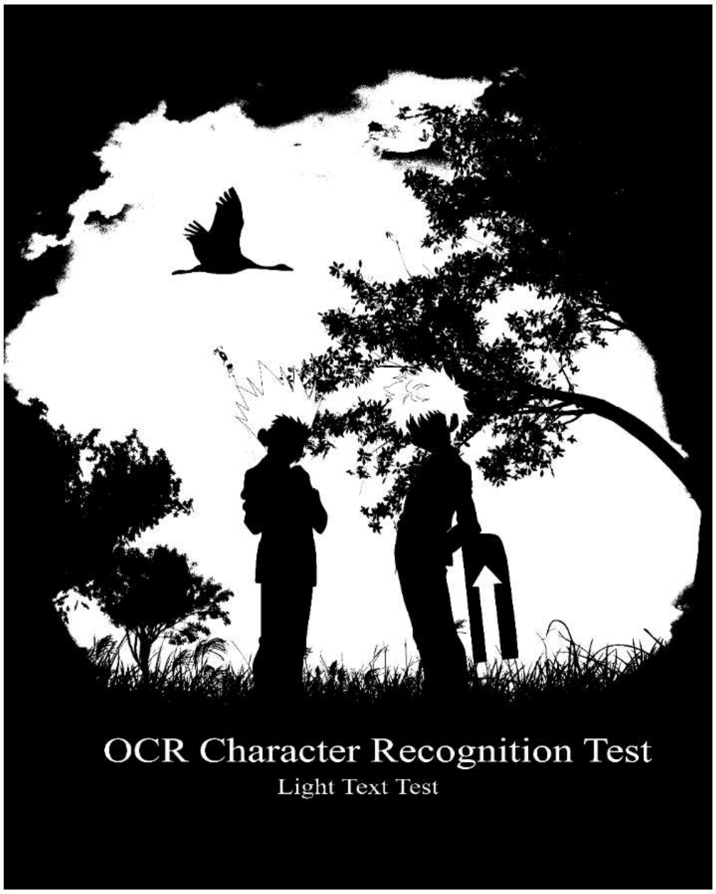
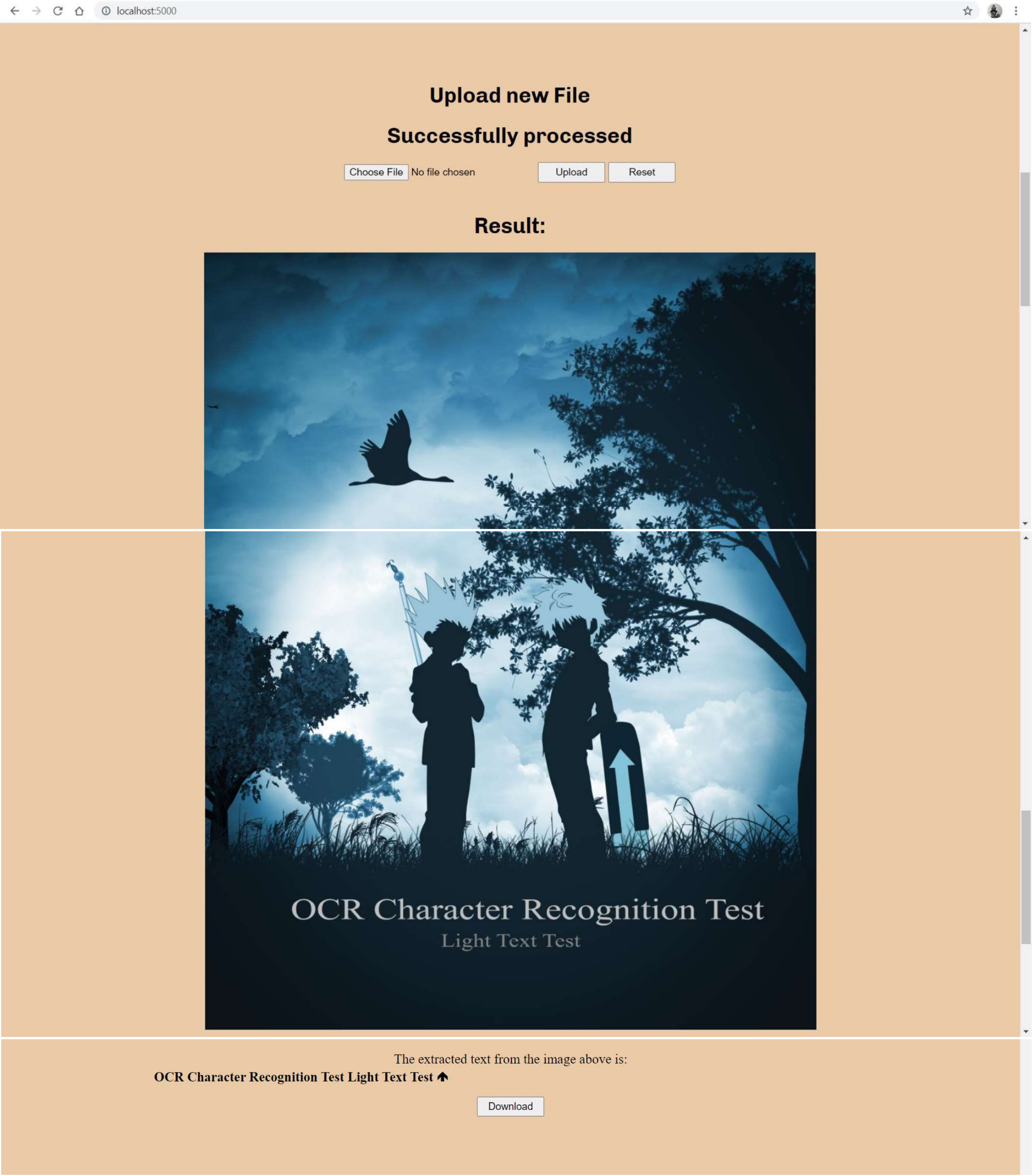


Figure 4.5 Image after Pre-Processing



Upload new File

Successfully processed

Result:

The extracted text from the image above is:

OCR Character Recognition Test Light Text Test 🡅

Figure 5.6 Image Before Pre-Processing and the result

CHAPTER 5. CONCLUSION AND FUTURE SCOPE

OCR technology is becoming very widespread in professional environments such as historical archives, museums, and libraries. This is a great way to preserve ancient texts or images in a digital format. More importantly, these documents can also be examined in the digital domain without disturbing the original physical materials.

In the old days of paper databases, looking for a specific piece of information was always a long process. The task could have been even more time-consuming had the database not been organized properly. By contrast, OCR technology can help users identify specific information in a matter of seconds. Since digital texts are fully searchable, this technology transforms paper documents, images, and even handwritten transcriptions into electronic records that you can seamlessly access with a computer search.

In addition to the obvious working advantages of OCR, it is also worth mentioning that this particular technology provides many benefits in terms of sheer physical space. Many businesses had the opportunity to downsize quite a lot since they no longer need large spaces to manage and store massive amounts of paper documents. On other occasions, they were able to use the areas that were previously meant for storage to handle different tasks, thus increasing the productivity of the business and even allowing the opportunity to branch out.

In conclusion, OCR is a very remarkable technology that holds a lot of potential. In this day and age, such tools are already quite advanced. However, Optical Character Recognition is going to look even better in the future. AI is on the way to becoming one of the most influential trends in the coming years, revolutionizing information as we know it.

We are currently in efforts to optimize the pre-processing phase to make the image readable for the pytesseract library, which in turn will result in increased output accuracy. Pre-processing of images can be done via several ways including using libraries such as OpenCV, Pillow, etc. Image processing is a vast domain and as that technology gets developed so will OCR character recognition,

Another approach to improve OCR Character recognition is to train a machine learning model which will better classify and read text from a variety of images. Several models like pytesseract already exist but even they cannot read every type of font so with the development of Machine Learning and AI it is possible in the future to read texts with an astounding level of accuracy.

CHAPTER 6. SOCIETAL APPLICATION

Through this website, our aim is to provide electronic or mechanical conversion of images of typed, handwritten, printed text into machine encoded text, whether from a scanned document, a photo of a document or from subtitle text superimposed on an image.

OCR character recognition is widely used as a form of data entry for printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, letters, printouts of static-data, or any suitable documentation. It is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining.

This technology has various applications like Automatic number plate recognition, in airports for passport recognition and information extraction, Traffic Sign Recognition, Defeating CAPTCHA anti-bot systems though these are specifically designed to Prevent OCR, assistive technology for blind and visually impaired users and making scanned documents searchable by converting them to searchable PDFs.

The newest update to Google Translate for Android adds Google Goggles' optical character recognition (OCR) technology, so you can translate text using only your device's camera lens. Whereas previously you had to type, hand-write (onscreen), or say your text aloud in order to use the app, this new feature requires none of the above.