**Jaypee E. Ayawan**

Provided a simple PHP code for OCR, using a windows machine make it work and create a manual to be save in a pdf file.

**Manual**

1. First you have to download the tesseract application and install.

- <https://github.com/UB-Mannheim/tesseract/wiki> or https://digi.bib.uni-mannheim.de/tesseract/

1. After installing the OCR Application in any location, In my PC its installed in “C:\Program Files\Tesseract-OCR”

- update this line of code **“$tesseractPath = "C:\\Program Files\\Tesseract-OCR\\tesseract.exe";**” depending on where the tesseract application was installed

Part 1

1. **Provide information on what an OCR is, as the first page**

* Optical Character Recognition (OCR) is a technology that converts text from images into machine-readable text.
* OCR is also known as text recognition or text extraction.
* **How does OCR work?**
* OCR software analyzes images to recognize characters like numbers, letters, and symbols.
* It can extract the text as words, paragraphs, or text blocks.
* Some OCR software can convert the characters into editable text.
* Advanced OCR software can also export the text's layout, formatting, and size.
* **What are some uses of OCR?**
* Scanning documents like forms, invoices, reports, and articles
* Extracting text from images like posters, product labels, and street signs
* Digitizing and processing printed materials like bank checks
* Identifying license plate numbers by toll collection cameras
* Identifying road signs in self-driving cars
* **History of OCR**
* The earliest use of OCR was in telegraphy technology and reading devices for the blind.
* Emanuel Goldberg invented a machine that read printed characters and converted them into telegraph code.
* Yann LeCun developed an AI model that could recognize handwritten numbers.

1. **List all the needed software used for running the program including the logo, version, and link**
2. First you have to download the tesseract application and install.

- <https://github.com/UB-Mannheim/tesseract/wiki> or <https://digi.bib.uni-mannheim.de/tesseract/>

1. Provide comments on the source code on what it does, to be presented in a 3-column table with titles - code/line, function, output (if necessary)

|  |  |  |
| --- | --- | --- |
| **Line of Code** | **Function** | **Output** |
| $outputText = ""; | Initializes a variable to store extracted text. |  |
| $uploadedImage = ""; | Initializes a variable to store the uploaded image path. |  |
| if ($\_SERVER["REQUEST\_METHOD"] == "POST" && isset($\_FILES["image"])) { | Checks if the form is submitted via POST and an image file is uploaded. | Proceeds if conditions are met |
| $targetDir = "uploads/"; | Defines the directory where uploaded images will be stored. | "uploads/" |
| if (!file\_exists($targetDir)) { mkdir($targetDir, 0777, true); } | Creates the upload directory if it doesn't exist, with full permissions. | "uploads" folder |
| $imageFileType = strtolower(pathinfo($targetFile, PATHINFO\_EXTENSION)); | Extracts and converts the file extension to lowercase. | "jpg", "png", etc. |
| $allowedTypes = ["jpg", "jpeg", "png", "bmp", "gif", "tiff"]; | Defines an array of allowed image formats. | Allowed types list |
| if (in\_array($imageFileType, $allowedTypes)) { | Checks if the uploaded file type is in the allowed list. | Proceeds if valid, otherwise shows error |
| if (move\_uploaded\_file($\_FILES["image"]["tmp\_name"], $targetFile)) { | Moves the uploaded file to the target directory. | Returns true if successful |
| $uploadedImage = $targetFile; | Stores the uploaded file path. | "uploads/filename.ext" |
| $tesseractPath = "C:\\Program Files\\Tesseract-OCR\\tesseract.exe"; | Defines the path to the Tesseract OCR executable. | "C:\\Program Files\\Tesseract-OCR\\tesseract.exe" |
| $outputFile = "output"; | Defines a temporary output file name for extracted text. | "output" |
| $command = "\"$tesseractPath\" \"$targetFile\" \"$outputFile\""; | |  | | --- | |  |  |  | | --- | | Forms the command to execute Tesseract OCR on the uploaded image. | | "C:\\Program Files\\Tesseract-OCR\\tesseract.exe uploads/filename.ext output" |
| exec($command); | Executes the OCR command. | Runs Tesseract OCR |
| if (file\_exists("$outputFile.txt")) { | Checks if the extracted text file exists. | Proceeds if file is found |
| $outputText = file\_get\_contents("$outputFile.txt"); | Reads the extracted text from the file. | Extracted text from the image |
| unlink("$outputFile.txt"); | Deletes the temporary text file after reading. | File removed |
| else { $outputText = "OCR failed. No text extracted."; } | Sets an error message if OCR fails. | "OCR failed. No text extracted." |
| else { $outputText = "File upload failed!"; } | Sets an error message if the file upload fails. | "File upload failed!" |
| else { $outputText = "Invalid file type! Please upload an image."; } | Sets an error message if the file type is not allowed. | "Invalid file type! Please upload an image." |
| <!DOCTYPE html> ... <html> | |  | | --- | |  |  |  | | --- | | Starts the HTML page. | | Renders an HTML page |
| <h2>Upload an Image for OCR</h2> | Displays a heading. | |  | | --- | |  |  |  | | --- | | "Upload an Image for OCR" | |
| <form action="" method="post" enctype="multipart/form-data"> | Creates a form for image upload. | File input form |
| <input type="file" name="image" required> | Adds a file input field. | User selects an image |
| <button type="submit">Upload & Extract Text</button> | Adds a submit button. | Button labeled "Upload & Extract Text" |
| <?php if (!empty($uploadedImage)): ?> | |  | | --- | |  |  |  | | --- | | Checks if an image was uploaded | | Displays image if true |
| <img src="<?php echo $uploadedImage; ?>" alt="Uploaded Image"> | Displays the uploaded image. | Image preview |
| <?php if (!empty($outputText)): ?> | Checks if text was extracted. | Displays text if true |
| <textarea rows="10" cols="60"><?php echo htmlspecialchars($outputText); ?></textarea> | Displays extracted text in a textarea. | Extracted text |

1. **Identify the limitation of the program**

**Technical Limitations**

|  |  |
| --- | --- |
| **Limitation** | **Explanation** |
| **Limited Image Format Support** | Only supports jpg, jpeg, png, bmp, gif, and tiff. Other formats like pdf or webp are not processed. |
| **No Preprocessing of Images** | The script does not enhance image quality (e.g., binarization, noise reduction) before running OCR, which may affect accuracy. |
| **Dependent on Tesseract OCR** | Requires **Tesseract-OCR** to be installed and correctly configured. Any misconfiguration leads to OCR failure. |
| **Command Execution Issues** | The script uses exec(), which may be disabled on some servers for security reasons, making OCR processing fail. |
| **Does Not Handle Multi-Page Documents** | If the image contains multiple columns or complex formatting, text extraction may not be accurate. |
| **No Language Selection for OCR** | Tesseract supports multiple languages, but this script does not provide an option to specify a different language. |

**Security Limitations**

|  |  |
| --- | --- |
| **Limitation** | **Explanation** |
| **Potential File Upload Vulnerability** | The script checks file extensions but does not verify the actual MIME type, which could allow malicious file uploads. |
| **No Input Validation on File Names** | Attackers could upload files with names containing special characters, potentially leading to file path manipulation. |
| **No Rate Limiting** | A user can upload multiple files in quick succession, which can overload the server. |
| **No Secure Storage of Uploaded Files** | Uploaded files are stored in the /uploads/ folder without additional security measures. An attacker might access them directly. |
| **No CSRF Protection** | The form lacks a CSRF token, making it susceptible to cross-site request forgery attacks. |

**Usability Limitations**

|  |  |
| --- | --- |
| **Limitation** | **Explanation** |
| **No Feedback During Processing** | Users do not receive progress updates while OCR is running. This can make them think the page is unresponsive. |
| **Limited UI/UX** | The design is very basic. It lacks features like drag-and-drop uploads, file preview before submission, or error messages displayed attractively. |
| **No Error Logging** | If something fails (e.g., OCR error, upload issue), the script does not log details, making debugging harder. |
| **No Mobile Optimization** | The UI does not adjust for mobile devices, making the form harder to use on small screens. |

**Possible Enhancements**

1. **Improve security**

**-** Use “mime\_content\_type()” to verify actual file type, store files securely, and limit file size.

1. **Enhance OCR accuracy**

**-** Add image preprocessing using OpenCV or an image processing library before sending it to Tesseract.

1. **Improve user experience**

- Implement a progress indicator, allow users to select OCR languages, and improve UI styling.

1. **Implement logging**

- Store logs for debugging failed uploads or OCR processing errors.

Part 2

1. Create additional feature such as use of camera
2. Create another part of the manual that will improve its usability base from the identified limitation in part 1, provide details, do the same like in part 1