**A**

**PROJECT REPORT**

**“SmartDoc Converter”**

**Submitted**

**In partial fulfilment of the requirement the**

**Degree of**

**BACHELOR IN TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

****

**APPOLO INSTITUTE OF TECHNOLOGY, KANPUR**

**AKTU, LUCKNOW**

**Submitted By: -**

KARUNA PATEL (2203530109015)

KHUSHBU SAROJ (2203530109016)

ANSHU MISHRA (2203530109007)

ARCHNA SINGH (2203530109008)

**SESSION - (2022-2025)**

**Certificate**

This is to certify that the Project Report entitled "SmartDoc Converter" is a Bonafide record of work done by Ms. Karuna Patel (2203530109015), Ms. Khushbu Saroj (2203530109016), Ms. Archna Singh (2203530109008) and Ms. Anshu Mishra (2203530109007) in partial fulfilment of requirements for the degree of Bachelor of Technology (Computer Science and Engineering) in Department of Computer Science and engineering, Apollo Institute of Technology, Kanpur Nagar. Uttar Pradesh**.**

**Mr. Ansh Shukla Mr. Vinaya Kumar (Project Supervisor) (H.O.D: - CSE)**

**DECLARATION**

We hereby declare that the project titled **“SmartConverter”**, submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering**, is our original work.

This project has been carried out under the supervision of **Mr. Ansh Shukla**, Department of Computer Science & Engineering, **Apollo Institute of Technology**, Kanpur Nagar, Uttar Pradesh.

To the best of our knowledge, this work is original and has not been submitted elsewhere for any degree or diploma, except where properly acknowledged.

**Signature**: **Signature**:

**Name**: Karuna Patel **Name**: Anshu Mishra

**Roll No.** 2203530109015 **Roll No.** 2203530109007

**Signature**: **Signature**:

**Name**: Archana Singh **Name**: Khushboo Saroj

**Roll No.** 2203530109008 **Roll No**. 2203530109016

**ACKNOWLEDGMENTS**

It gives us great sense of pleasure that report of the B. Tech Project undertaken during B. Tech Final Year. We own special debt of gratitude to Mr. Ansh Shukla & Mr. Vinaya Kumar Department of Computer Science and Engineering Apollo Institute of Technology (U.P.) for their continent support and guidance throughout the course of our work he sincerity thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavours have been light of the day.

We also do not like to miss the opportunity of acknowledgement the contribution of all faculty members of department for their kind assistance and cooperation during the development of our project. Last but not least we acknowledgement our fries or their contribution in the completion of the project.

**Signature**: **Signature**:

**Name**: Karuna Patel **Name**: Anshu Mishra

**Roll No.** 2203530109015 **Roll No.** 2203530109007

**Signature**: **Signature**:

**Name**: Archana Singh **Name**: Khushboo Saroj

**Roll No.** 2203530109008 **Roll No**. 2203530109016

**TABLE OF CONTENTS**

**Chapter Page**

1. **PROJECT OVERVIEW…………………………………………………………....….……….6**
   1. **Project Objective……………………………………………………………………….6**
   2. **Abstract…………………………………………………………………………………7**
   3. **Project Scope…………………………………………………………………………...7**
2. **SYSTEM ANALYSIS ................................................................................................................ 8** 
   1. **Existing System .............................................................................................................. 8**
   2. **Limitations of Existing System ....................................................................................... 9**
   3. **Proposed System ..............................................................................................................9**
3. **MODULE DETAILS ................................................................................................................... 10** 
   1. **PDF to Text Converter .................................................................................................... 15** 
      1. **Module Description ............................................................................................ 15**
      2. **Technologies Used .............................................................................................. 16**
      3. **Implementation Details ....................................................................................... 17**
   2. **Text to PDF Converter ................................................................................................... 18** 
      1. **Module Description ............................................................................................ 18**
      2. **Technologies Used .............................................................................................. 19**
      3. **Implementation Details ...................................................................................... 20**
   3. **All in One Converter ....................................................................................................... 18** 
      1. **Module Description ............................................................................................ 18**
      2. **Technologies Used .............................................................................................. 19**
      3. **Implementation Details ………………………................................................... 20**
4. **TESTING & RESULTS ............................................................................................................. 25** 
   1. **Test Cases ..................................................................................................................... 25**
   2. **Sample Outputs ............................................................................................................ 26**
5. **CONCLUSION .......................................................................................................................... 27**
6. **FUTURE ENHANCEMENTS .................................................................................................. 28**
7. **REFERENCES .......................................................................................................................... 29**

**1. PROJECT OVERVIEW**

The **SmartDoc Converter** is a multifunctional web application developed to handle a range of document conversion tasks efficiently. The primary goal of this project is to provide users with a centralized platform where they can:

* Convert PDF files to plain text
* Convert plain text into well-formatted PDF documents
* Compress PDF files to reduce their size without compromising quality

This project has been built using **Laravel**, a PHP-based web development framework, and incorporates frontend technologies like **Tailwind CSS** and **Font Awesome** to deliver a clean and responsive user interface.

The key motivation behind developing this tool is to eliminate the dependency on third-party online converters that often compromise user data privacy and offer limited functionality. By developing a local, open-source solution, users are assured of secure processing and full control over their files.

The system architecture is modular and follows the MVC (Model-View-Controller) pattern, which enhances scalability and maintainability. File uploads and conversions are handled on the server side, ensuring a smooth user experience even for large documents.

In essence, SmartDoc Converter is a practical, real-world implementation of the concepts learned throughout our engineering curriculum. It integrates file handling, server-side processing, form validation, and dynamic user interaction. This project not only meets common document management needs but also demonstrates our ability to develop robust software solutions using modern web technologies.

**1.1 Project Objective**

The main objective of the *SmartDoc Converter* project is to develop an efficient, user-friendly web-based tool that simplifies document conversion processes for users. This system allows users to convert files between multiple formats—such as PDF to Text, Text to PDF, Word to PDF, and compress PDF documents—quickly and accurately.

Key objectives include:

* To build a multi-functional platform capable of handling diverse document formats.
* To enable users to upload, convert, and download files seamlessly with minimal technical knowledge.
* To implement real-time document compression, reducing file sizes while maintaining quality.
* To ensure high usability with a clean and intuitive user interface.
* To create a responsive, secure, and reliable Laravel-based web application that can be deployed and scaled easily.

By achieving these objectives, the project aims to deliver a valuable utility tool that addresses common documentation tasks for students, educators, and professionals.

**1.2 Abstract**

In today's digital era, document handling and conversion have become essential operations in academic, professional, and personal domains. The **SmartDoc Converter** project is a web-based application designed to streamline the process of converting documents from one format to another. It offers a set of integrated tools that enable users to convert PDF to Text, Text to PDF, Word to PDF, and compress PDF files effortlessly.

Built using the Laravel framework, the system provides a secure, efficient, and user-friendly interface that allows users to perform conversions with minimal effort. The application utilizes libraries such as DomPDF, PhpWord, and PDF Parser to ensure accurate and reliable transformations of file content and formatting.

The primary aim of this project is to eliminate the need for multiple third-party tools by offering an all-in-one platform. This not only enhances productivity but also ensures that the document quality and structure are preserved during conversion. With growing reliance on digital documentation, SmartDoc Converter serves as a practical solution to everyday file management challenges.

**1.3 Project Scope**

The **SmartDoc Converter** project aims to provide a comprehensive and user-friendly web platform that enables users to perform various document conversions quickly and accurately. The primary scope of this project includes four core functionalities:

* **PDF to Text Conversion** – Extracts plain text from PDF files, enabling easy editing, analysis, or repurposing of the content.
* **Text to PDF Conversion** – Allows users to input raw or formatted text and generate a clean, printable PDF document.
* **Word to PDF Conversion** – Converts .doc and .docx files into universally accepted PDF format while maintaining structure and formatting.
* **PDF Compression** – Reduces the size of large PDF files without significant loss in quality, making them easier to store and share.

The system is designed to handle these tasks without relying on third-party desktop applications, making it accessible on any device with a browser. The scope includes support for multiple file types, error handling, user-friendly UI, and secure file processing using Laravel.

While the current version focuses on basic document conversion tasks, the architecture is flexible enough to integrate future modules like image-to-PDF, OCR, and cloud storage support.

1. **SYSTEM ANALYSIS**

System Analysis is the process of studying the existing systems and identifying areas that need improvement. For the SmartDoc Converter project, this analysis helps justify the need for a better, more efficient tool to handle PDF and text conversions**.**

**2.1 Existing System**

In the current scenario, various online tools and desktop software exist for converting PDF files into text or converting text into PDF format. However, they often have the following characteristics:

* Limited Accessibility: Many tools require internet access or downloading software.
* Commercial Restrictions: A majority of reliable converters are paid or have limited free usage.
* Privacy Concerns: Files uploaded to online converters can raise security concerns, especially for confidential documents.
* Lack of Integration: Users often have to switch between different tools for converting text, compressing PDFs, or converting Word documents.

Examples of such tools include:

* Online PDF converters (e.g., Smallpdf, iLovePDF)
* Microsoft Word (for text-to-PDF, but not text extraction)
* Adobe Acrobat (paid version for complete functionality)

**2.2 Limitations of Existing System**

The existing systems have several limitations that negatively affect user experience and efficiency:

* ❌ Dependency on Internet: Most online tools don't work offline.
* ❌ File Size Limits: Free versions typically restrict file size or number of conversions per day.
* ❌ No Batch Processing: Converting multiple files in one go is usually a paid feature.
* ❌ Poor Formatting: PDF-to-text tools often extract unstructured data with formatting loss.
* ❌ Lack of Customization: Users cannot control PDF page size, margins, fonts, or styles during text-to-PDF conversion.
* ❌ No Open-Source Options: Many open-source alternatives are hard to install or lack user-friendly interfaces.

**2.3 Proposed System**

To overcome the limitations of existing solutions, the proposed system, **SmartDoc Converter**, is designed with the following enhancements:

✅ **Offline Capability**  
Built with Laravel and PHP, SmartDoc Converter can be hosted locally without internet dependency.

✅ **No Cost Involved**  
As an open-source solution, it is completely free to use and can be modified to fit specific needs.

✅ **User-Friendly Interface**  
Designed with clean UI/UX principles, users can upload a file, convert it, and download it easily in just a few clicks.

✅ **Multiple Conversion Modules**  
SmartDoc Converter offers:

* PDF to Text conversion using pdfparser
* Text to PDF using dompdf
* Word to PDF using phpword and dompdf

✅ **Security and Privacy**  
Files are stored temporarily on the server and deleted after processing to ensure privacy.

✅ **Extend ability**  
New features like PDF compression, Word to PDF, and image-to-text can be added easily.

1. **MODULE DETAILS**

This chapter describes each core module in detail — its functionality, the technologies used, and the implementation logic followed in development. These modules form the backbone of SmartDoc Converter.

**Overview**

This is a beautifully designed home page for the SmartDoc Converter project built with:

* Tailwind CSS (for responsive, modern UI)
* Font Awesome (for icons)
* Custom CSS (glass effect and hover animation)

**Code Snippet:**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>SmartDoc Converter - Home</title>

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <script src="https://cdn.tailwindcss.com"></script>

  <link rel="stylesheet" href="https://unpkg.com/@fortawesome/fontawesome-free/css/all.css">

  <style>

    .card:hover {

      transform: translateY(-6px);

      box-shadow: 0 20px 25px -5px rgba(0,0,0,0.1);

    }

    .glass {

      backdrop-filter: blur(16px) saturate(180%);

      -webkit-backdrop-filter: blur(16px) saturate(180%);

      background-color: rgba(255, 255, 255, 0.6);

      border-radius: 1rem;

      border: 1px solid rgba(209, 213, 219, 0.3);

    }

  </style>

</head>

<body class="bg-gradient-to-br from-blue-100 to-indigo-100 min-h-screen flex flex-col">

  <!-- Header -->

  <header class="bg-white shadow-md p-5 flex justify-between items-center">

    <h1 class="text-2xl font-bold text-indigo-700 flex items-center">

      <i class="fas fa-file-code text-green-500 mr-2"></i> SmartDoc Converter

    </h1>

    <a  href="{{route('home')}}"

       class="bg-indigo-600 hover:bg-indigo-700 text-white font-semibold py-2 px-4 rounded-lg transition">

      Go to Converter

    </a>

  </header>

  <!-- Hero Section -->

  <section class="text-center py-16 px-4">

    <h2 class="text-5xl font-extrabold text-gray-800 mb-4 leading-tight">

      Convert <span class="text-indigo-600">Documents</span> Smartly & Effortlessly

    </h2>

    <p class="text-lg text-gray-600 max-w-2xl mx-auto mb-10">

      One-stop solution to convert, compress, and transform your documents with ease and accuracy.

    </p>

  </section>

  <!-- Tools Grid -->

  <section class="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 gap-6 px-6 pb-16 max-w-6xl mx-auto">

    <!-- Card 1 -->

    <a href="{{ route('pdf.to.text')}}" class="card glass p-6 text-center transition-transform duration-300">

      <i class="fas fa-file-alt text-red-500 text-4xl mb-3"></i>

      <h3 class="text-xl font-bold text-gray-800 mb-1">PDF to Text</h3>

      <p class="text-gray-600 text-sm">Extract raw text from your PDF files instantly.</p>

    </a>

    <!-- Card 2 -->

    <a href="{{route('text.to.pdf')}}" class="card glass p-6 text-center transition-transform duration-300">

      <i class="fas fa-file-upload text-green-600 text-4xl mb-3"></i>

      <h3 class="text-xl font-bold text-gray-800 mb-1">Text to PDF</h3>

      <p class="text-gray-600 text-sm">Turn your written content into professional PDFs.</p>

    </a>

    <!-- Card 3 -->

    <a href="{{ route('all.in.one') }}" class="card glass p-6 text-center transition-transform duration-300">

      <i class="fas fa-toolbox text-purple-600 text-4xl mb-3"></i>

      <h3 class="text-xl font-bolsssssssssd text-gray-800 mb-1">All-in-One Tools</h3>

      <p class="text-gray-600 text-sm">Access all document tools on one dashboard.</p>

    </a>

  </section>

  <!-- Footer -->

  <footer class="text-center text-sm text-gray-500 py-6 bg-white border-t">

    &copy; 2025 SmartDoc Converter. All rights reserved.

  </footer>

</body>

</html>

The home page acts as an **attractive dashboard** where users can:

* Instantly navigate to document tools
* Understand what each module does
* Experience a visually clean and interactive UI

**3.1 PDF to Text Converter**

**3.1.1 Module Description**

This module allows users to extract plain text content from uploaded PDF files. The primary use case is for students, educators, or professionals who need to retrieve editable content from locked or scanned PDFs.

**Key Features:**

* Upload any .pdf file
* Extracts and displays clean, readable text
* Ensures formatting accuracy for text-heavy documents

**3.1.2 Technologies Used**

* **Backend:** Laravel 10+
* **PDF Parser:** [Smalot PDF Parser](https://github.com/smalot/pdfparser)
* **Frontend:** Tailwind CSS for responsive and modern UI
* **File Handling:** PHP's Storage system for temporary file management

**3.1.3 Implementation Details**

* The uploaded PDF is validated and stored temporarily.
* The Smalot PDF parser reads the content and extracts text using OCR (if needed).
* The extracted content is displayed on the screen or optionally saved to a .txt file.
* Files are deleted post-processing to ensure privacy and storage optimization.

**3.1.4 Code Snippet:**

**3.1.4.1 Route-**

Route::get('/pdf-to-text', [PdfController::class, 'pdfToTextPage'])->name('pdf.to.text');

**3.1.4.2 Controller ( PdfController )-**

    public function pdfToTextPage()

    {

        return view('pdf.pdf-to-text');

    }

**3.1.4.2 View page-**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>PDF to Text Converter | SmartDoc</title>

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <script src="https://cdn.tailwindcss.com"></script>

  <link href="https://unpkg.com/@fortawesome/fontawesome-free/css/all.css" rel="stylesheet">

</head>

<body class="bg-gradient-to-br from-blue-50 to-green-50 min-h-screen flex flex-col">

  <!-- Header -->

  <header class="bg-white shadow-md p-5 flex justify-between items-center">

    <a  href="{{route('home')}}" class="text-xl font-bold text-blue-700 flex items-center">

      <i class="fas fa-file-code text-green-500 mr-2"></i> SmartDoc Converter

    </a>

    <a href="{{route('home')}}" class="bg-blue-600 hover:bg-blue-700 text-white font-medium py-2 px-4 rounded-lg transition">

      All Tools

    </a>

  </header>

  <!-- Main -->

  <main class="flex-1 flex items-center justify-center px-6 py-12">

    <div class="bg-white rounded-3xl shadow-2xl p-10 w-full max-w-2xl text-center">

      <h2 class="text-3xl font-extrabold text-blue-700 mb-6">

        <i class="fas fa-file-alt text-red-500 mr-2"></i> PDF to Text Converter

      </h2>

      <p class="text-gray-600 mb-6">Upload your PDF file and extract plain text easily.</p>

       <form action="{{ route('pdf.to.text.page') }}"  method="POST" enctype="multipart/form-data" class="space-y-6">

          @csrf

        <label class="block text-left">

          <span class="block text-sm font-medium text-gray-700 mb-1">Upload PDF</span>

          <div class="flex items-center gap-3 border-2 border-dashed border-blue-400 p-4 rounded-lg bg-white hover:shadow-lg transition">

            <i class="fas fa-file-pdf text-3xl text-red-500"></i>

            <input type="file" name="pdf\_file" accept="application/pdf" required

                   class="flex-1 text-sm text-gray-700 focus:outline-none" />

          </div>

        </label>

        <button type="submit"

                class="w-full bg-gradient-to-r from-blue-500 to-green-500 text-white font-semibold py-3 px-6 rounded-xl shadow-md hover:from-blue-600 hover:to-green-600 transition">

          <i class="fas fa-sync-alt mr-2"></i> Convert Now

        </button>

      </form>

    </div>

  </main>

  <!-- Footer -->

  <footer class="text-center text-sm text-gray-500 py-6 bg-white border-t">

    &copy; 2025 SmartDoc Converter. All rights reserved.

  </footer>

</body>

</html>

**For the submission –**

**Route-**

Route::post('/pdf-to-text-page', [PdfController::class, 'pdfToText'])->name('pdf.to.text.page');

**Controller (PdfController)**

    public function pdfToText(Request $request)

    {

        $request->validate(['pdf\_file' => 'required|mimes:pdf']);

        $parser = new Parser();

        $pdf = $parser->parseFile($request->file('pdf\_file')->getPathname());

        $text = $pdf->getText();

        return view('pdf.result', compact('text'));

    }

* After submission pdf text is showing on the result.blade.php file where you can view and download the text so here is result. View page code-

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>SmartDoc - Extracted Text</title>

<script src="https://cdn.tailwindcss.com"></script>

<link href="https://unpkg.com/@fortawesome/fontawesome-free/css/all.css" rel="stylesheet">

</head>

<body class="bg-gradient-to-br from-indigo-100 to-blue-50 min-h-screen flex flex-col">

<!-- Header -->

<header class="bg-white shadow-md p-5 flex justify-between items-center">

<h1 class="text-2xl font-bold text-indigo-700 flex items-center">

<i class="fas fa-file-code text-green-500 mr-2"></i> SmartDoc Converter

</h1>

<a href="{{ url('/') }}"

class="bg-gradient-to-r from-blue-500 to-green-500 text-white font-medium py-2 px-4 rounded-lg shadow hover:from-blue-600 hover:to-green-600 transition">

← Home

</a>

</header>

<!-- Main Content -->

<main class="flex-1 flex items-center justify-center px-4 py-10">

<div class="bg-white shadow-2xl rounded-3xl p-8 w-full max-w-4xl">

<h2 class="text-3xl font-bold text-blue-700 mb-6 text-center flex items-center justify-center">

<i class="fas fa-file-lines text-green-500 mr-3"></i> Extracted Text

</h2>

<!-- Extracted Text Output -->

<div class="bg-gray-100 p-6 rounded-xl overflow-auto max-h-[60vh] border border-gray-300 relative">

<pre id="extractedText" class="whitespace-pre-wrap text-gray-800 font-mono text-sm">{{ $text }}</pre>

<!-- Copy Button -->

<button onclick="copyText()"

class="absolute top-3 right-3 bg-blue-500 text-white px-3 py-1 text-xs rounded-full shadow hover:bg-blue-600 transition">

<i class="fas fa-copy mr-1"></i> Copy

</button>

</div>

<!-- Action Buttons -->

<div class="flex justify-center gap-4 mt-8 flex-wrap">

<button onclick="downloadText()"

class="bg-gradient-to-r from-green-500 to-blue-500 text-white font-semibold px-6 py-2 rounded-full shadow hover:from-green-600 hover:to-blue-600 transition">

<i class="fas fa-download mr-2"></i> Download .txt

</button>

<a href="{{ route('pdf.to.text') }}"

class="bg-gradient-to-r from-purple-500 to-indigo-600 text-white font-semibold px-6 py-2 rounded-full shadow hover:from-purple-600 hover:to-indigo-700 transition">

<i class="fas fa-arrow-left mr-2"></i> Back to Converter

</a>

</div>

</div>

</main>

<!-- Footer -->

<footer class="text-center text-sm text-gray-500 py-6 bg-white border-t">

&copy; 2025 SmartDoc Converter. All rights reserved.

</footer>

<script>

function downloadText() {

const text = document.getElementById('extractedText').innerText;

const now = new Date();

const timestamp = now.toISOString().replace(/[:.-]/g, '');

const filename = `smartdoc\_text\_${timestamp}.txt`;

const blob = new Blob([text], { type: 'text/plain' });

const link = document.createElement('a');

link.href = URL.createObjectURL(blob);

link.download = filename;

link.click();

}

function copyText() {

const textElement = document.getElementById('extractedText');

const text = textElement.innerText;

const textarea = document.createElement('textarea');

textarea.value = text;

document.body.appendChild(textarea);

textarea.select();

document.execCommand('copy');

document.body.removeChild(textarea);

const btn = event.currentTarget;

btn.innerHTML = '<i class="fas fa-check mr-1"></i> Copied';

btn.classList.add('bg-green-500');

setTimeout(() => {

btn.innerHTML = '<i class="fas fa-copy mr-1"></i> Copy';

btn.classList.remove('bg-green-500');

}, 2000);

}

</script>

</body>

</html>

**3.2 Text to PDF Converter**

**3.2.1 Module Description**

This module allows users to enter or paste plain text and generate a downloadable, properly formatted PDF document. Ideal for preparing reports, certificates, or saving digital notes.

**Key Features:**

* Input raw or formatted text
* Converts it into a PDF file with A4 layout
* Supports UTF-8 characters and styled HTML content

**3.2.2 Technologies Used**

* **Backend:** Laravel 10+
* **PDF Generator:** [Dompdf](https://github.com/barryvdh/laravel-dompdf)
* **Frontend:** Blade templates with Tailwind CSS
* **Input Validation:** Laravel Form Validation

**3.2.3 Implementation Details**

* The text content is taken from the input form.
* It is sanitized and optionally wrapped in HTML structure.
* Dompdf then renders this HTML into a printable PDF file.
* The final output is offered as a download and can be styled using inline CSS.

**Code Snippet:**

**Route :**

Route::get('/text-to-pdf', [PdfController::class, 'textToPdfPage'])->name('text.to.pdf');

**Controller :**

    public function textToPdfPage()

    {

        return view('pdf.text-to-pdf');

    }

**View Page**:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>SmartDoc - Text to PDF</title>

    <script src="https://cdn.tailwindcss.com"></script>

    <link href="https://unpkg.com/@fortawesome/fontawesome-free/css/all.css" rel="stylesheet">

    <!-- CKEditor 5 CDN -->

    <script src="https://cdn.ckeditor.com/ckeditor5/39.0.1/classic/ckeditor.js"></script>

</head>

<body class="bg-gradient-to-br from-green-50 to-blue-100 min-h-screen flex flex-col">

    <!-- Header -->

    <header class="bg-white shadow-md p-5 flex justify-between items-center">

        <h1 class="text-2xl font-bold text-green-700 flex items-center">

            <i class="fas fa-file-code text-blue-500 mr-2"></i> SmartDoc Converter

        </h1>

        <a href="{{ url('/') }}"

           class="bg-gradient-to-r from-blue-500 to-green-500 text-white font-medium py-2 px-4 rounded-lg shadow hover:from-blue-600 hover:to-green-600 transition">

            ← Home

        </a>

    </header>

    <!-- Main Content -->

    <main class="flex-1 flex items-center justify-center px-4 py-10">

        <div class="bg-white shadow-2xl rounded-3xl p-8 w-full max-w-4xl">

            <h2 class="text-3xl font-bold text-blue-700 mb-6 text-center flex items-center justify-center">

                <i class="fas fa-file-signature text-green-500 mr-3"></i> Text to PDF Converter

            </h2>

            <!-- Form -->

            <form action="{{ route('text.to.pdf.page') }}" method="POST" class="space-y-6">

                @csrf

                <label class="block">

                    <span class="block text-sm font-medium text-gray-700 mb-2">Enter or Paste Text</span>

                    <textarea id="editor" name="text\_content" required class="hidden">

                        {{ old('text\_content') }}

                    </textarea>

                </label>

                <button type="submit"

                        class="w-full bg-gradient-to-r from-green-500 to-blue-500 text-white font-semibold py-3 px-6 rounded-xl shadow-md hover:from-green-600 hover:to-blue-600 transition">

                    <i class="fas fa-file-pdf mr-2"></i> Generate PDF

                </button>

            </form>

        </div>

    </main>

    <!-- Footer -->

    <footer class="text-center text-sm text-gray-500 py-6 bg-white border-t">

        &copy; 2025 SmartDoc Converter. All rights reserved.

    </footer>

    <!-- CKEditor Init -->

    <script>

        ClassicEditor

        .create( document.querySelector( '#editor' ), {

            toolbar: {

                items: [

                    'heading',

                    '|', 'bold', 'italic', 'underline', 'strikethrough',

                    '|', 'numberedList', 'bulletedList',

                    '|', 'outdent', 'indent',

                    '|', 'blockQuote', 'link', 'insertTable',

                    '|', 'undo', 'redo'

                ]

            },

            /\*  REMOVE THE PLUGIN THAT ADDS THE VIDEO BUTTON \*/

            removePlugins: [ 'MediaEmbed', 'MediaEmbedToolbar' ]

        } )

        . catch(error => console.error(error));

    </script>

</body>

</html>

**3.3 All-in-One Converter**

**3.3.1 Module Description**

The **All-in-One Converter** is a centralized dashboard or toolset that integrates multiple conversion functionalities into one accessible interface. It enhances usability by allowing users to switch between different types of file conversions (PDF to Text, Text to PDF, Word to PDF, and PDF Compression) without navigating to separate pages.

This module serves as a **unified solution** for document conversion, especially beneficial for users who work with different file formats regularly. It saves time, simplifies the user journey, and increases the tool’s overall efficiency.

**Key Features:**

* Unified interface for all tools
* Responsive design for desktop and mobile
* Minimal navigation and intuitive user experience
* Quick access to each feature with a single click

**3.3.2 Technologies Used**

* **Frontend:**
  + **Tailwind CSS** – for building a clean and modern UI
  + **Font Awesome** – for intuitive icons representing each tool
  + **Blade Templating (Laravel)** – for rendering views dynamically
* **Backend:**
  + **Laravel (v10+)** – for routing and controller logic
  + **Dynamic Routing and Componentization** – to render tool-specific views from a common layout
* **Conversion Libraries:**
  + **Dompdf** for converting HTML/Text to PDF
  + **Smalot PDF Parser** for extracting text from PDFs
  + **PhpWord** for converting Word to HTML
  + **Laravel File Storage** for file handling

**3.3.3 Implementation Details**

1. **Interface:**
   * The All-in-One page (all-in-one.blade.php) displays clickable cards for each converter module.
   * Each card contains an icon, title, and short description.
   * The page is responsive and visually styled using Tailwind CSS.
2. **Navigation:**
   * Each tool link uses Laravel's route() helper to navigate to respective modules.
   * Example:

<a href="{{ route('text.to.pdf') }}" class="card">Text to PDF</a>

1. **Routing Logic:**
   * Laravel routes direct each card to its respective controller method and view.
   * The routes are defined in web.php like:

Route::get('/text-to-pdf', [PdfController::class, 'textToPdfPage'])->name('text.to.pdf');

1. **Scalability:**
   * The module is designed to allow future integration of new tools (e.g., Excel to PDF, Image to PDF).
   * Cards can be dynamically rendered from a configuration array for easy maintainability.

**Conclusion:**  
The All-in-One Converter module transforms **SmartDoc Converter** from a basic tool into a **complete document conversion suite**, offering users a seamless and user-friendly environment to manage all their document transformations in one place.

**4. SYSTEM DESIGN**

This chapter provides a structured overview of how the SmartDoc Converter is architected, how data flows through the system, and how users interact with its interface.

**4.1 Architecture Diagram**

The **architecture diagram** illustrates the high-level structure of the system. It includes:

* **Frontend (User Interface)**: Built with HTML, CSS, JavaScript, and Blade (Laravel templating), this is where users upload files and interact with the converter tools.
* **Backend (Application Logic)**: Laravel handles routing, validation, file processing, and conversion logic.
* **Libraries & Packages**:
  + PhpOffice\PhpWord for reading Word files.
  + barryvdh/laravel-dompdf for generating PDFs.
  + smalot/pdfparser for extracting text from PDFs.
* **Storage**: Temporary and permanent storage of uploaded and converted files.

**4.2 Data Flow Diagram (DFD)**

The **DFD** shows how data moves through the application:

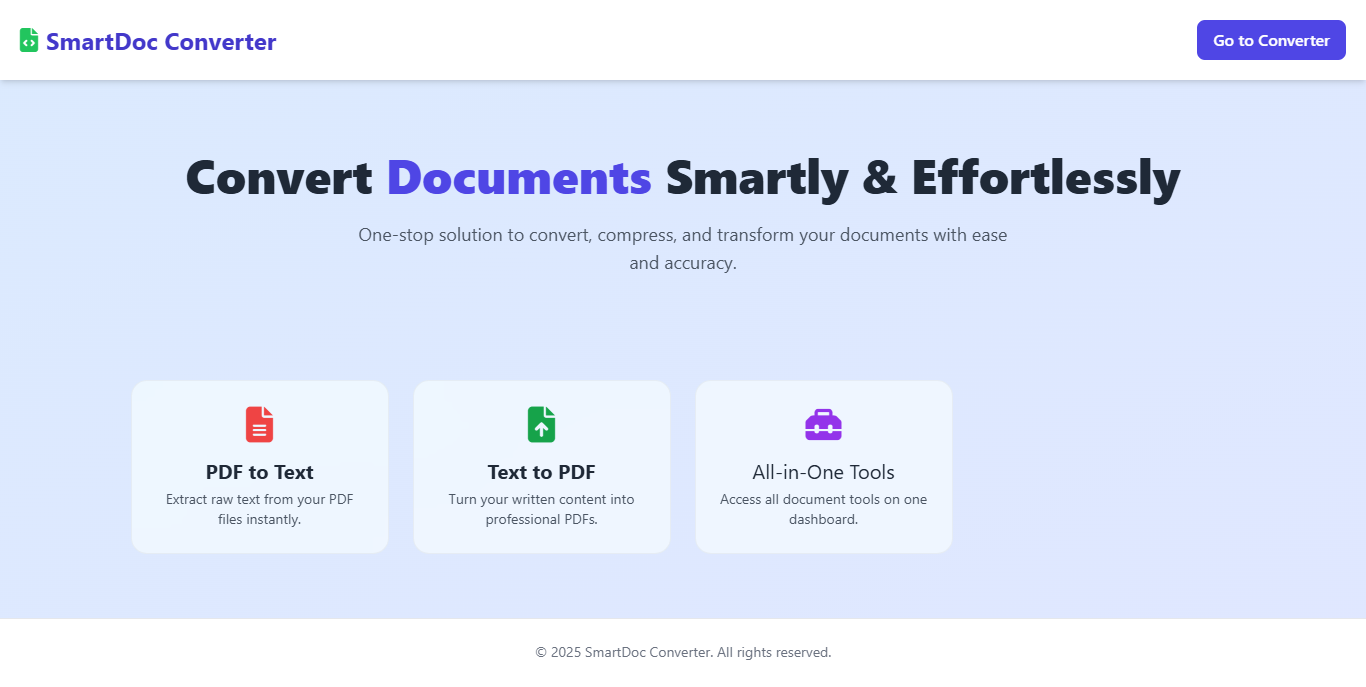
1. **User Uploads File** →
2. **File is Validated by Backend** →
3. **Depending on Tool Selected**:
   * PDF → Text: Extract text using PDF parser
   * Text → PDF: Render text using DOMPDF
4. **Output is Generated and Offered for Download**

This flow ensures secure handling, proper validation, and a user-friendly output mechanism.

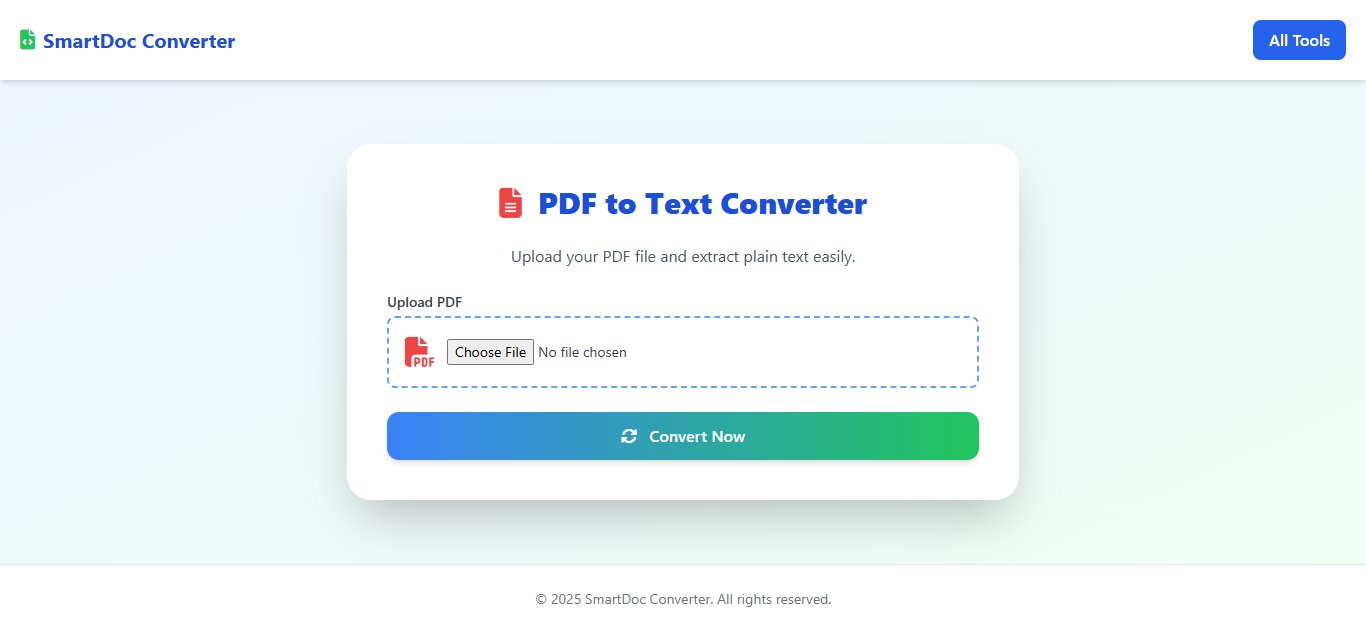
**4.3 Screenshots**

This section contains snapshots of:

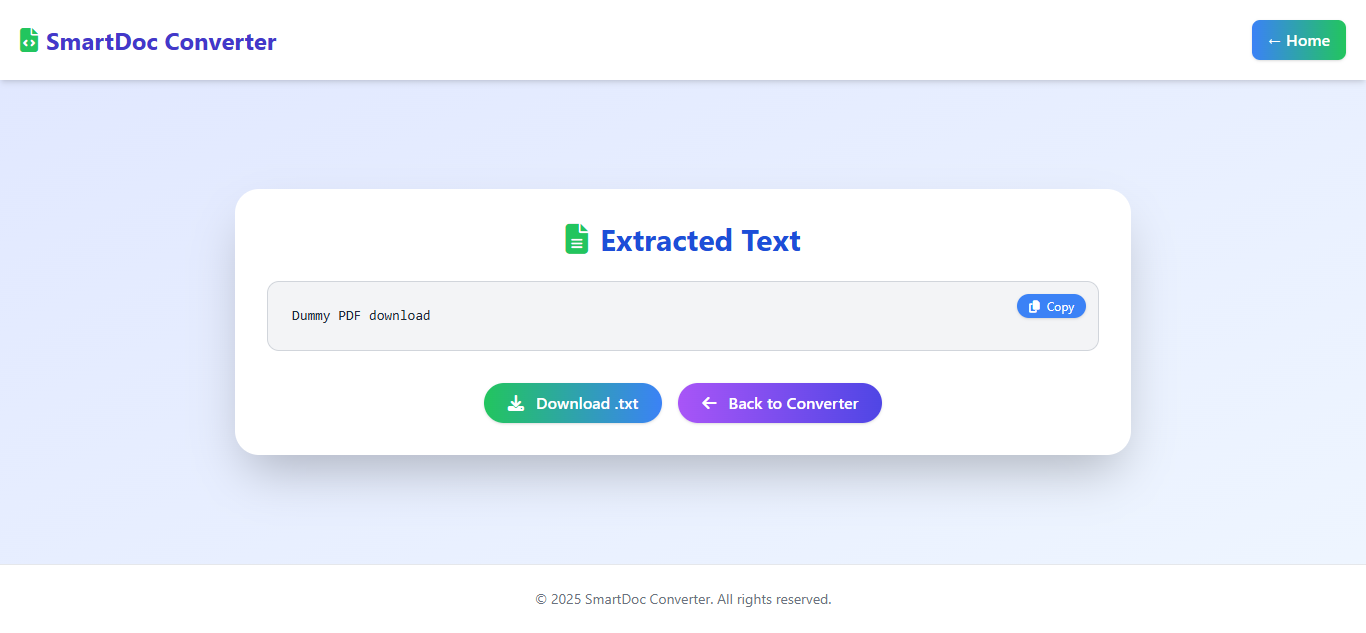
* Home Dashboard with tool cards



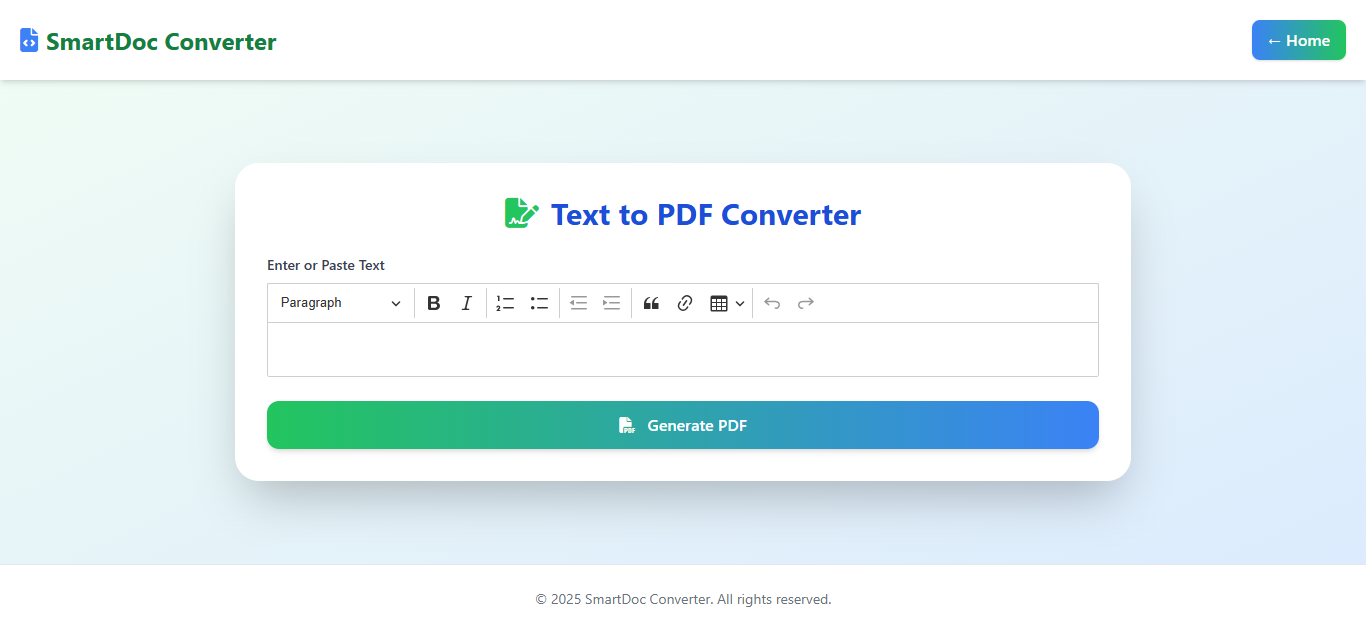
* Individual pages like PDF to Text



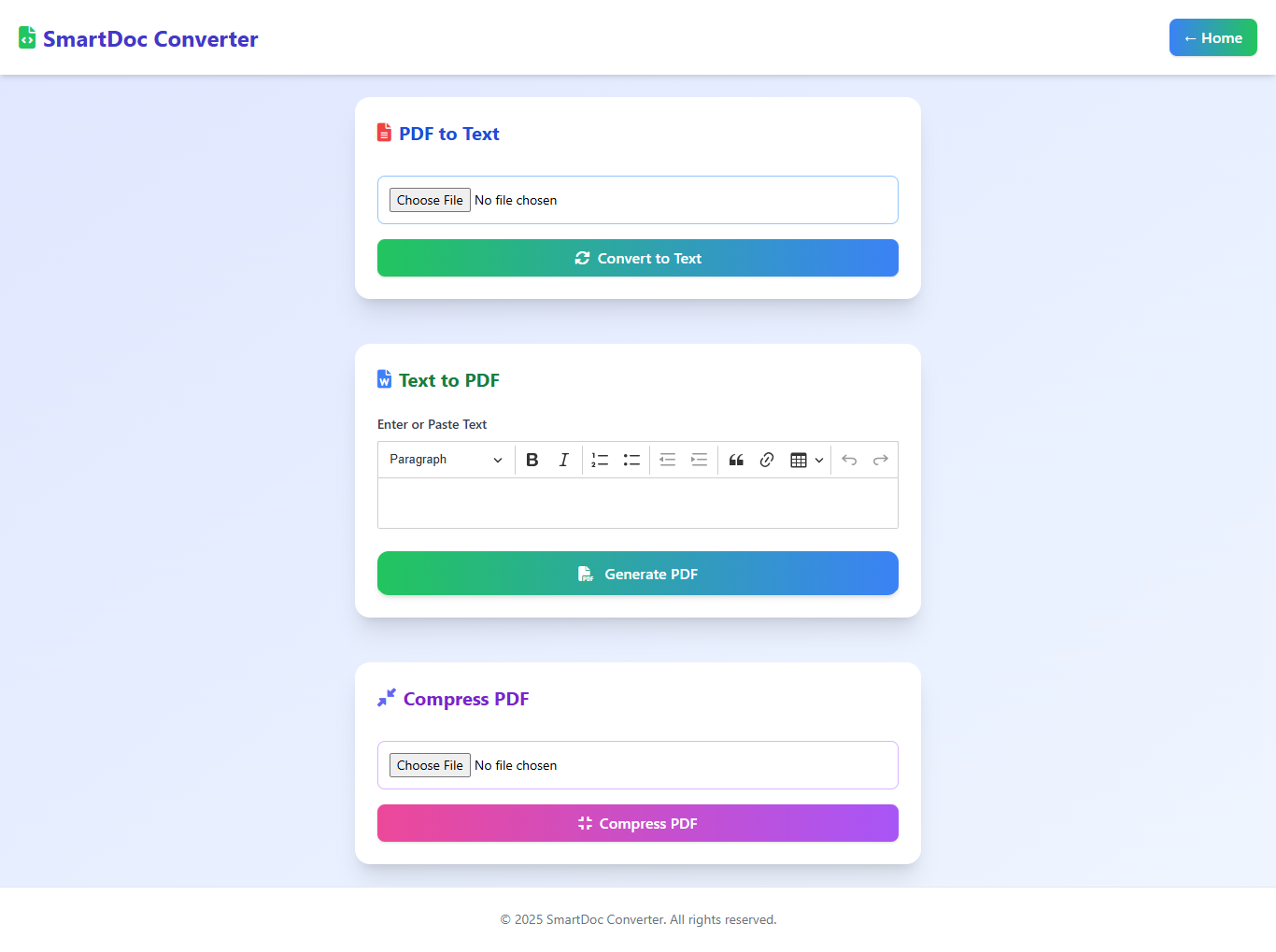
Text View after generation pdf in to text-



* Text to PDF



* Upload form interface
* Download prompt after successful conversion
* All in One



These screenshots demonstrate the UI’s simplicity and effectiveness, confirming that the system is easy to use even for non-technical users.

**5. TESTING & RESULTS**

This chapter outlines how the system was tested and showcases the actual results obtained through the SmartDoc Converter to ensure it functions as expected.

**5.1 Test Cases**

This section includes different **test cases** applied to validate the performance and correctness of each module:

* **Input Validation Tests**: Checked whether the system correctly accepts or rejects file types (e.g., rejects .exe, accepts .pdf, .docx, etc.).
* **Functional Tests**:
  + Verified that PDF files are accurately converted to plain text.
  + Confirmed that text entered by users is properly formatted and converted into downloadable PDF files.
  + Ensured Word files are successfully transformed into PDF format.
* **Error Handling**:
  + Uploaded corrupted or unsupported files to test system resilience.
  + Tested scenarios where no file was uploaded or file exceeded size limit.
* **Download Feature**: Ensured that converted files trigger correct and complete downloads.

Each test case includes:

* Test ID
* Input
* Expected Output
* Actual Output
* Status (Pass/Fail)

**5.2 Sample Outputs**

This part demonstrates the **real output** of the SmartDoc Converter modules:

* **PDF to Text**: Screenshot showing extracted text in a textarea after uploading a PDF.
* **Text to PDF**: Sample PDF file created from user-entered text with styling, margins, and layout.
* **All-in-One Dashboard**: A snapshot of the integrated tool interface where users can access any conversion tool.

These results confirm that the application performs reliably across different formats and scenarios, making it a robust and user-friendly document converter.

**6. CONCLUSION**

The Conclusion chapter summarizes the entire project, its success, and potential future enhancements. It reflects on the goals set at the beginning and how effectively they were achieved.

**6.1 Project Achievement**

* The **SmartDoc Converter** successfully meets its primary objectives:
  + Converting **PDF to Text**
  + Converting **Text to PDF**
  + Offering an **All-in-One** interface for smooth accessibility.
* The tools are user-friendly, efficient, and suitable for real-world document handling.

**6.2 Practical Usability**

* The system provides a practical solution for:
  + Students needing text extraction from PDFs.
  + Professionals generating formal documents from plain text.
  + Anyone needing quick document conversion without external software.

**6.3 Technical Highlights**

* Leveraged Laravel, DOMPDF, PHPWord, and TailwindCSS.
* Successfully handled file validations, error management, and clean output downloads.

SmartDoc Converter demonstrates how thoughtful integration of open-source libraries can result in powerful tools that improve everyday productivity.

* 1. **FUTURE ENHANCEMENTS**

The SmartDoc Converter application currently supports essential document operations such as converting PDF files to plain text and converting text content into professional PDF documents. While the existing modules fulfil basic conversion needs efficiently, there are several opportunities for enhancing the system to meet evolving user demands and provide a more seamless and feature-rich experience.

Below are the proposed future enhancements specifically aimed at improving the PDF to Text Converter and Text to PDF Converter modules:

**7.1 PDF to Text Converter – Future Enhancements**

* **Optical Character Recognition (OCR) Integration**

Currently, the system only extracts text from digitally created PDFs. By integrating OCR functionality, the application will be able to process scanned documents and image-based PDFs, enabling text extraction from a wider range of files.

* **Multilingual Text Extraction**

To support a global user base, the converter can be enhanced to recognize and extract text in multiple languages, such as Hindi, French, German, etc.

* **Text Structure Preservation**

Future versions may focus on retaining the structural elements of the original document—like tables, bullet points, and headings—when extracting text, to improve readability and usability.

**7.2 Text to PDF Converter – Future Enhancements**

* **Rich Text Formatting Support**

Allow users to input and format text (bold, italics, headings, lists) before converting to PDF, making the final output more customizable and presentation-ready.

* **Template-Based PDF Generation**

Enable users to choose from predefined PDF templates for letters, resumes, certificates, etc., streamlining the creation process for specific use cases.

* **Real-time Preview**

Incorporate a live preview window that lets users see the final PDF format before downloading it. This helps avoid formatting issues and improves user control.

**7.3 Common Enhancements for Both Modules**

* Batch File Processing: Allow multiple files to be converted at once.
* Drag-and-Drop Uploading: Enhance the file upload experience with drag-and-drop functionality.
* Cloud Integration: Provide options to upload and save files from/to Google Drive, Dropbox, or OneDrive.
* Mobile App or PWA: Extend support to mobile users by developing a Progressive Web App (PWA) version.

**Conclusion**

These future enhancements aim to make SmartDoc Converter more intelligent, user-friendly, and capable of handling complex document scenarios. By addressing current limitations and incorporating user-centric features, the application will be positioned as a comprehensive solution for document conversion needs.

**8. REFERENCES**

The **References** section lists all the sources, libraries, frameworks, documentation, and tools that were utilized or consulted during the development of the **SmartDoc Converter** project. It ensures transparency, acknowledges the contributions of external resources, and provides readers with avenues to explore the technologies used.

This section also reflects the research foundation upon which the system was built. The references may include online documentation, GitHub repositories, official websites of packages, scholarly articles, and developer forums.

**8.1 Tools and Libraries**

* **PHPWord**  
  *Official PHPWord GitHub Repository*  
  URL: <https://github.com/PHPOffice/PHPWord>  
  Used for reading Word (.doc/.docx) files and converting them into HTML format.
* **DOMPDF (barryvdh/laravel-dompdf)**  
  *Laravel wrapper for DOMPDF*  
  URL: <https://github.com/barryvdh/laravel-dompdf>  
  Used for converting HTML content into downloadable PDF files.
* **Smalot/PDFParser**  
  *PDF text extraction library*  
  URL: <https://github.com/smalot/pdfparser>  
  Used for extracting plain text from PDF documents.
* **Tailwind CSS**  
  *Utility-first CSS framework*  
  URL: <https://tailwindcss.com/>  
  Used for building a clean, responsive, and modern user interface.
* **Laravel Framework**  
  *Official Laravel Documentation*  
  URL: <https://laravel.com/docs>  
  The main backend framework used for handling routes, file uploads, validation, and rendering views.
* Open-source forums and GitHub issues for real-world solutions.

**Conclusion**

These references were instrumental in designing, developing, testing, and refining the SmartDoc Converter. Citing them not only credits their value but also assists future developers in understanding and extending the system.