

Project Report submitted

On

“Nirmal PDF Converter”

Submitted By:
Ravi Sharma
A student of Computer Science & Engineering
At SRM University, Sonipat

Submitted To:
Mr. Mohit Gupta
(Ch. Manager ERP & IT)

**For the fulfillment of the award of the certificate
Of internship at**

**Information & Technology Department
Powergrid Corporation of India**



पावरग्रिड
POWERGRID

Approval Sheet

Title of the Project:

Nirmal PDF Converter

Intern:

Ravi Sharma

BTech 4th Year Computer Science & Engineering Student at
SRM University, Sonipat

Supervisor:

Mr. Mohit Gupta

This is to certify that the project titled "Nirmal PDF Converter" conducted by Ravi Sharma, a BTech 4th year Computer Science & Engineering student at SRM University, has been reviewed and approved by the undersigned for the issuance of the Internship Completion Certificate.

Mr. Mohit Gupta

Ch. Manager - ERP & IT

Powergrid Corporation of India, Gurgaon

Candidate's Declaration

Title of the Project:

Nirmal PDF Converter

Intern:

Ravi Sharma

BTech 4th Year Computer Science & Engineering Student at
SRM University, Sonipat

Declaration:

I hereby declare that the project report entitled "**Nirmal PDF Converter**" submitted to the Human Resource Department, Powergrid Academy of Leadership, Powergrid Corporation of India, is a record of an original work done by me under the guidance of Mr. Mohit Gupta, Ch. Manager, ERP & IT. This project work is submitted in the fulfillment of the requirements for the award of the Internship Completion Certificate at the Powergrid Corporation of India.

I further declare that this project has not been submitted, in part or full, for the award of any other degree or diploma to this university or any other institution.

Ravi Sharma (11021210017)

B.Tech Computer Science (AI & DS)

SRM University, Sonipat

Certificate

This is to certify that the project report entitled "Nirmal PDF Converter" submitted by Ravi Sharma, BTech 4th Year Computer Science Student, has been carried out under the guidance of Mr. Mohit Gupta, Ch. Manager ERP & IT, Powergrid Corporation of India.

The report is submitted in fulfillment of the requirements for the award of the Internship Completion Certificate at the Powergrid Corporation of India.

It is certified that the work presented in this project is original and has not been submitted elsewhere for the award of any degree or diploma.

Mr. Mohit Gupta

Ch. Manager - ERP & IT

Powergrid Corporation of India, Gurgaon

Acknowledgement

I extend my heartfelt thanks to Mr. Mohit Gupta, Ch. Manager ERP & IT, Powergrid Corporation of India, Gurgaon for his invaluable guidance, encouragement, and insightful feedback throughout the duration of this project. His expertise and dedication have been instrumental in the successful completion of this research.

I am deeply grateful to Mr. Ved Prakash, Executive Secretary (HRD), Powergrid Academy of Leadership, Power Grid Corporation of India Limited, for providing me with the opportunity to undertake this internship. His leadership and vision have been a source of inspiration for me.

Finally, I appreciate the resources and facilities provided by Powergrid Corporation of India, which have significantly contributed to the successful completion of this research.

Ravi Sharma (11021210017)

B.Tech Computer Science (AI & DS)

SRM University, Sonipat

Index

S.No.	Title	Page No.
1.	Section I, Introduction - <ul style="list-style-type: none">• Powergrid Corporation of India• Problem Statement• Nirmal PDF Converter	1-5
2.	Section II, Platforms & Libraries - <ul style="list-style-type: none">• Visual Studio IDE• Streamlit• Python Libraries• SQL Express Database	6-13
3.	Section III, File Conversions - <ul style="list-style-type: none">• Excel to PDF• JPG to PDF• PDF to JPG• PNG to PDF• Word to PDF	14-18
4.	Section IV, PDF Operations - <ul style="list-style-type: none">• Compress PDF• Extract PDF• Organize PDF• Protect PDF• Sign PDF• Watermark PDF• Unlock PDF• Split PDF• Merge PDF	19-27
5.	Section V, Backend - <ul style="list-style-type: none">• Database Connectivity• User Data• File Meta Data	28-30
6.	Section VI, Security - <ul style="list-style-type: none">• Login & Authentication• File Signatures• Stored Procedures	31-33
7.	Section VII, User Interface	34-62
8.	Conclusion	63
9.	References	64

Section I, Introduction

Powergrid Corporation of India

Power Grid Corporation of India Limited (POWERGRID), is a Schedule 'A', 'Maharatna' Public Sector Enterprise of Govt. of India which was incorporated on 23rd Oct 1989 under the Company Act, 1956. POWERGRID is a listed Company, with 51.34% holding of Government of India and the balance is held by Institutional Investors and public.

POWERGRID's mission is to "undertake the transmission of electric power through an interconnected network" and to provide reliable, efficient, and secure electricity transmission services. The company's vision is to be "a global transmission company with dominant leadership in emerging power markets, ensuring reliability, safety, and economy."

As of the latest data, POWERGRID operates over 172,000 circuit kilometers of transmission lines and more than 260 substations with a transformation capacity exceeding 420,000 MVA. The company's robust and expansive network ensures the seamless flow of electricity across various regions, connecting remote and urban areas alike. This vast infrastructure is crucial for the integration of renewable energy sources into the national grid, supporting India's commitment to sustainable energy.

POWERGRID is at the forefront of adopting advanced technologies to enhance the reliability and efficiency of the power transmission system. The company has implemented state-of-the-art systems such as the Supervisory Control and Data Acquisition (SCADA) system, Wide Area Measurement Systems (WAMS), and various smart grid technologies. These innovations help in real-time monitoring, control, and management of the grid, reducing transmission losses and improving overall operational efficiency.

POWERGRID has extended its expertise beyond national borders, undertaking consultancy assignments and executing projects in over 20 countries, including Nepal, Bhutan, Bangladesh, and Sri Lanka. The company's international footprint underscores its technical prowess and leadership in the global power transmission sector. POWERGRID has received numerous awards and accolades for its exceptional performance, including recognition for its operational efficiency, project management, and innovation.

Problem Statement

The need for secure and efficient document management is critical for any organization, especially for a large and strategically important entity like Powergrid Corporation of India. The company deals with a vast amount of documentation that needs to be managed, shared, and stored securely. Currently, employees often rely on third-party applications for various file conversion and PDF-related tasks. However, these third-party solutions present several challenges and risks:

- 1. Security Risks:** Third-party applications can pose significant security risks. Sensitive information may be exposed to unauthorized access, data breaches, and other security vulnerabilities during the file conversion and manipulation processes.
- 2. Dependence on External Products:** Relying on external software can lead to dependency issues. Any changes in the third-party application's availability, pricing, or terms of service can disrupt the workflow and productivity of Powergrid employees.
- 3. Lack of Customization:** Third-party solutions may not be fully customizable to meet the specific needs and standards of Powergrid Corporation of India. Custom features, security protocols, and compliance requirements might not be adequately addressed by off-the-shelf software.

To address these issues, I was tasked with developing a secure, in-house application that supports various file conversions and PDF operations. I was allowed to use any programming language and framework available to create this application. The application should not only meet the file conversion needs of the organization but should also store the related file Meta data and the user login data with a functional dashboard and management information system for the users.

The creation of this application for Powergrid Corporation of India aims to ensure that all file conversion and PDF management tasks are performed securely within the organization's controlled environment. This reduces the dependency on third-party applications and mitigates the associated security risks. Additionally, it allows for the implementation of customized features that adhere to Powergrid's specific requirements and standards, enhancing overall efficiency and security in document management processes.

Nirmal PDF Converter

The solution I developed for this problem statement is the Nirmal (निर्मल) PDF Converter, a comprehensive web application that utilizes Streamlit and other Python libraries to meet the organization's requirements. This application is designed to perform various file conversions and PDF operations securely within the organization's controlled environment.

The File conversion Formats Supported –

1. Word to PDF

The application supports seamless conversion from Word (DOCX) documents to PDF format. This conversion ensures that documents maintain their original layout, fonts, and images, while also providing enhanced security and compatibility across different platforms.

2. Merging PDF's

The application supports merging multiple PDF documents into a single cohesive file. This feature allows users to combine several PDFs into one, simplifying document organization and enhancing workflow efficiency. Merged PDFs retain their individual contents, such as pages, bookmarks, and annotations.

3. PNG to PDF

The application supports converting PNG images to PDF format. This feature allows users to take a Multiple PNG image and convert it into a PDF document, ensuring the image retains its original quality and resolution. This functionality is ideal for making sure that it is easier view on various devices.

4. JPG to PDF

The application provides functionality to convert multiple JPEG (JPG) images into PDF format. This feature enables users to convert individual JPEG images into PDF documents, preserving the image's quality and details. It simplifies the process of sharing images in a universally readable format.

5. XLS to PDF

The application streamlines the process of converting Excel (XLS/XLSX) spreadsheets into PDF documents. This feature-rich application ensures that all data, formatting, and calculations from the original spreadsheets are faithfully preserved in the resulting PDF files.

6. Protect PDF

The application seamlessly secures/protects PDF files with password protection, ensuring sensitive documents remain confidential and accessible only to authorized users while simultaneously ensuring that all textual data and images remains preserved in the original form.

7. Unlock PDF

The application effortlessly unlocks password-protected PDF files, converting them back into standard PDFs. This ensures that users can access and manage their documents without restrictions while preserving all textual data and images in their original format.

8. PDF to JPG

The application seamlessly converts PDF files into JPG images, ensuring all textual content and images remain intact. Users can effortlessly convert their PDF documents into image format without compromising on clarity or formatting, making it easy to share, view, and manage.

9. Organise PDF

The application allows users to add or remove pages from their PDF files with ease. Users can effortlessly manage their PDF documents by inserting or deleting pages as needed, ensuring that their documents are perfectly tailored to their requirements without compromising.

10. Split PDF

The application enables users to split their PDF files into multiple smaller documents. Users can easily divide their PDF files by selecting specific pages or ranges, ensuring that each new document retains the original formatting and content, making it convenient to share and manage.

11. Extract PDF

The application allows users to extract specific pages from their PDF files. Users can select the pages they want to extract and save them as a new PDF document, ensuring that the formatting and content remain intact.

12. Compress PDF

The application enables users to compress PDF files, reducing their size without compromising the quality of the content. This feature helps in managing storage space and facilitates easier sharing and faster uploading of documents, ensuring that all textual content and images remain clear and intact.

13. Watermark PDF

The application enables users to add watermark to the PDF files without compromising the quality of the content. It provides users with the option to choose between various fonts, font colors and font size, simultaneously ensuring that all textual content and images remain clear and intact.

14. Digital Sign PDF

The application allows users to digitally sign PDF files, maintaining the integrity and quality of the content ensuring that the digital signature is applied securely and accurately, incorporating details such as the organization name, time, and date at the bottom-left of the page.

Other Features supported by the application –

1. Login and authentication -

To ensure secure access and maintain the confidentiality of sensitive data, the application incorporates a robust login and authentication system. Users must authenticate themselves using a secure login mechanism before accessing the application. This process involves validating user credentials against a stored database to ensure that only authorized personnel can use the application.

2. Dashboard –

The application features a user-friendly dashboard that serves as the central hub for accessing and viewing relevant information. The dashboard displays an overview of recent activities, including recent file conversions, uploads, and other relevant operations. This helps users quickly access their recent tasks. The dashboard also includes visualizations that provide insights into usage statistics, conversion trends, and other key metrics.

3. Management Information System –

The application features an integrated Management Information System (MIS) that provides administrators and managers with comprehensive tools for monitoring and managing the application's usage and performance. The MIS allows for the monitoring of user activities, including file conversions and other operations performed within the application. This helps in tracking usage patterns and identifying any anomalies.

The Management Information System fetches data directly from the database providing the feature to sort the data month wise and download it in the format of csv file if necessary.

Section II, Platforms & Libraries

Visual Studio IDE

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE typically consists of:

Visual Studio IDE for Python –

Visual Studio is a powerful and versatile IDE developed by Microsoft. While it supports a wide range of programming languages and development activities, it also provides robust features specifically tailored for Python development. Here's an overview of how Visual Studio supports Python development:

- **Python Development Environment:** Visual Studio provides a fully integrated Python development environment that includes all the necessary tools for writing, testing, and debugging Python code. It supports both CPython and IronPython, allowing developers to choose the Python interpreter that best suits their needs.
- **IntelliSense:** Visual Studio's IntelliSense feature offers advanced code completion, parameter info, quick info, and member lists. This helps in writing code faster and with fewer errors by providing suggestions and auto-completions based on the code context.
- **Debugging:** Visual Studio offers a powerful debugger that allows developers to set breakpoints, step through code, inspect variables, and evaluate expressions. The debugger supports both local and remote debugging, making it easier to identify and fix issues in Python applications.
- **Virtual Environments:** Visual Studio supports the creation and management of Python virtual environments. This is crucial for isolating dependencies and ensuring that projects have the correct packages and versions installed.
- **Package Management:** Visual Studio integrates with Python's package management systems, such as pip and conda. Developers can easily install, update, and manage Python packages and libraries directly from the IDE.

In summary, Visual Studio provides a comprehensive, feature-rich IDE for Python development, making it easier for developers to write, debug, and deploy Python applications efficiently.

Python Streamlit

Streamlit is an open-source Python library designed to simplify the creation of interactive web applications for data science and machine learning projects. Launched in 2019, Streamlit aims to provide an easy-to-use framework that allows developers to build and share web apps with minimal effort and without requiring extensive knowledge of web development technologies.

Key Features of streamlit –

- 1. Ease of Use:** Streamlit's core strength lies in its simplicity. Developers can create web applications with just a few lines of Python code, leveraging familiar data science libraries such as Pandas, NumPy, and Matplotlib. The straightforward API allows users to convert scripts into interactive web apps seamlessly.
- 2. Real-Time Interactivity:** Streamlit apps automatically update in real-time as users interact with the interface. This interactivity is achieved with minimal effort, allowing developers to create dynamic applications that respond instantly to user inputs.
- 3. Components and Layouts:** Streamlit provides a range of built-in components, such as sliders, buttons, and text inputs, that facilitate the creation of interactive and user-friendly interfaces. Layout options, such as columns and expandable containers, enable developers to organize app content effectively.
- 4. Deployment:** Streamlit applications can be easily deployed to the web, either using Streamlit's own deployment service or by deploying on cloud platforms such as AWS, Heroku, or Google Cloud. This flexibility ensures that applications are accessible from anywhere with an internet connection.
- 5. Custom Components:** While Streamlit provides a wide range of built-in components, it also supports the creation and integration of custom components. This feature allows developers to extend the functionality of their applications by incorporating additional interactive elements and features.

How It Works -

Streamlit operates as a lightweight server that automatically handles updates and interactions between the frontend and backend of the application. When a user interacts with the app (e.g., by clicking a button or adjusting a slider), Streamlit re-runs the entire Python script to update the app's state and render the new results. This approach simplifies the development process by eliminating the need for complex front-end development and state management.

Python Libraries

To perform the various functionalities of Nirmal PDF Converter, such as conversion between different file formats and performing PDF operations, the application utilizes a set of robust and up-to-date Python libraries. These libraries are open-source and freely accessible to developers and users. By leveraging these libraries, the application ensures reliable and efficient processing of documents while benefiting from the continuous improvements and community support associated with open-source software.

Here are the libraries which were utilized to achieve the various functionalities –

- 1. docx2pdf:** The docx2pdf library facilitates the conversion of Microsoft Word documents (.docx) into PDF files. This is particularly useful in scenarios where standardized document formats are needed, such as in reporting and documentation workflows. By converting Word documents to PDF, users can ensure that formatting remains consistent across different platforms and devices, while also making documents more secure and less prone to accidental edits.
- 2. pywin32:** The pywin32 library provides Python bindings for the Windows API, enabling interactions with various Windows services and applications. It is commonly used for automating tasks in Microsoft Office products like Word and Excel. For example, it allows Python scripts to manipulate Excel workbooks, automate report generation, and interact with COM objects. This library is crucial for integrating Python scripts into existing Windows-based workflows and automating repetitive tasks.
- 3. PyMuPDF:** PyMuPDF (also known as Fitz) is a Python binding for the MuPDF library, which provides powerful PDF processing capabilities. It supports reading, writing, and modifying PDF files and is capable of extracting text, images, and metadata. PyMuPDF also supports rendering pages to images and handling various document formats, making it a versatile tool for detailed PDF analysis, extraction, and manipulation.
- 4. PyPDF2:** PyPDF2 is a Python library that focuses on basic PDF file manipulation. It allows users to merge multiple PDFs into one, split a single PDF into several files, rotate pages, and extract text from PDFs. It is useful for creating custom PDFs and performing

standard PDF operations, although it does not support advanced features like form handling or encryption.

5. **pdf2image:** The pdf2image library converts PDF documents into image files, such as PNG or JPEG. This transformation is useful for creating image previews of PDF pages, embedding PDF content in web pages, or performing image-based analysis on PDF documents. The library is designed to handle the conversion process efficiently, preserving the quality of the original PDF content.
6. **Pillow:** Pillow is an extensive library for image processing in Python. It supports a wide range of image file formats and provides capabilities for opening, editing, and saving images. With Pillow, users can perform operations such as resizing, cropping, rotating, filtering, and drawing on images. It is widely used for tasks that require manipulation or enhancement of image files.
7. **fpdf:** The fpdf library is used to generate PDF files programmatically. It allows for the creation of new PDF documents from scratch, with support for adding text, images, shapes, and custom formatting. It is particularly useful for generating reports, invoices, and other documents where automated PDF creation is required.
8. **openpyxl:** Openpyxl provides tools for reading and writing Excel files in the .xlsx format. It allows for detailed manipulation of Excel workbooks, including accessing and modifying cell values, formatting sheets, and handling formulas. This library is useful for automating Excel-based workflows and generating complex reports from data stored in spreadsheets.
9. **xlrd:** The xlrd library enables reading data from older Excel file formats (.xls). It provides functionality for extracting cell data from Excel spreadsheets and is often used in data extraction tasks. However, it does not support writing to Excel files or handling newer .xlsx formats, limiting its use to legacy Excel files.
10. **fitz:** The fitz library, another Python binding for MuPDF, provides similar functionalities to PyMuPDF. It allows for detailed manipulation and analysis of PDF files, including text extraction, image extraction, and rendering. FitZ is commonly used for tasks that involve handling or modifying PDF content.

- 11. pyodbc:** PyODBC is a library that facilitates database connections and interactions using the ODBC interface. It enables Python applications to connect to a wide variety of SQL databases, execute SQL queries, and fetch results. It is essential for integrating Python applications with databases and performing data retrieval and manipulation tasks.
- 12. matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It supports a wide range of chart types, including line plots, bar charts, scatter plots, and histograms. It is highly customizable, making it suitable for producing publication-quality graphs and visualizations.
- 13. plotly:** Plotly is a library for creating interactive, web-based visualizations. It supports a variety of chart types, including 3D plots and maps, and provides interactive features such as zooming, panning, and tooltips. Plotly's focus on interactivity and visual appeal makes it ideal for building dashboards and exploratory data analysis tools.
- 14. altair:** Altair is a declarative visualization library that simplifies the creation of complex charts through a concise, intuitive syntax. It uses a JSON-based specification to define visualizations, making it easy to create interactive and informative plots with minimal code. Altair is designed for creating clear and effective visualizations for data analysis.
- 15. pdfplumber:** Pdfplumber is a specialized library for extracting text, tables, and metadata from PDF documents. It provides advanced capabilities for parsing structured data from PDFs, making it suitable for tasks that require detailed analysis and extraction of content from complex PDF layouts.
- 16. pyhanko:** PyHanko is a library focused on digital signatures for PDF documents. It allows users to add digital signatures, stamps, and other authentication marks to PDFs, ensuring the integrity and authenticity of the documents. PyHanko supports various signing methods and customization options, making it ideal for secure document handling.

SQL Express Database

Nirmal PDF Converter not only excels in converting files and performing various PDF operations but also incorporates advanced features for managing user data and metadata. The application provides functionalities such as User Authentication, Management Information System (MIS) and Dashboard.

To support these functionalities, Nirmal PDF Converter utilizes Microsoft SQL Server Express. SQL Server Express is a free, lightweight version of Microsoft SQL Server that provides robust database management capabilities. It is designed for small to medium-sized applications and includes:

- 1. Database Storage:** SQL Server Express stores and manages the application's user data, file metadata, and other related information. It ensures that all data is securely stored and readily accessible.
- 2. Data Integrity and Security:** SQL Server Express offers features to ensure data integrity and security, including transactional support and access controls. This helps maintain the accuracy and confidentiality of the data managed by Nirmal PDF Converter.
- 3. Scalability and Performance:** While SQL Server Express is a scaled-down version of SQL Server, it still provides essential database features and performance optimizations. It supports efficient querying and data management.
- 4. Data Backup and Recovery:** SQL Server Express includes features for backing up and restoring databases, ensuring data protection and recovery in case of hardware failures or data loss. Regular backups can be scheduled to safeguard critical information.
- 5. Database Management Tools:** SQL Server Express comes with tools like SQL Server Management Studio (SSMS) that provide a graphical interface for database management, query execution, and performance monitoring. These tools simplify database administration and help ensure smooth operation.

- 6. Reporting and Analytics:** It supports basic reporting and data analysis functionalities, allowing users to generate reports and gain insights from the stored data. This is beneficial for creating management reports and visualizing data trends.
- 7. Integration with Other Microsoft Technologies:** SQL Server Express integrates seamlessly with other Microsoft technologies, such as Excel and Power BI. This integration enables users to import, export, and analyze data from SQL Server Express using familiar tools.
- 8. Support for Advanced Features:** Even though it is a lightweight version, SQL Server Express supports advanced database features such as stored procedures, triggers, and views, which are essential for implementing complex logic.

By integrating SQL Server Express, Nirmal PDF Converter ensures a reliable and scalable solution for managing user authentication, file metadata, and application data, while providing a seamless and secure experience for its users

SQL Server Management Studio (SSMS) is an integrated environment provided by Microsoft for managing and administering SQL Server databases. Nirmal PDF Converter leverages SSMS to effectively interact with SQL Server Express for database management and operations. Here's how SSMS is used within the context of Nirmal PDF Converter:

- 1. Database Creation and Configuration:** SSMS is used to create and configure the database that Nirmal PDF Converter uses. This involves setting up the database schema, including tables for user credentials, file metadata, and other relevant information. SSMS provides a graphical interface for defining database structures and relationships, making it easier to organize and manage data.
- 2. Query Execution:** SSMS allows for the execution of SQL queries to retrieve, insert, update, or delete data within the SQL Server Express database. Nirmal PDF Converter uses these queries to perform various operations, such as validating user credentials during login, storing file metadata after conversions, and generating reports for the management information system (MIS).

- 3. Data Management and Maintenance:** SSMS provides tools for managing and maintaining the database. This includes tasks such as creating and managing indexes to optimize query performance, backing up and restoring databases to ensure data safety, and monitoring database performance. These capabilities ensure that Nirmal PDF Converter operates smoothly and that data integrity is maintained.
- 4. Security and Access Control:** Through SSMS, administrators can configure security settings and access controls for the database. This involves defining user roles and permissions to control who can access and modify different parts of the database. Nirmal PDF Converter uses these security features to protect sensitive information, such as user credentials and file metadata, from unauthorized access.
- 5. Monitoring and Troubleshooting:** SSMS offers tools for monitoring database activity and troubleshooting issues. Administrators can use SSMS to review database logs, track performance metrics, and identify potential problems. This helps ensure that Nirmal PDF Converter remains reliable and that any issues are addressed promptly.
- 6. Database Design and Reporting:** SSMS enables users to design and visualize database schemas and create reports. For Nirmal PDF Converter, this functionality is useful for designing the database schema to meet the application's requirements and generating reports that provide insights into user activity and file processing.
- 7. Stored Procedures:** SSMS enable users to create Stored Procedures. Stored procedures are a collection of SQL statements that are saved and stored in the database. Once created, they can be executed (or "called") multiple times without needing to rewrite the SQL code. They are useful for encapsulating complex logic, performing repetitive tasks, and managing data in a controlled manner.

In summary, SQL Server Management Studio (SSMS) plays a crucial role in the development and maintenance of the database infrastructure for Nirmal PDF Converter. It provides a comprehensive suite of tools for database management, query execution, security, and performance monitoring, ensuring that the application's data operations are efficient, secure, and well-maintained.

Section III, Implementation – File Conversions

Excel to PDF

In Nirmal, the xlrld library is utilized to read data from Excel files. This library allows the extraction of cell values from various sheets in an Excel workbook, which is essential for converting Excel content into a PDF format.

By leveraging xlrld for reading Excel files for generating PDFs, the function “convert_xls_to_pdf” effectively converts spreadsheet data into a well-formatted PDF document, handling multiple pages and ensuring that content is presented clearly.

```
def convert_xls_to_pdf(excel_file, columns_per_page):
    workbook = xlrld.open_workbook(file_contents=excel_file.read())
    pdf = FPDF()

    for sheet_name in workbook.sheet_names():
        worksheet = workbook.sheet_by_name(sheet_name)
        num_cols = worksheet.ncols
        num_rows = worksheet.nrows
        num_pages = (num_cols + columns_per_page - 1) // columns_per_page

        for page_num in range(num_pages):
            pdf.add_page()
            pdf.set_font("Arial", size=12)

            pdf.set_left_margin(10)
            pdf.set_right_margin(10)

            # Determine columns to include on this page
            start_col = page_num * columns_per_page
            end_col = min(start_col + columns_per_page, num_cols)

            # Iterate through rows and columns for the selected range
            for row in range(num_rows):
                for col in range(start_col, end_col):
                    cell_value = worksheet.cell_value(row, col)
                    if cell_value is not None:
                        pdf.cell(38, 11, str(cell_value), border=1)
            pdf.ln()
```

JPG to PDF

In the Nirmal PDF Converter application, the PIL (Pillow) library is employed to handle image files (.jpg) and convert them into PDF format using the FPDF library. The provided code snippet demonstrates this process.

The function begins by creating a temporary PDF file using `tempfile.NamedTemporaryFile()`. This file will store the resulting PDF and is managed with the .pdf suffix.

It initializes a new PDF document with `FPDF()` and configures automatic page breaks with a margin of 15 mm to ensure content does not overlap the page borders.

```
try:
    with tempfile.NamedTemporaryFile(delete=False, suffix=".pdf")
        as temp_pdf:
            temp_pdf_path = temp_pdf.name

            pdf = FPDF()
            pdf.set_auto_page_break(auto=True, margin=15)

            image = Image.open(temp_jpg_path)

            pdf.add_page()

            img_width_mm, img_height_mm = pdf.w - 2 * pdf.l_margin,
            pdf.h - 2 * pdf.b_margin
            img_width, img_height = image.size
            aspect_ratio = img_width / img_height

            # Determine orientation based on aspect ratio
            if aspect_ratio >= 1:
                pdf.image(temp_jpg_path, x=pdf.l_margin,
                    y=pdf.t_margin, w=img_width_mm)
            else:
                pdf.image(temp_jpg_path, x=pdf.l_margin,
                    y=pdf.t_margin, h=img_height_mm)

            # Close the image
            image.close()

            # Save the PDF document
            pdf.output(temp_pdf_path)
```

PDF to JPG

Nirmal leverages the pdf2image library to transform each page of the PDF into an image, utilizing poppler as the underlying rendering tool. poppler must be installed on the system and its path specified in the function. The images produced are processed by the Python Imaging Library (PIL), which handles saving these images in JPG format.

The “convert_pdf_to_jpg” function creates a temporary PDF file to hold the uploaded document, converts it to images, and then saves each image as a JPG file, returning a list of these JPG images for further use.

```
def convert_pdf_to_jpg(pdf_file):
    with tempfile.NamedTemporaryFile(delete=False, suffix=".pdf") as temp_pdf:
        temp_pdf.write(pdf_file.read())
        temp_pdf_path = temp_pdf.name

    images = convert_from_path(temp_pdf_path, poppler_path=r'C:\Program Files
(x86)\poppler-24.02.0\Library\bin')

    jpg_images = []
    for i, image in enumerate(images):
        img_byte_arr = io.BytesIO()
        image.save(img_byte_arr, format='JPEG')
        img_byte_arr = img_byte_arr.getvalue()
        jpg_images.append((f"page_{i + 1}.jpg", img_byte_arr))
    return jpg_images
```

```
if is_pdf(uploaded_file):
    if st.button("Convert to JPG"):
        st.info("Download the Converted JPG File(s)")
        try:
            jpg_files = convert_pdf_to_jpg(uploaded_file)
            for filename, jpg in jpg_files:
                st.download_button(
                    label=f"Download {filename}",
                    data=jpg,
                    file_name=filename,
                    mime="image/jpeg"

                )
```

PNG to PDF

In the Nirmal PDF Converter application, the PIL (Pillow) library is employed to handle image files (.png) and convert them into PDF format using the FPDF library. The provided code snippet demonstrates this process.

The function begins by creating a temporary PDF file using `tempfile.NamedTemporaryFile()`. This file will store the resulting PDF and is managed with the `.pdf` suffix.

It initializes a new PDF document with `FPDF()` and configures automatic page breaks with a margin of 15 mm to ensure content does not overlap the page borders.

```
if temp_png_paths:
    try:
        with tempfile.NamedTemporaryFile(delete=False, suffix=".pdf") as
            temp_pdf:
                temp_pdf_path = temp_pdf.name
                pdf = FPDF()
                pdf.set_auto_page_break(auto=True, margin=15)

                for temp_png_path in temp_png_paths:
                    image = Image.open(temp_png_path)

                    pdf.add_page()

                    img_width_mm, img_height_mm = pdf.w - 2 * pdf.l_margin,
                    pdf.h - 2 * pdf.b_margin
                    img_width, img_height = image.size
                    aspect_ratio = img_width / img_height

                    if aspect_ratio >= 1:
                        pdf.image(temp_png_path, x=pdf.l_margin,
                                y=pdf.t_margin, w=img_width_mm)
                    else:
                        pdf.image(temp_png_path, x=pdf.l_margin,
                                y=pdf.t_margin, h=img_height_mm)

                    # Close the image
                    image.close()

                    # Save the PDF document
                    pdf.output(temp_pdf_path)
```

Word to PDF

Nirmal utilizes the `pythoncom` and `word2pdf` libraries to facilitate the conversion of Word documents to PDF. The `pythoncom` library is used to handle COM objects, enabling interaction with Microsoft Word via its automation API, which is essential for file conversions on Windows. The `word2pdf` library provides a straightforward method to convert DOCX files into PDF format.

The uploaded DOCX file is saved temporarily using `tempfile.NamedTemporaryFile`. This temporary file is used for the conversion process. The `convert` function from the `word2pdf` library is used to convert the DOCX file to a PDF, which is also saved as a temporary file.

```
if uploaded_file is not None:
    file_size = len(uploaded_file.getvalue())
    docx_magic_number = b"\x50\x4B\x03\x04"
    first_bytes = uploaded_file.read(len(docx_magic_number))
    uploaded_file.seek(0)

    if first_bytes == docx_magic_number:
        st.success("Word Document uploaded successfully!")
        with tempfile.NamedTemporaryFile(delete=False, suffix=".docx") as temp_docx:
            temp_docx.write(uploaded_file.read())
            temp_docx_path = temp_docx.name
        with tempfile.NamedTemporaryFile(delete=False, suffix=".pdf") as temp_pdf:
            temp_pdf_path = temp_pdf.name

        convert(temp_docx_path, temp_pdf_path)

        with open(temp_pdf_path, "rb") as f:
            st.download_button(
                label="Download PDF",
                data=f,
                file_name="converted.pdf",
                mime="application/pdf"
            )

        os.remove(temp_docx_path)
        os.remove(temp_pdf_path)
    else:
        st.warning("Please upload a valid DOCX file.")
```


Section IV, PDF Operations

Compress PDF

Nirmal PDF Converter uses the Ghostscript library to compress PDF files effectively by leveraging its powerful PostScript and PDF processing capabilities. Ghostscript provides various options to optimize PDF content, including compressing images, library offers several compression modes, including default, maximum, low etc.

To use Ghostscript, it must be installed on the system. After installation, Ghostscript should be accessible from the system's PATH, allowing Nirmal to invoke it seamlessly through Python code.

```
def compress(input_file_path, output_file_path, power=0):
    quality = {
        0: "/default",
        1: "/prepress",
        2: "/printer",
        3: "/ebook",
        4: "/screen"
    }
    gs = get_ghostscript_path()
    st.info("Compressing PDF ...")
    initial_size = os.path.getsize(input_file_path)
    subprocess.call([
        gs,
        "-sDEVICE=pdfwrite",
        "-dCompatibilityLevel=1.4",
        "-dPDFSETTINGS={}".format(quality[power]),
        "-dNOPAUSE",
        "-dQUIET",
        "-dBATCH",
        "-sOutputFile={}".format(output_file_path),
        input_file_path,
    ])
    final_size = os.path.getsize(output_file_path)
    ratio = 1 - (final_size / initial_size)
    st.success(f"The PDF was compressed to {initial_size / 1024:.2f} KB with  
Compression ratio: {ratio:.2%}.")

    return output_file_path
```

Extract PDF

Nirmal utilizes the PyPDF2 library to extract specific pages from a PDF file by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

The `extract_pdf_pages` function in Nirmal uses PyPDF2 to extract specified pages from a PDF document. It initializes a PdfReader to read the input PDF and then iterates over the requested page numbers. For each valid page, it creates a new PdfWriter, adds the page to it, and saves the page to a bytes buffer. The function collects all extracted pages into a list, which can then be returned for further processing or download by the user.

```
def extract_pdf_pages(input_pdf, page_numbers):
    pdf_reader = PdfReader(input_pdf)
    extracted_pages = []

    for page_num in page_numbers:
        if 1 <= page_num <= len(pdf_reader.pages):
            pdf_writer = PdfWriter()
            pdf_writer.add_page(pdf_reader.pages[page_num - 1])

            # Save to a bytes buffer
            pdf_buffer = io.BytesIO()
            pdf_writer.write(pdf_buffer)
            pdf_buffer.seek(0)
            extracted_pages.append((f'Page_{page_num}.pdf', pdf_buffer))
        else:
            st.warning(f"Page number {page_num} is out of range.")

    return extracted_pages
```

“`io.BytesIO()`” is a class in Python’s `io` module that provides a memory buffer for binary data. It's used to create an in-memory stream that can be read from and written to like a file, but without the need for actual file I/O operations.

Organize PDF

Nirmal utilizes the PyPDF2 library to organize pages of a PDF file (Adding pages to a PDF and removing pages from a PDF) by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

The “**add_pdf_pages**” function is used to add a page at the end of the input PDF.

```
def add_pdf_pages(input_pdf, pages_to_add):
    pdf_reader = PdfReader(input_pdf)
    pdf_writer = PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        pdf_writer.add_page(pdf_reader.pages[page_num])

    for page_data in pages_to_add:
        pdf_writer.add_page(page_data)

    pdf_buffer = io.BytesIO()
    pdf_writer.write(pdf_buffer)
    pdf_buffer.seek(0)

    return pdf_buffer
```

The “**delete_pdf_pages**” function is used to add a page at the end of the input PDF.

```
def delete_pdf_pages(input_pdf, pages_to_delete):
    pdf_reader = PdfReader(input_pdf)
    pdf_writer = PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        if page_num + 1 not in pages_to_delete:
            pdf_writer.add_page(pdf_reader.pages[page_num])
    pdf_buffer = io.BytesIO()
    pdf_writer.write(pdf_buffer)
    pdf_buffer.seek(0)

    return pdf_buffer
```

Protect PDF

Nirmal utilizes the PyPDF2 library to Protect a PDF by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

To set a password using PyPDF2, the process begins with reading an existing PDF file using the PdfReader class. After loading the PDF, a PdfWriter object is created to manage modifications. Pages are added to the PdfWriter using the add_page() method. Once the desired pages are included, the encrypt() method of PdfWriter is used to apply a password to the document. This method allows specifying both a user password and an owner password to secure the PDF file.

```
def protect_pdf(uploaded_file, output_path, password):
    reader = PdfReader(uploaded_file)
    writer = PdfWriter()

    for page in reader.pages:
        writer.add_page(page)
    writer.encrypt(password)

    with open(output_path, 'wb') as output_file:
        writer.write(output_file)
```

The “protect_pdf” function is later called in code to encrypt the PDF and the user is prompted to enter a password.

```
if password:
    protected_file_path = "protected_pdf.pdf"
    protect_pdf(uploaded_file, protected_file_path, password)
    st.success(f"PDF protected successfully!")
    with open(protected_file_path, 'rb') as f:
        pdf_bytes = f.read()
    st.download_button(label="Download Protected PDF",
                      data=pdf_bytes, file_name="protected_pdf.pdf")
else:
    st.warning("Please enter a password.")
```

Sign PDF

Nirmal PDF Converter utilizes the pyhanko library to sign PDFs, ensuring the integrity and authenticity of documents. PyHanko is a library designed for digital signing and handling of PDFs in Python. It offers functionality for creating and validating digital signatures, ensuring document integrity and authenticity.

* A **PKCS#12** file is a secure container format used to store and transport cryptographic keys and certificates. It commonly contains a private key along with its associated public key certificate and optionally, a chain of certificates

```
def sign_pdf(p12_file, password, pdf, add_footer=False):
    try:
        # Create a temporary file for the PKCS#12 data
        with tempfile.NamedTemporaryFile(delete=False, suffix='.p12') as temp_p12_file:
            temp_p12_file.write(p12_file.read())
            temp_p12_path = temp_p12_file.name

        # Load the PKCS#12 file with the signer information
        signer = signers.SimpleSigner.load_pkcs12(
            pfx_file=temp_p12_path, passphrase=password.encode('utf-8')
        )

        # Read the PDF data
        pdf_data = pdf.read()

        # Open the PDF document
        pdf_stream = BytesIO(pdf_data)
        w = IncrementalPdfFileWriter(pdf_stream)

        # Sign the PDF
        signed_pdf_stream = BytesIO()
        signers.sign_pdf(
            w, signers.PdfSignatureMetadata(field_name='Signature1'),
            signer=signer,
        )

        # Save the signed PDF
        w.write(signed_pdf_stream)
        signed_pdf_stream.seek(0)
```

Watermark PDF

Nirmal uses the PIL library's Image, ImageDraw, and ImageFont modules to create and apply watermarks to PDF files. The create_watermark_image function generates a watermark image with customizable text, font size, color, and rotation.

```
def create_watermark_image(text, width, height, font_path, font_size,
font_color):

    # Create an image with transparent background
    watermark_image = Image.new("RGBA", (width, height), (0, 0, 0, 0))
    draw = ImageDraw.Draw(watermark_image)

    # Load the selected font
    try:
        font = ImageFont.truetype(font_path, font_size)
    except IOError:
        font = ImageFont.load_default()

    # Calculate text size and position
    bbox = draw.textbbox((0, 0), text, font=font)
    text_width = bbox[2] - bbox[0]
    text_height = bbox[3] - bbox[1]
    x = (width - text_width) / 2
    y = (height - text_height) / 2

    # Draw the text onto the image
    draw.text((x, y), text, font=font, fill=font_color)

    # Rotate the image
    watermark_image = watermark_image.rotate(45, expand=1)

    # Save the image to a temporary file
    with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmp_file:
        watermark_image.save(tmp_file, format="PNG")
        tmp_file.seek(0)
    return tmp_file.name
```

The ImageDraw.Draw() method is used to draw text onto the image. After drawing the text, the watermark image is rotated by 45 degrees for a diagonal effect. The resulting image is saved as a PNG file to a temporary location using tempfile.NamedTemporaryFile(), which is then returned for use in watermarking PDFs.

Unlock PDF

Nirmal utilizes the PyPDF2 library to unlock a PDF by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

The PyPDF2 library provides functionality for unlocking encrypted PDF files. Using PyPDF2, one can read an encrypted PDF with PdfReader, and if the PDF is encrypted, it can be decrypted using the decrypt() method by providing the correct password. Once decrypted, a new PdfWriter object is created to handle the extraction and rewriting of the PDF pages. The decrypted pages are added to the PdfWriter, which then writes the pages to a BytesIO buffer.

```
def unlock_pdf(uploaded_file, password):
    reader = PdfReader(uploaded_file)

    if reader.is_encrypted:
        reader.decrypt(password)
        writer = PdfWriter()

        for page in reader.pages:
            writer.add_page(page)

        output_pdf = BytesIO()
        writer.write(output_pdf)
        output_pdf.seek(0)

        return output_pdf
    else:
        return None
```

Split PDF

Nirmal utilizes the PyPDF2 library to split pages of a PDF by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

The code defines a function `split_pdf_by_range` that splits a PDF document into separate files based on specified page ranges. It begins by reading the input PDF with PdfReader. For each range provided, it creates a new PdfWriter instance to handle the extraction and writing of pages. Pages within the specified range are added to this writer. The resulting PDF content is saved into a BytesIO buffer, which is then appended to a list of split PDFs.

```
def split_pdf_by_range(input_pdf, ranges):
    pdf_reader = PdfReader(input_pdf)
    split_pdfs = []

    for start, end in ranges:
        pdf_writer = PdfWriter()
        for page_num in range(start - 1, end):

            pdf_writer.add_page(pdf_reader.pages[page_num])

        # Save to a bytes buffer
        pdf_buffer = io.BytesIO()
        pdf_writer.write(pdf_buffer)
        pdf_buffer.seek(0)
        split_pdfs.append((f'Pages_{start}_to_{end}.pdf', pdf_buffer))
```


Merge PDF

Nirmal utilizes the PyPDF2 library to merge two or more PDF files by using its PdfReader and PdfWriter classes. The PdfReader class reads the input PDF, while the PdfWriter class is used to create new PDFs containing only the requested pages.

The merge_pdfs function is designed to consolidate multiple PDF files into a single, cohesive document. It begins by initializing a PdfWriter object, which is responsible for creating and managing the output PDF. The function iterates through each provided PDF file, using a PdfReader object to read the contents.

For each PDF, it loops through all its pages and adds them to the PdfWriter, effectively combining all pages from the input PDFs. To handle the output, the function creates a temporary file with a .pdf suffix using tempfile.NamedTemporaryFile, ensuring that the merged PDF is stored securely without leaving permanent files on the system.

```
def merge_pdfs(pdf_files):
    pdf_writer = PdfWriter()
    for pdf_file in pdf_files:
        pdf_reader = PdfReader(pdf_file)
        for page_num in range(len(pdf_reader.pages)):
            page = pdf_reader.pages[page_num]
            pdf_writer.add_page(page)

    merged_pdf_file = tempfile.NamedTemporaryFile(delete=False,
        suffix=".pdf").name
    with open(merged_pdf_file, "wb") as output_pdf:
        pdf_writer.write(output_pdf)

    return merged_pdf_file
```

Section V, Backend

Database Connectivity

As discussed in the previous section Nirmal uses MS SQL express database to store the “User Login” & “File Meta Data” as well as to verify user credentials.

The database is connected using the connect_to_db function, which establishes a connection with the PowergridDB database. This function utilizes pyodbc.connect to establish a connection with the PowergridDB database. Various information about the database is specified in the string such as the Driver, Server (Localhost when hosting it locally), Database Name, User ID, Password and the Encryption Status.

```
def connect_to_db():
    conn = pyodbc.connect(
        "Driver={ODBC Driver 18 for SQL Server};"
        "Server=Z3PHYR\\SQLEXPRESS;"
        "Database=Confidential;"
        "UID=admin;"
        "PWD=powergrid;"
        "Encrypt=no;"
    )
    return conn
```

```
def check_credentials(username, password):
    conn = connect_to_db()
    cursor = conn.cursor()
    cursor.execute("{CALL sp_AuthenticateUser (?)}", username)
    result = cursor.fetchone()
    conn.close()
    if result:
        stored_password = result[0]
        return stored_password == password
    return False
```

The sp_AuthenticateUser stored procedure is defined under the programmability section of the database. It is called using the statement {CALL sp_AuthenticateUser (?)}, where the question mark represents the placeholder for the parameter required by the stored procedure.

User Data

The `log_user_login` function records user login details in the database. It connects to the database using the `connect_to_db` function and, if successful, executes the `sp_InsertUserLogIn` stored procedure with the parameters: `emp_name`, `user_host_address`, `user_host_name`, and `logged_in`.

The function handles potential errors by catching `pyodbc.Error` for database-specific errors and general exceptions for other issues, displaying error messages using `st.error`. Finally, the database connection is closed in the `finally` block to ensure proper resource management.

```
def log_user_login(emp_name, user_host_address, user_host_name, logged_in):
    try:
        conn = connect_to_db()
        if conn:
            cursor = conn.cursor()
            cursor.execute("{CALL sp_InsertUserLogIn (?, ?, ?, ?)}",
                           emp_name, user_host_address, user_host_name, logged_in)
            conn.commit()
    except pyodbc.Error as e:
        st.error(f"Error logging user login: {e}")
    except Exception as ex:
        st.error(f"Unexpected error: {ex}")
    finally:
        if conn:
            conn.close()
```

```
if check_credentials(username, password):
    st.session_state.authenticated = True
    st.session_state.username = username
    st.success("Login successful!")
    log_user_login(
        emp_name=username,
        user_host_address=socket.gethostbyname(socket.gethostname()),
        user_host_name=socket.gethostname(),
        logged_in=datetime.datetime.now()
    )
    st.experimental_rerun()
else:
    st.error("Invalid username or password.")
```

File Meta Data

The `insert_file_data` function is designed to insert file-related data into a SQL Server database. It begins by establishing a connection to the database using the `connect_to_db` function.

The function converts data types as needed—ensuring the `updated_on` timestamp is in the correct format, and converting size and status to integers. It then executes a stored procedure, “`sp_InsertFile`”, with the given parameters to insert file details such as name, description, and status into the database.

If any errors occur during this process, they are handled gracefully with detailed error messages, and the database connection is closed in the end. This ensures that file metadata is accurately recorded and any issues are communicated effectively.

```
def insert_file_data(name, description, status, content_type, size,
                    updated_by, updated_on, updated_from):
    try:
        conn = connect_to_db()
        if conn:
            cursor = conn.cursor()
            updated_on_sql = updated_on.strftime('%Y-%m-%d %H:%M:%S.%f')[:-3]
            if isinstance(updated_on, datetime.datetime) else updated_on
            size = int(size)
            status = int(status)
            cursor.execute("{CALL sp_InsertFile ( ?, ?, ?, ?, ?, ?, ?, ? )}",
                          (name, description, status, content_type, size,
                           updated_by, updated_on_sql, updated_from))

            conn.commit()

    except pyodbc.Error as e:
        st.error(f"Error inserting file data: {e}")
        st.write("SQL Error:", e)
        st.write("SQL State:", e.args[0])
        st.write("SQL Message:", e.args[1])
    except Exception as ex:
        st.error(f"Unexpected error: {ex}")
    finally:
        if conn:
            conn.close()
```

Section VI, Security Features

Login & Authentication

The very first page that the user navigates to access the application is the “Nirmal Login & Authentication” page. On this page, users are prompted to enter their username and password. These credentials are then verified against Powergrid's database to ensure that only authorized users can access the application.

The Main logic for authentication is achieved by the initializing session state for authentication as false by using streamlit's “st.session_state” function:

```
if 'authenticated' not in st.session_state:
    st.session_state.authenticated = False
```

```
if not st.session_state.authenticated:
    username = st.text_input("Username")
    password = st.text_input("Password", type="password")

    if st.button("Login"):
        if username.strip() == "" or password.strip() == "":
            st.warning("Please enter both username and password.")
        else:
            if check_credentials(username, password):
                st.session_state.authenticated = True
                st.session_state.username = username
                st.success("Login successful!")
                st.experimental_rerun()
            else:
                st.error("Invalid username or password.")
    else:
        st.info("You are logged in!")
        col1, col2, col3 = st.columns(3)
        with col1:
            if st.button("Logout <🌐> "):
                st.session_state.authenticated = False
                st.success("Logout successful.")
                st.experimental_rerun()
```

File Signatures

File signatures, also known as magic numbers or magic bytes, are unique sequences of bytes located at the beginning of a file that identify its format or type. These signatures are used to determine the file's nature and ensure it is compatible with the intended application or process.

In the Nirmal PDF Converter application, file signatures are used to verify the format of uploaded files. By checking the initial bytes of the file against known signatures for different file types (e.g., PDF, JPG, DOCX), the application ensures that only files of the expected format are processed. This method helps prevent the upload and handling of potentially harmful or unsupported files.

This Function is used to check the initial bytes of a “xlsx” File.

```
def is_xlsx(file):  
    xlsx_magic_number = b'\x50\x4B\x03\x04'  
    first_bytes = file.read(4)  
    file.seek(0)  
  
    return first_bytes == xlsx_magic_number
```

This Function is used to check the initial bytes of a “png” File.

```
def is_png(file):  
    png_magic_number = b'\x89PNG\r\n\x1a\n'  
  
    first_bytes = file.read(8)  
  
    return first_bytes[:8] == png_magic_number
```

Another Example is checking if the file is actually “.pdf” by using the “is_pdf” function.

```
def is_pdf(file):  
    file.seek(0)  
    magic_number = file.read(4)  
    file.seek(0)    return magic_number == b'%PDF'
```

Stored Procedures

Stored procedures are precompiled SQL statements that are stored and executed on the database server. They encapsulate a set of SQL commands that can be reused and executed with different parameters.

In the Nirmal PDF Converter application, stored procedures are used to handle database operations securely. For instance, the “sp_AuthenticateUser” procedure is used to verify user credentials. By using stored procedures, the application mitigates the risk of SQL injection attacks. This is because stored procedures separate SQL code from user inputs, which are passed as parameters. As a result, user inputs are treated as data rather than executable code, preventing malicious code from being executed and enhancing overall security.

Below is the example of a stored procedure “sp_AuthenticateUser”

```
USE [PowergridDB]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

ALTER PROCEDURE [dbo].[sp_AuthenticateUser]
    @Username NVARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT Password
    FROM EmployeesInformation
    WHERE Username = @Username;
END
```

The sp_AuthenticateUser stored procedure is designed to retrieve a user's password based on their username. It is defined in the PowergridDB database and operates as follows:

- 1. Input Parameter:** @Username NVARCHAR(50) - The procedure accepts a username as input.
- 2. Operation:** The procedure executes a SELECT query to fetch the password associated with the provided username from the EmployeesInformation table.

This procedure allows secure retrieval of user credentials while helping to prevent SQL injection.

Section VII, User Interface

Login & Authentication

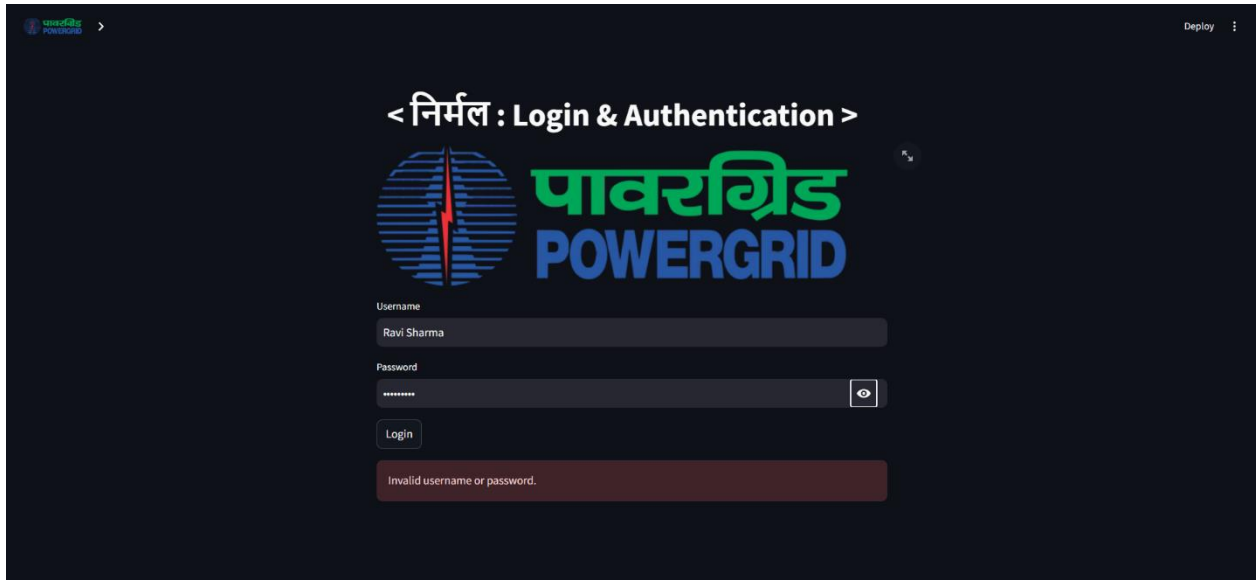
1. Log in to the Nirmal PDF Converter using the username and password parameter.



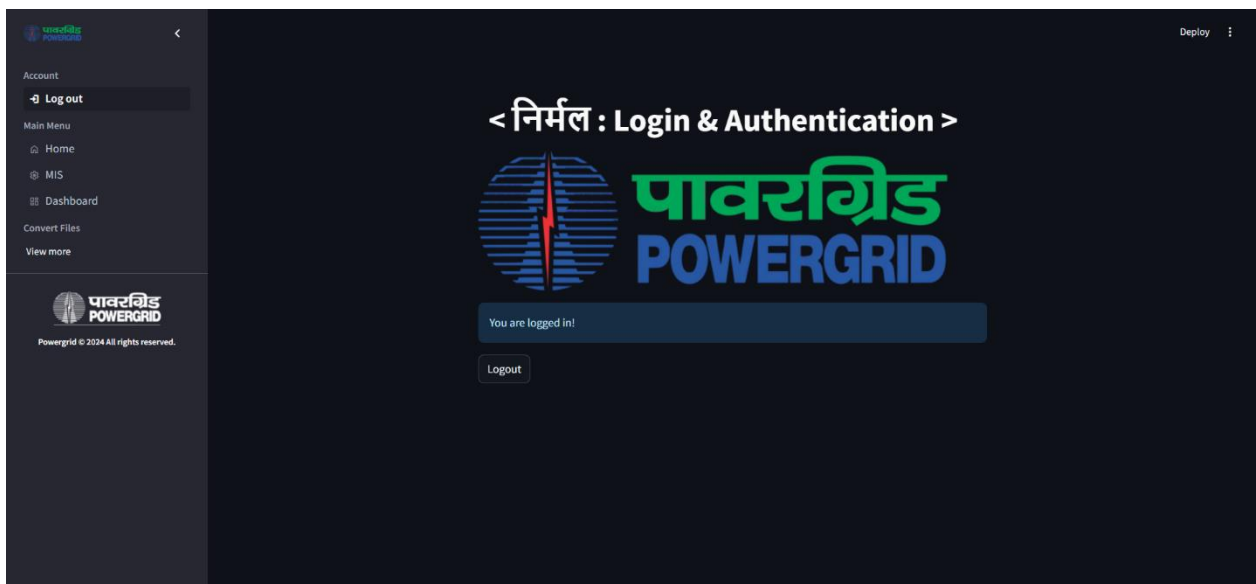
2. Users can toggle the visibility of their entered password by clicking on the eye icon, allowing them to hide or reveal the password as needed.



3. If a user input wrong login credentials then they may find the error “Invalid username and password” and are prompted to enter credentials again.



4. If a user input correct login credentials then they may find the message “login successful” with the side bar toggling to expanded state.



Navigation

1. Users can click on the "View More" option to explore all the services offered by the Nirmal PDF Converter.



2. The side bar provides a very easy and simple way to navigate between different services, a user can navigate just simply by clicking on each of the service.



Home Page

1. The very first section that is visible to the users is of “Main Menu” which contains three pages the Home Page, The Management Information System and the Dashboard.



MSI Page

1. The MIS Page shows the file Meta data and User login data fetching it from the database.

The screenshot displays the PowerGrid MIS interface. The left sidebar contains navigation links: Account, Log out, Main Menu, Home, MIS (selected), Dashboard, Convert Files, and View more. The main content area is titled "< निर्मल: Management Information System >" and features the PowerGrid logo. Below the title, the section is labeled "< File Meta Data >". A date range selector shows "2024/07/17 - 2024/07/17". A table lists file metadata with columns: FileId, Name, Description, Status, ContentType, Size, UpdatedBy, and Updi. The table contains six rows of data.

FileId	Name	Description	Status	ContentType	Size	UpdatedBy	Updi
0	1 testisi.pdf	PDF to JPG	1	.pdf	34,976	Ravi	2024
1	2 testisi.pdf	PDF to JPG	1	.pdf	34,976	Ravi	2024
2	3 Cyber Security Syllabus.jpg	JPG to PDF	1	.jpg	180,315	Ravi	2024
3	4 Cyber Security Syllabus.jpg	JPG to PDF	1	.jpg	180,315	Ravi	2024
4	5 AI Project Report - Converzo.docx	Word to PDF	1	.docx	45,314	Ravi	2024
5	6 RaviSharma'sResume.pdf	Compress PDF	1	.pdf	260,718	Ravi	2024

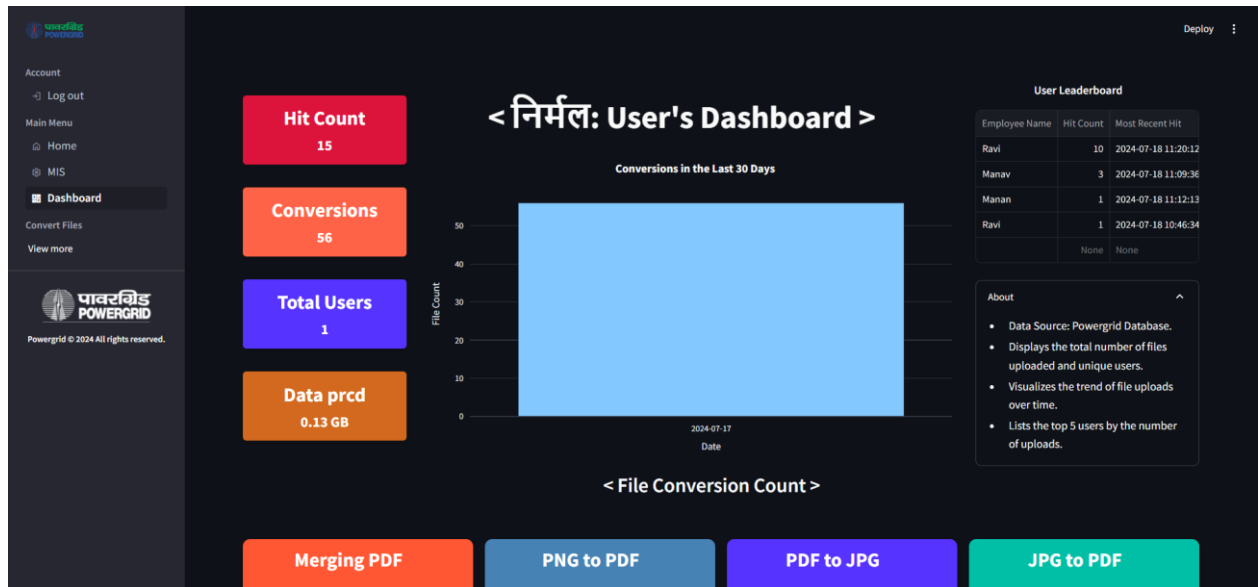
2. Clicking on the table prompts a download button which allows the users to download this data as the csv format.

This screenshot shows the same PowerGrid MIS interface as the previous one, but with an additional "Download as CSV" button located above the table. The button includes a download icon and a search icon. The table data remains the same.

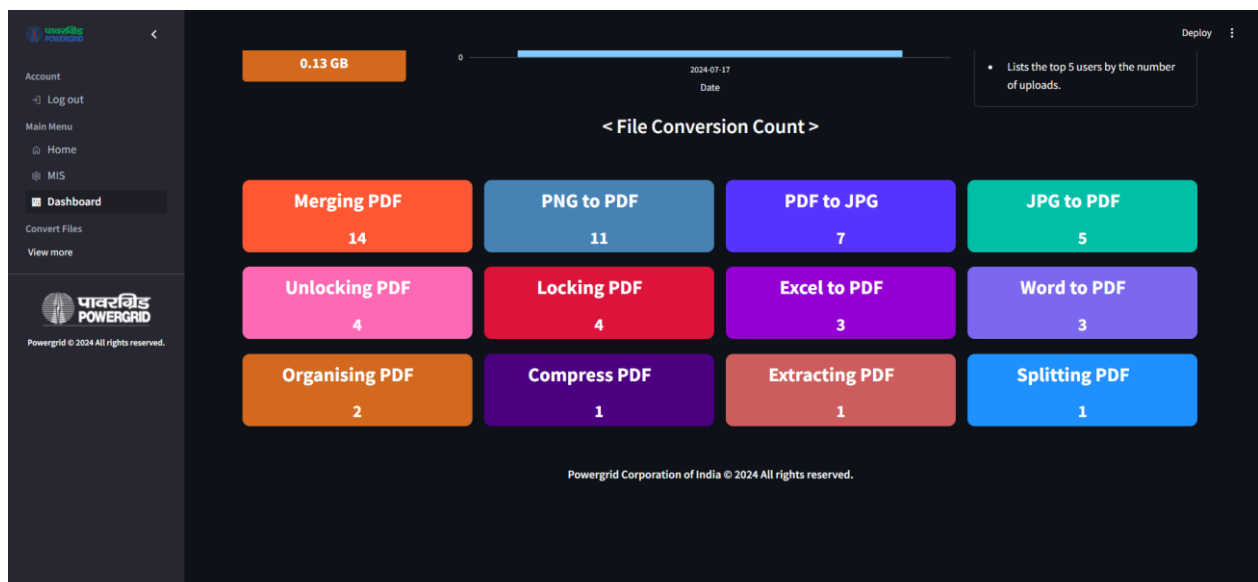
FileId	Name	Description	Status	ContentType	Size	UpdatedBy	Updi
0	1 testisi.pdf	PDF to JPG	1	.pdf	34,976	Ravi	2024
1	2 testisi.pdf	PDF to JPG	1	.pdf	34,976	Ravi	2024
2	3 Cyber Security Syllabus.jpg	JPG to PDF	1	.jpg	180,315	Ravi	2024
3	4 Cyber Security Syllabus.jpg	JPG to PDF	1	.jpg	180,315	Ravi	2024
4	5 AI Project Report - Converzo.docx	Word to PDF	1	.docx	45,314	Ravi	2024
5	6 RaviSharma'sResume.pdf	Compress PDF	1	.pdf	260,718	Ravi	2024

Dashboard Page

1. The dashboard provides a centralized hub for users to track and manage files effectively. It offers insights into usage statistics.



2. Users may scroll down the dashboard page to find information about the File conversion count.

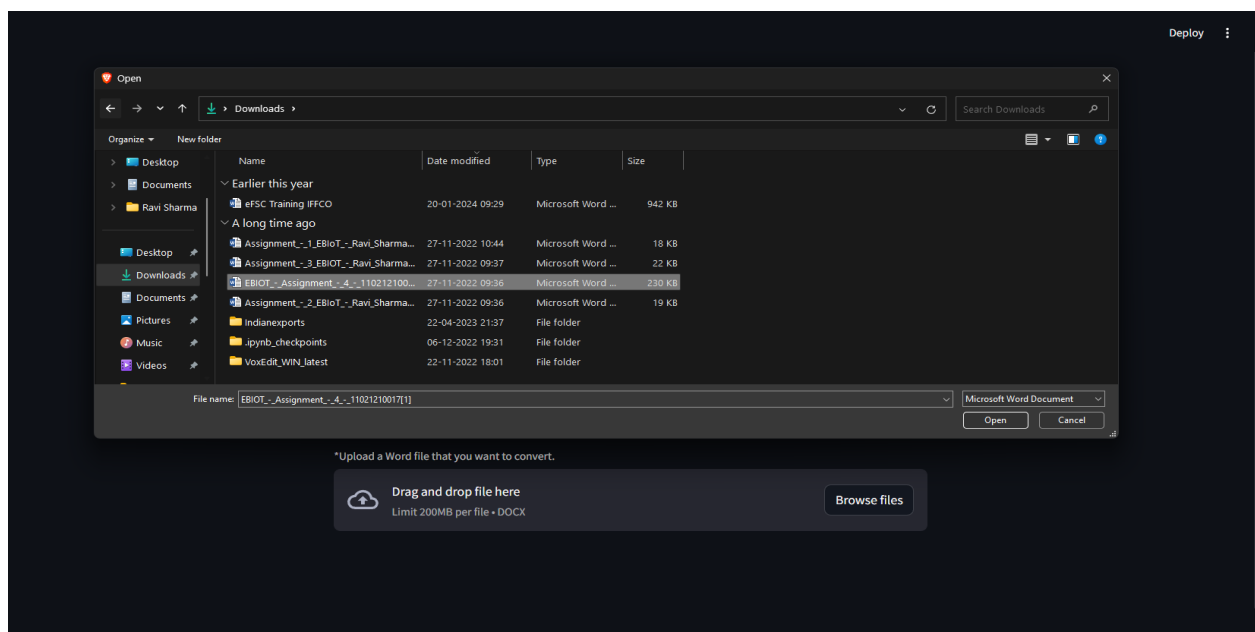


Word to PDF

1. On navigating through the side bar to the “Word to PDF” Page users will see this interface.



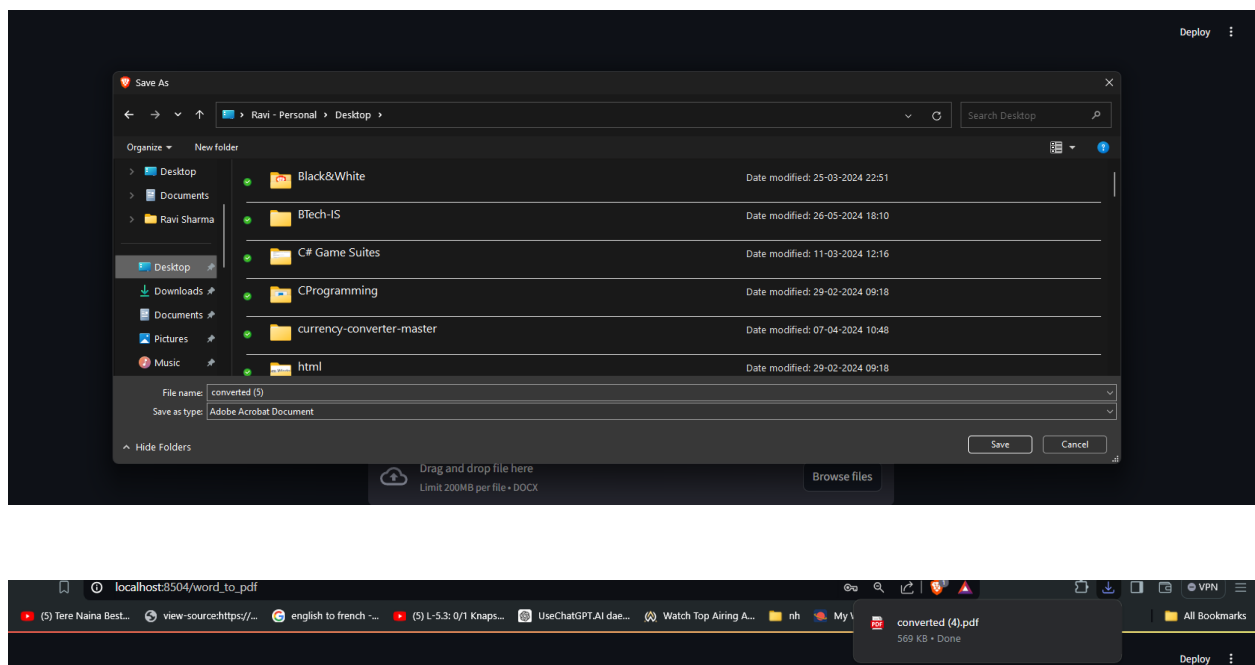
2. On clicking “Browse Files” users are prompted to select a docx file from their system that they want to convert to pdf.



3. On selecting the file from their system the file if valid is uploaded successfully and the process of conversion starts.



4. On clicking the “Download PDF” Button users can download the pdf.

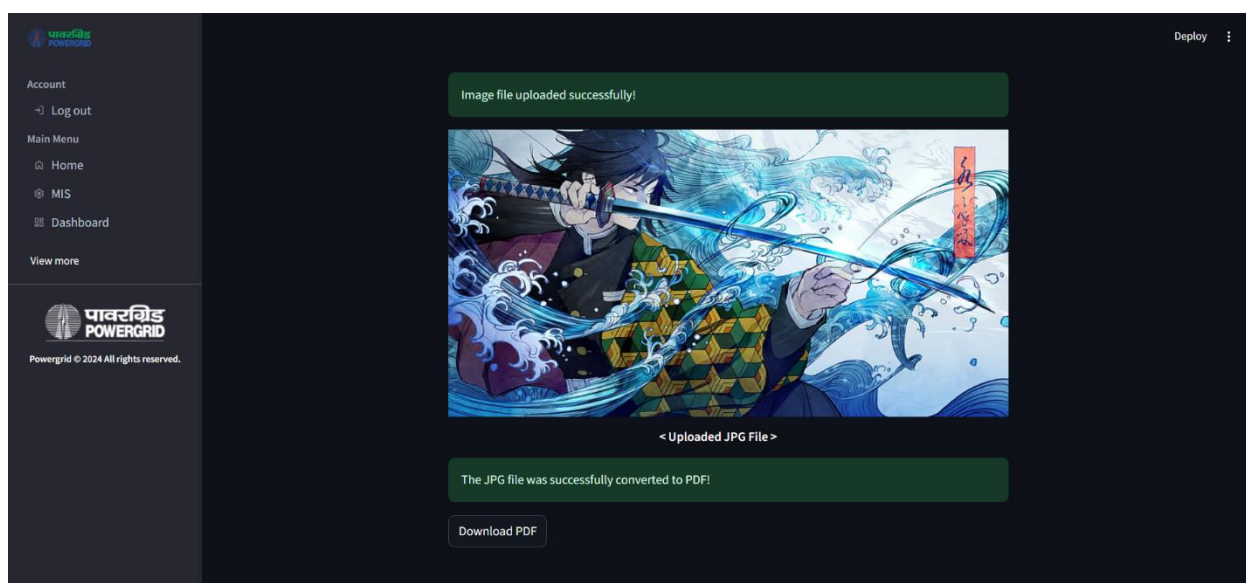


JPG to PDF

1. On navigating through the side bar to the “JPG To PDF” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select a jpg file from their system that they want to convert to pdf and download them.

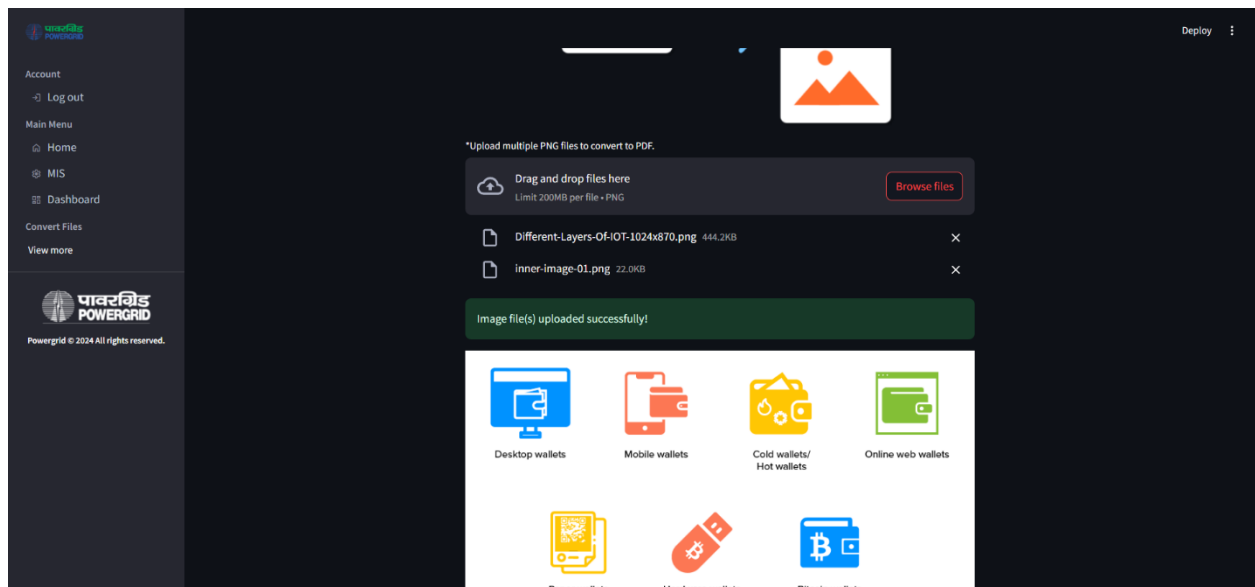


PNG to PDF

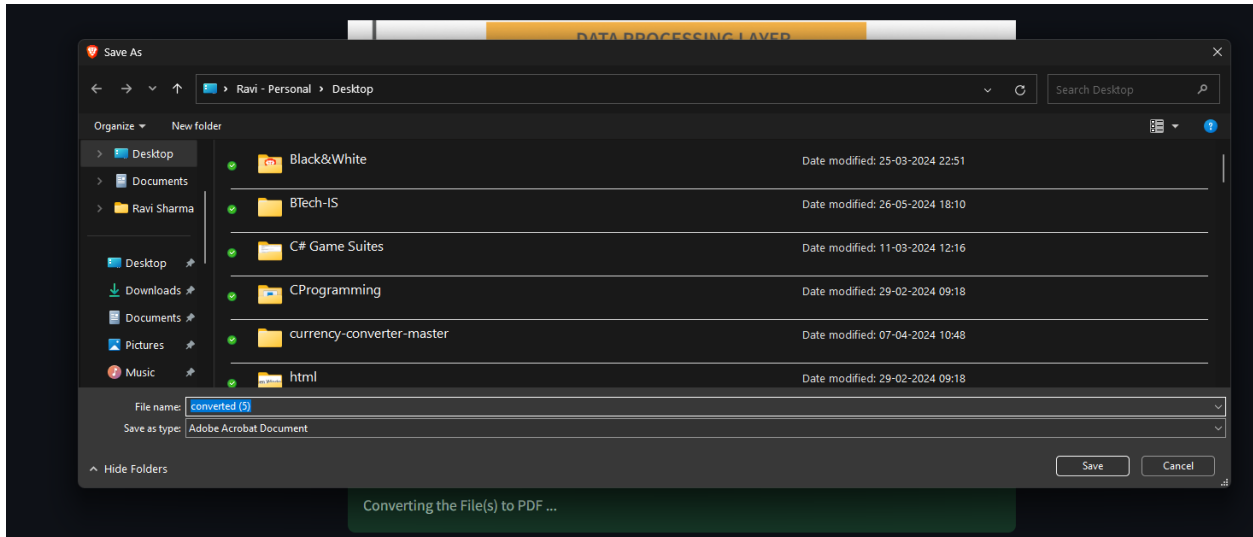
1. On navigating through the side bar to the “PNG To PDF” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select png file(s) from their system that they want to convert to pdf.

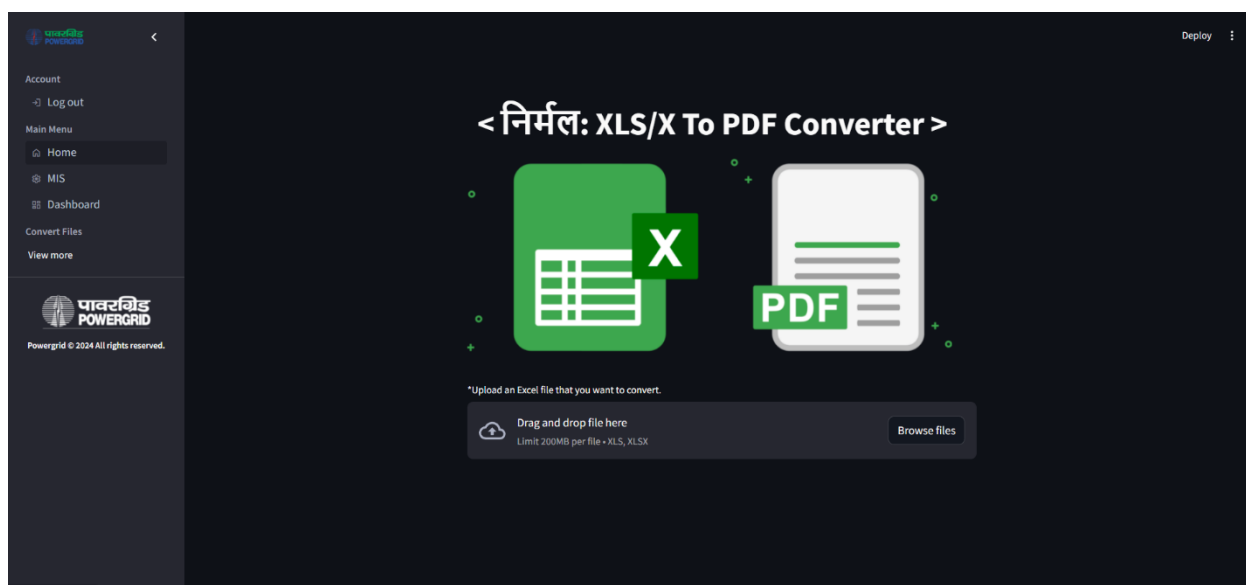


3. Users can download the converted PDF file by clicking on the “Download PDF” button.

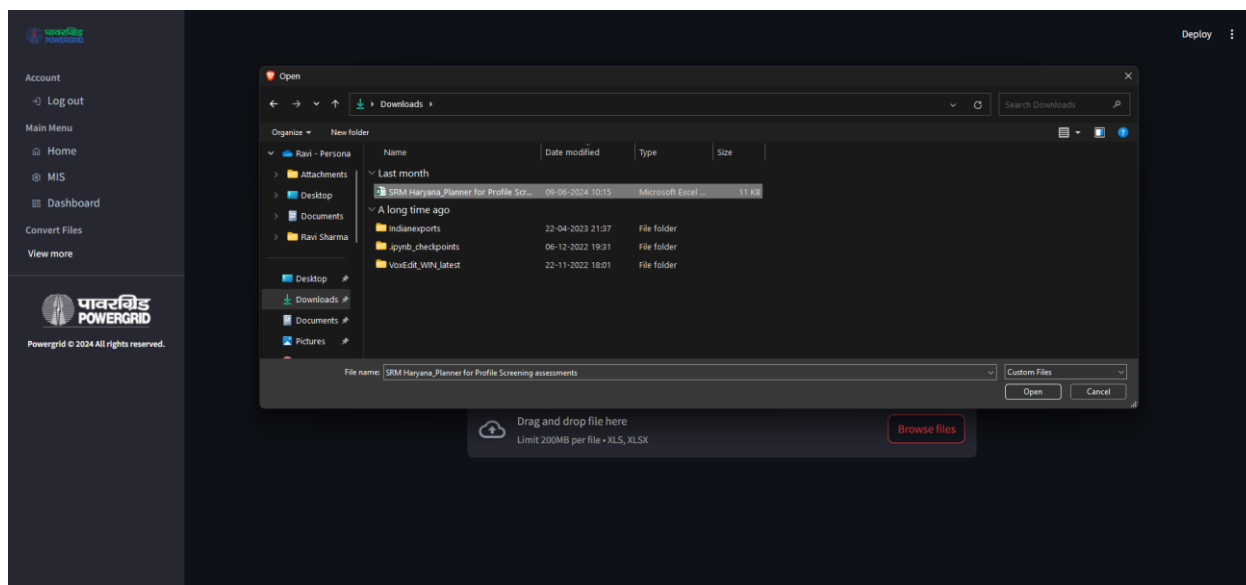


Excel to PDF

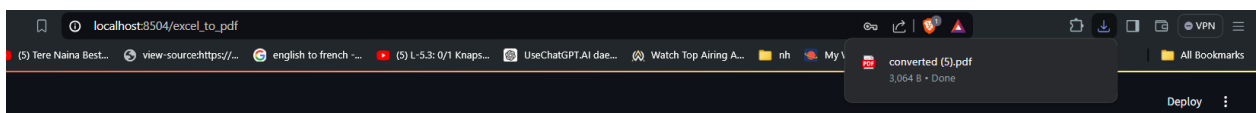
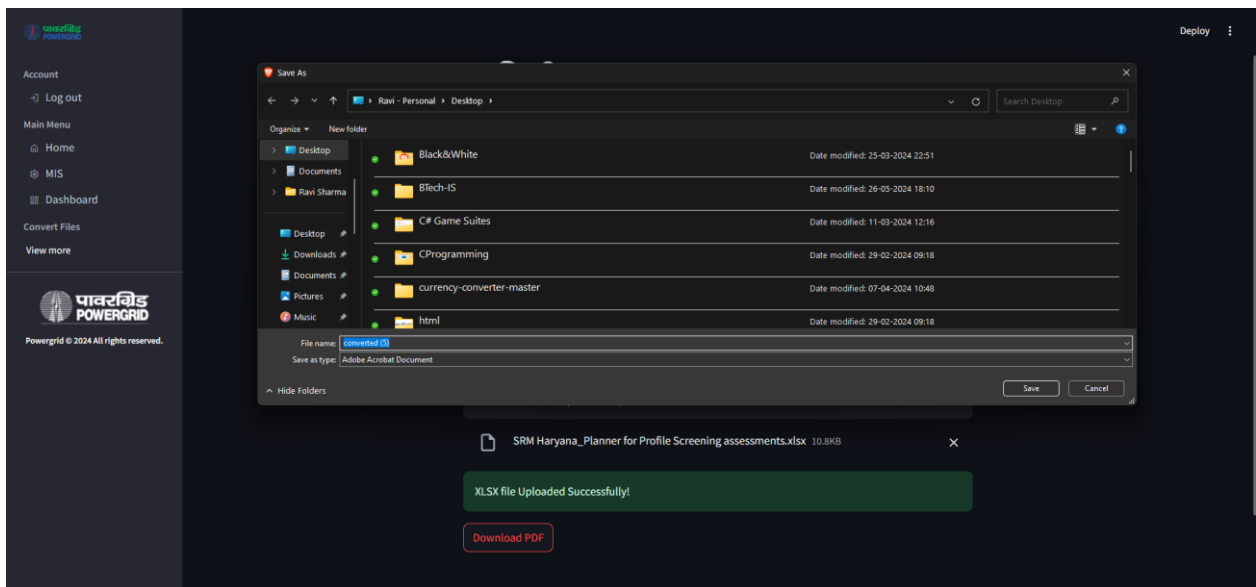
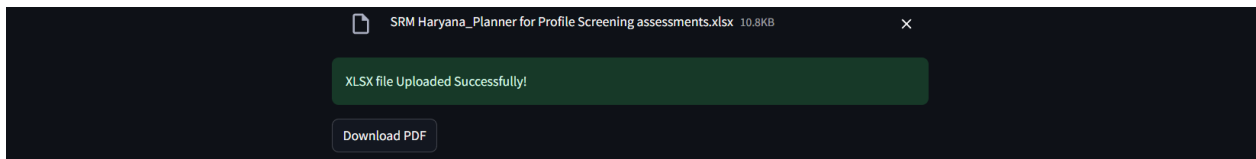
1. On navigating through the side bar to the “Excel To PDF” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select xls/xlsx file from their system that they want to convert to pdf.

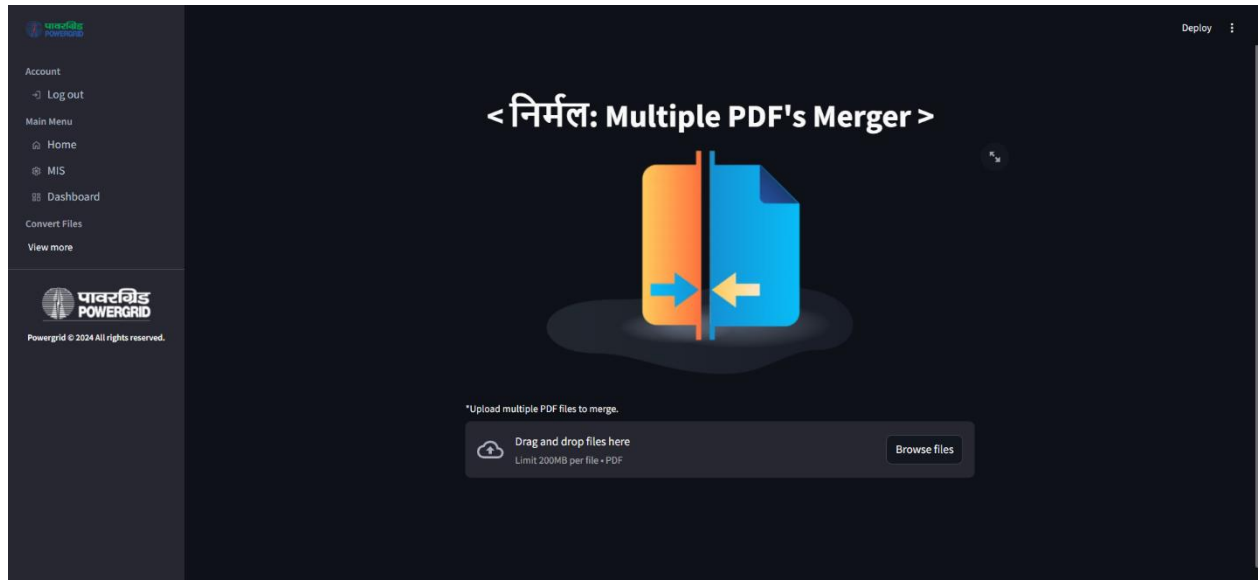


3. Users can download the converted PDF file by clicking on the “Download PDF” button.

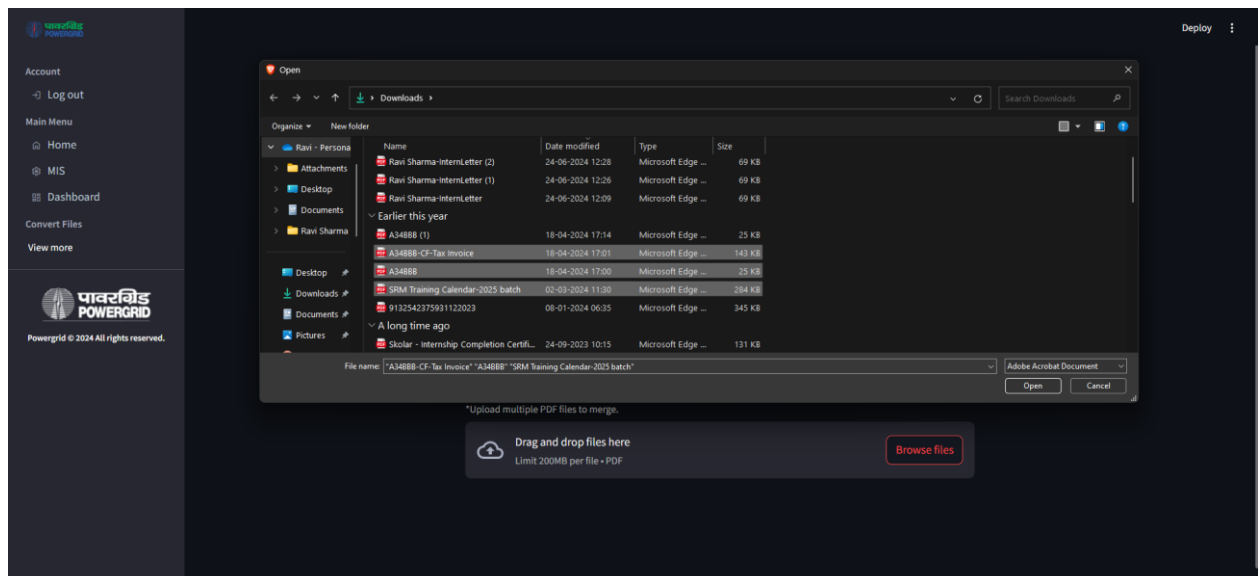


Merge PDF's

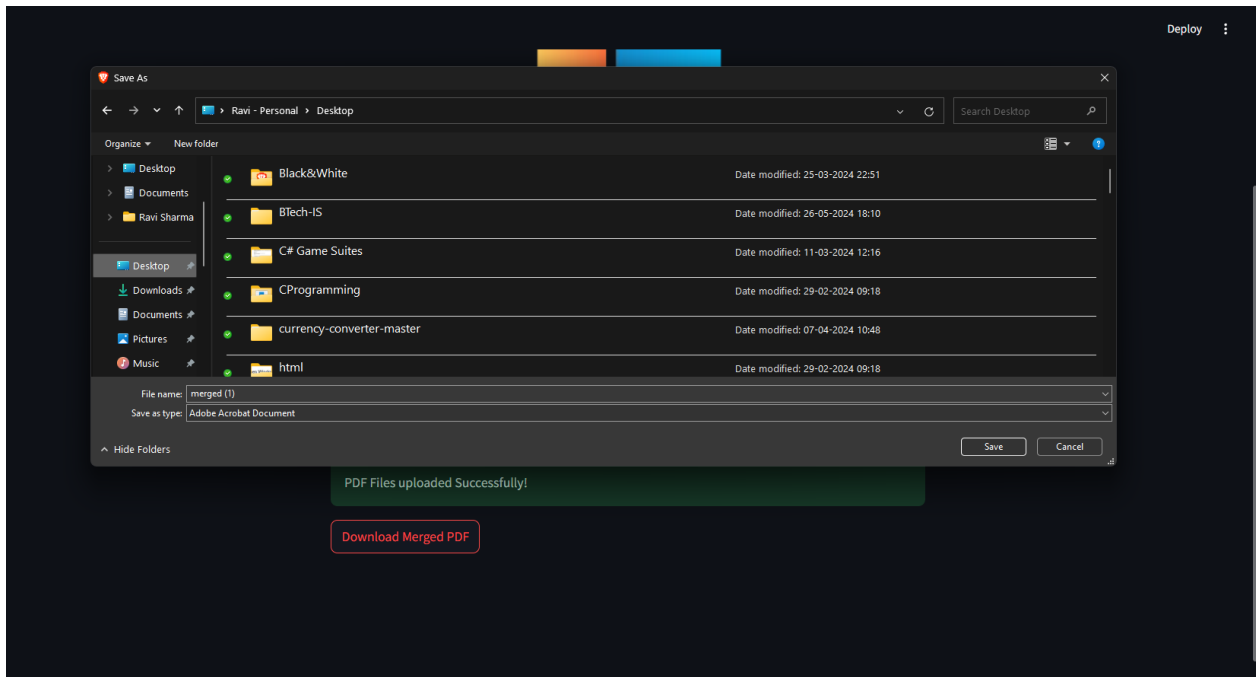
1. On navigating through the side bar to the “Merge PDF's” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select multiple PDF files from their system that they want to merge.



3. Users can download the converted PDF file by clicking on the “Download Merged PDF” button.

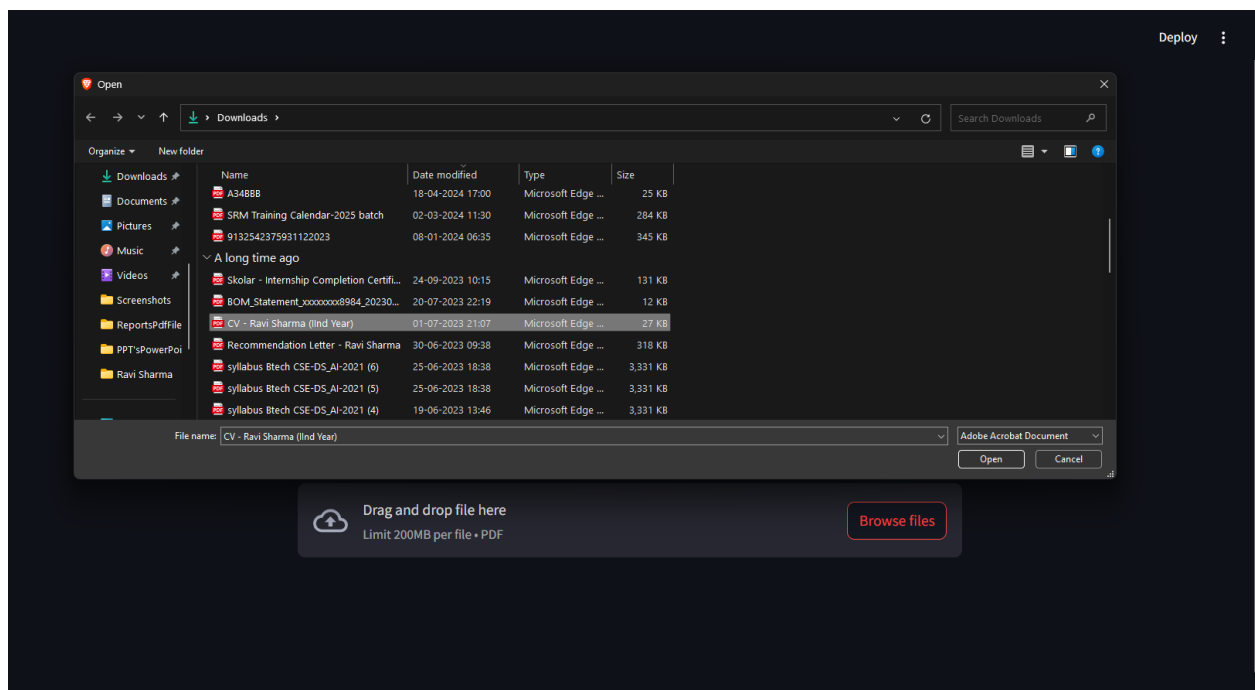


Protect PDF

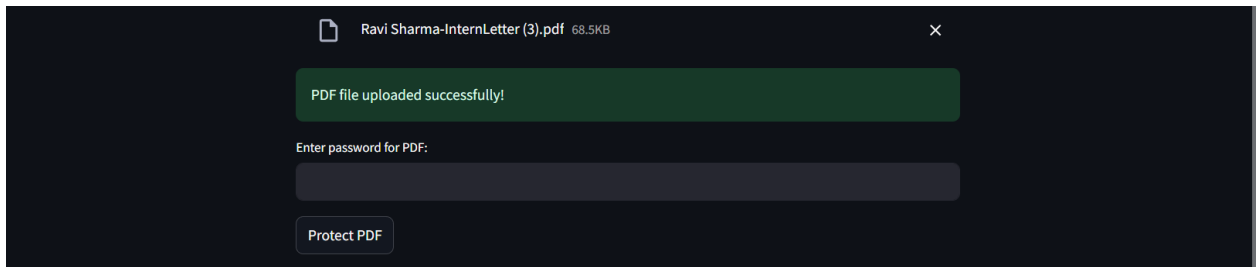
1. On navigating through the side bar to the “Protect PDF” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select a PDF from their system that they want to protect.



3. After successfully uploading the files users are asked to enter a new password with which they want to protect the PDF.



4. After successfully entering the password and on clicking “Protect PDF” button users can download the protected PDF.

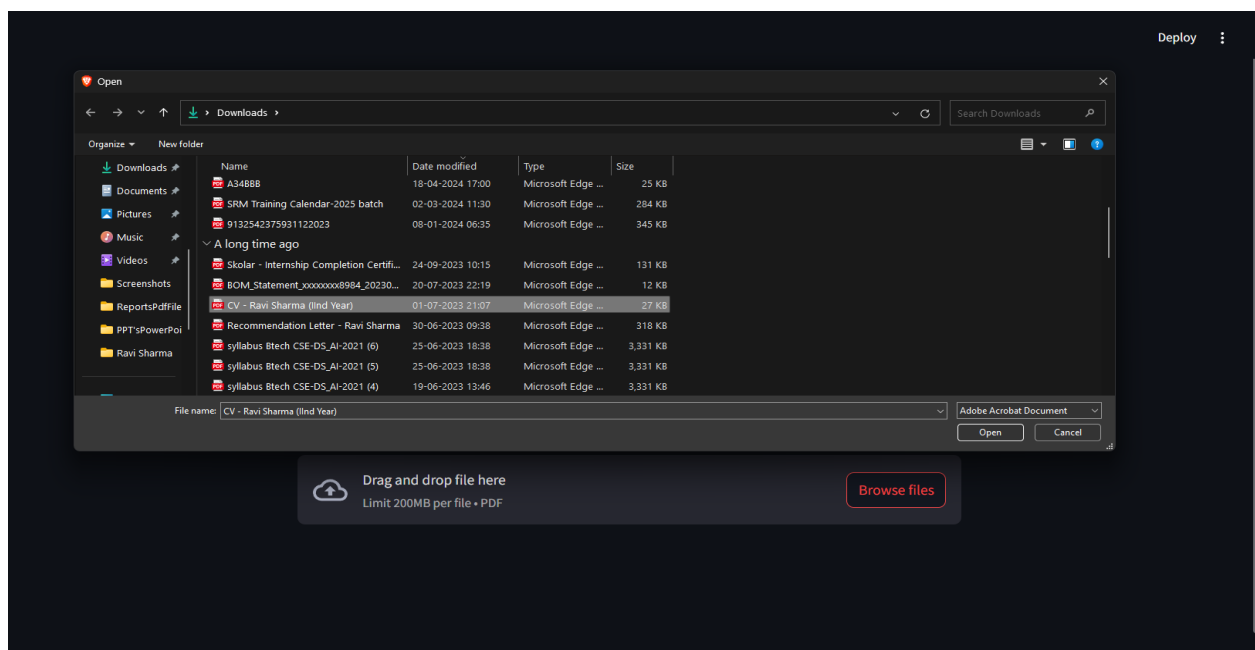


Unlock PDF

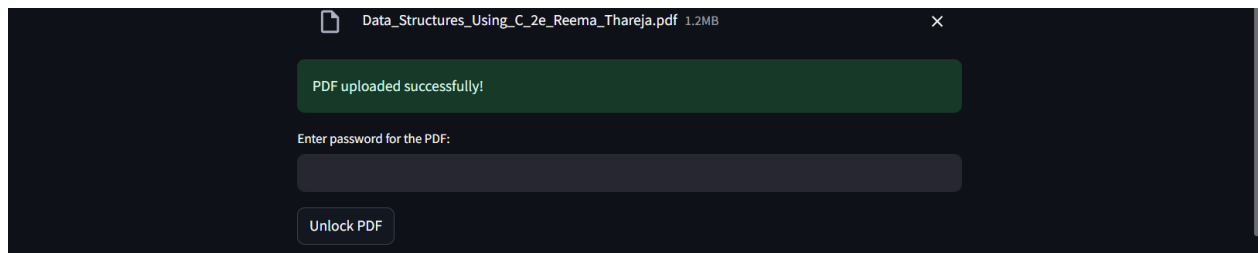
1. On navigating through the side bar to the “Unlock PDF” Page users will see this interface.



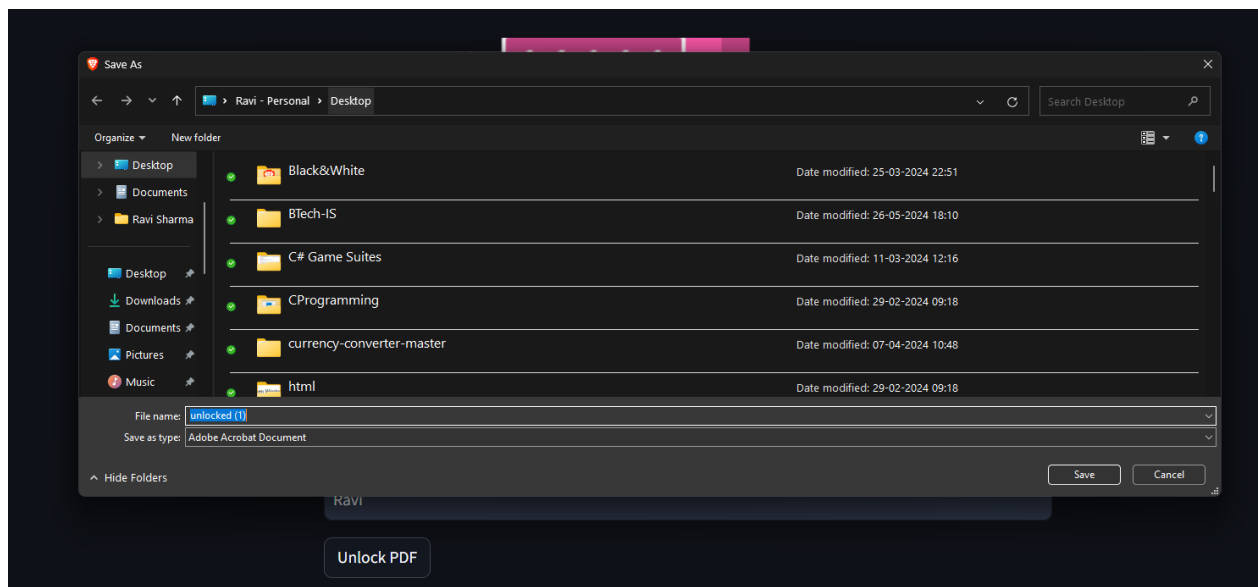
2. On clicking “Browse Files” users are prompted to select a PDF from their system that they want to unlock.



3. After successfully uploading the files users are asked to enter the password of the PDF to unlock it.

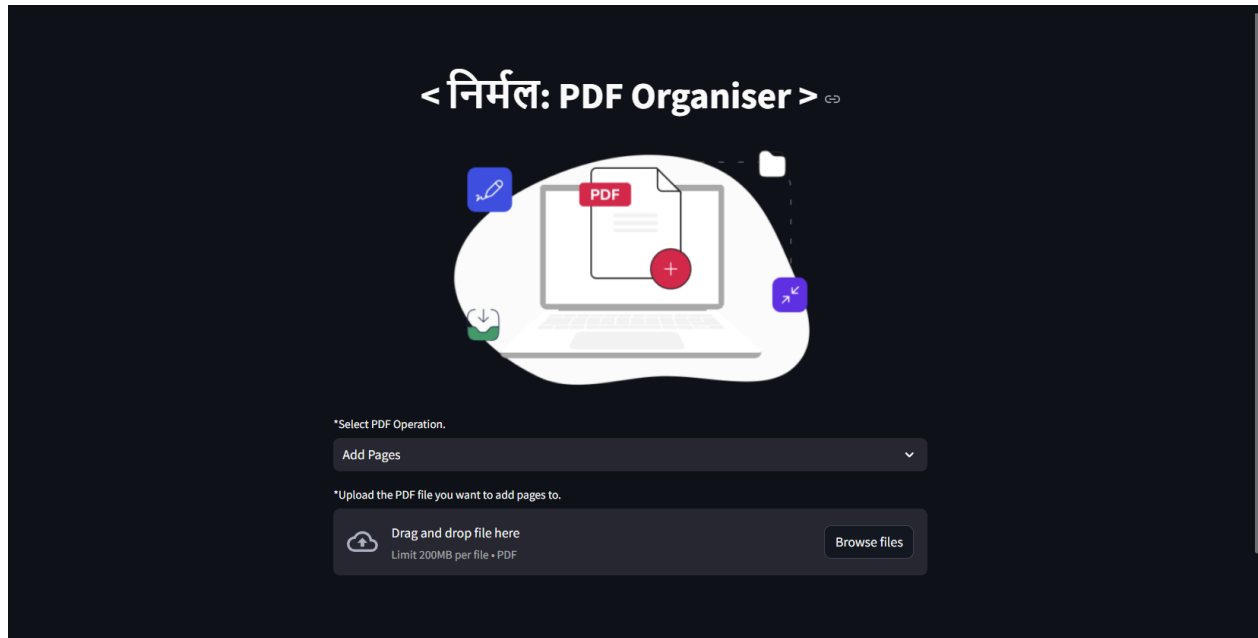


4. After successfully entering the password and on clicking “Unlock PDF” button users can download the Unlocked PDF.

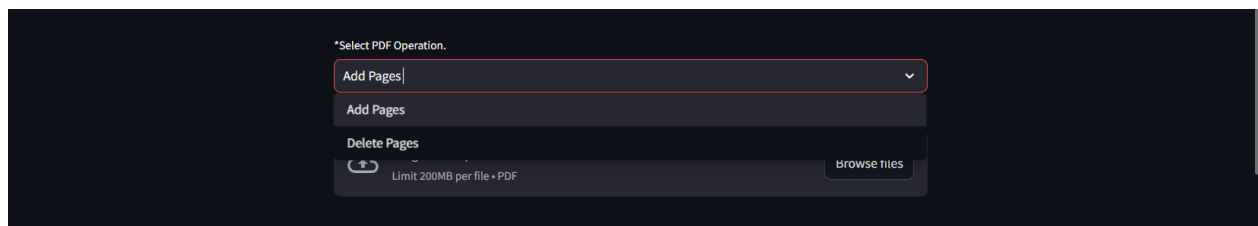


Organsie PDF

1. On navigating through the side bar to the “Organise PDF” Page users will see this interface.



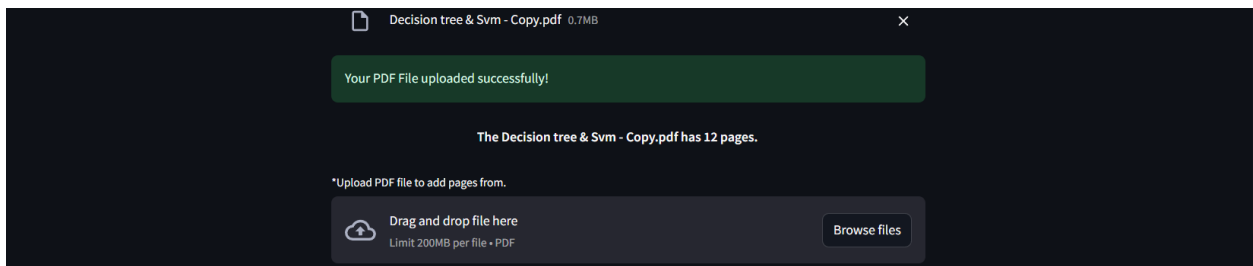
2. Users can choose between two operations that they perform to the PDF that are Add pages to the PDF and Delete pages from the PDF.



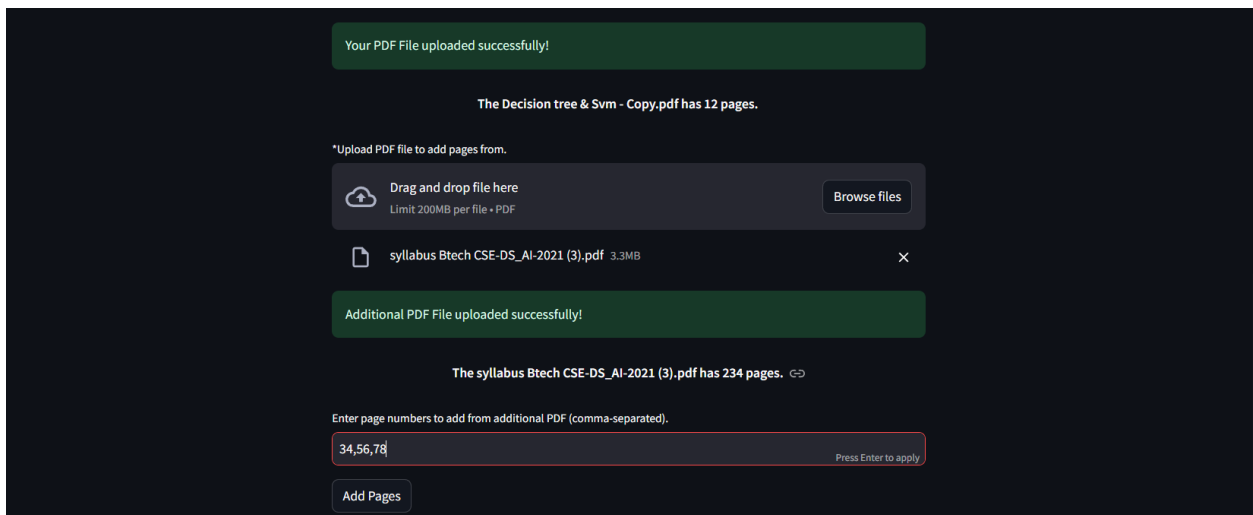
3. After choosing the operation that they want to perform they can browse the PDF from their device by clicking on “Browse files”.



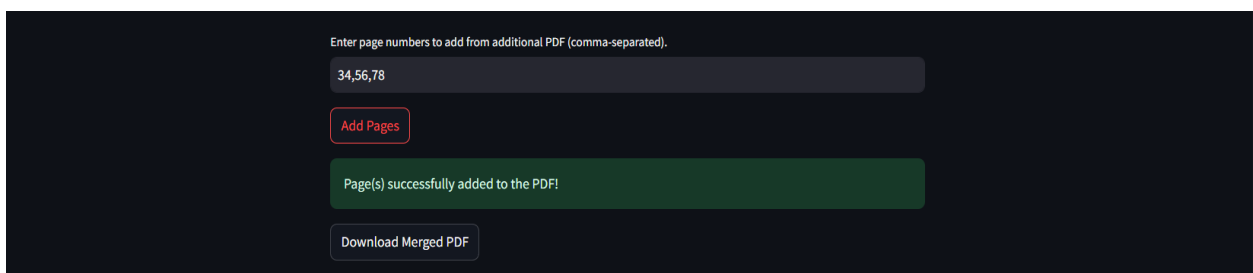
4. After successfully uploading the file the user is asked to select the file that they want to add pages from (if the user selected “Add Pages” as the operation).



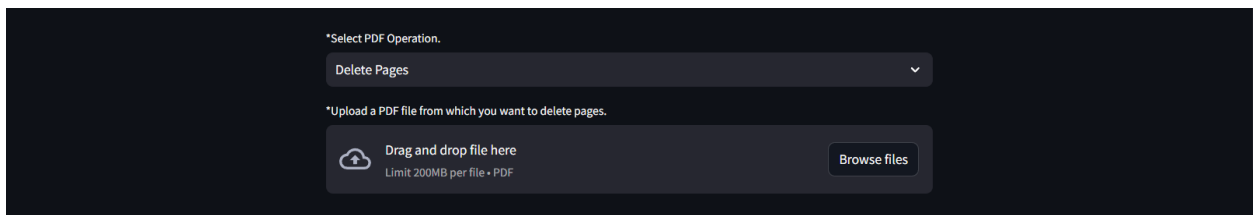
5. Users may now browse to select the file from which they want to add pages from and then type the pages they want to add (comma separated).



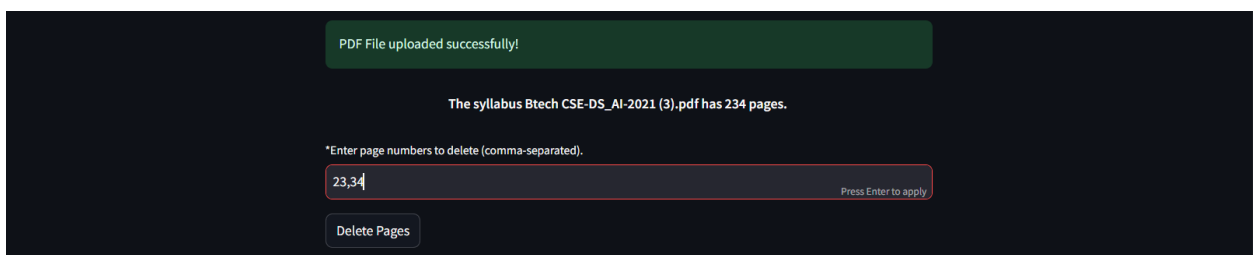
6. After Adding the pages by clicking on “Add Pages” users can now download the new PDF File with added pages.



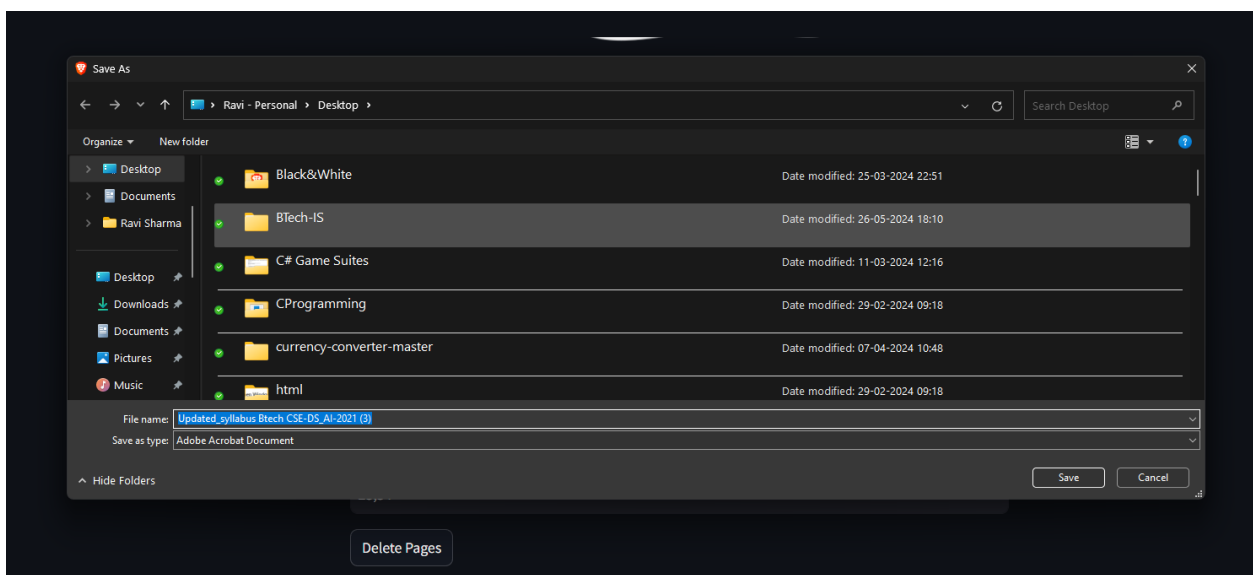
7. Now if the user wants to delete pages from the PDF, The user shall select the “Delete Pages” option from the expanded bar.



8. After browsing the file from which you want to delete the pages from the users is prompted to enter the page number that they want to delete.

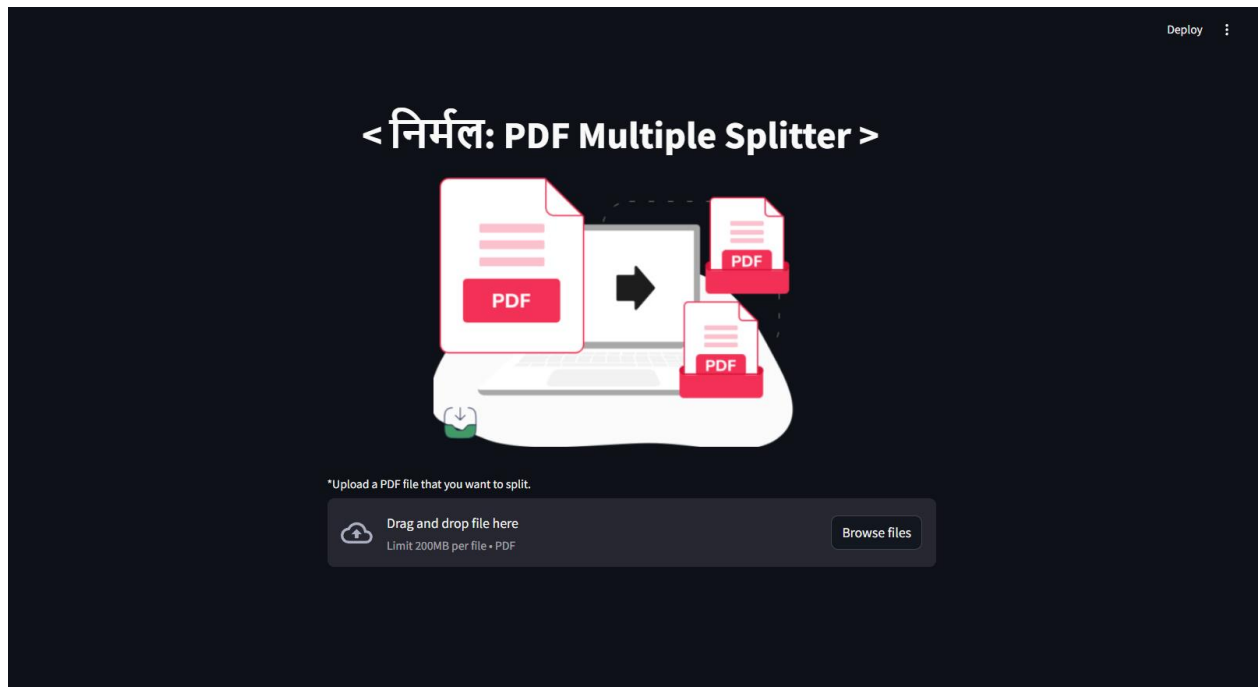


9. The user can now click on “Delete Pages” button to delete the respected pages from the PDF and download the new PDF.

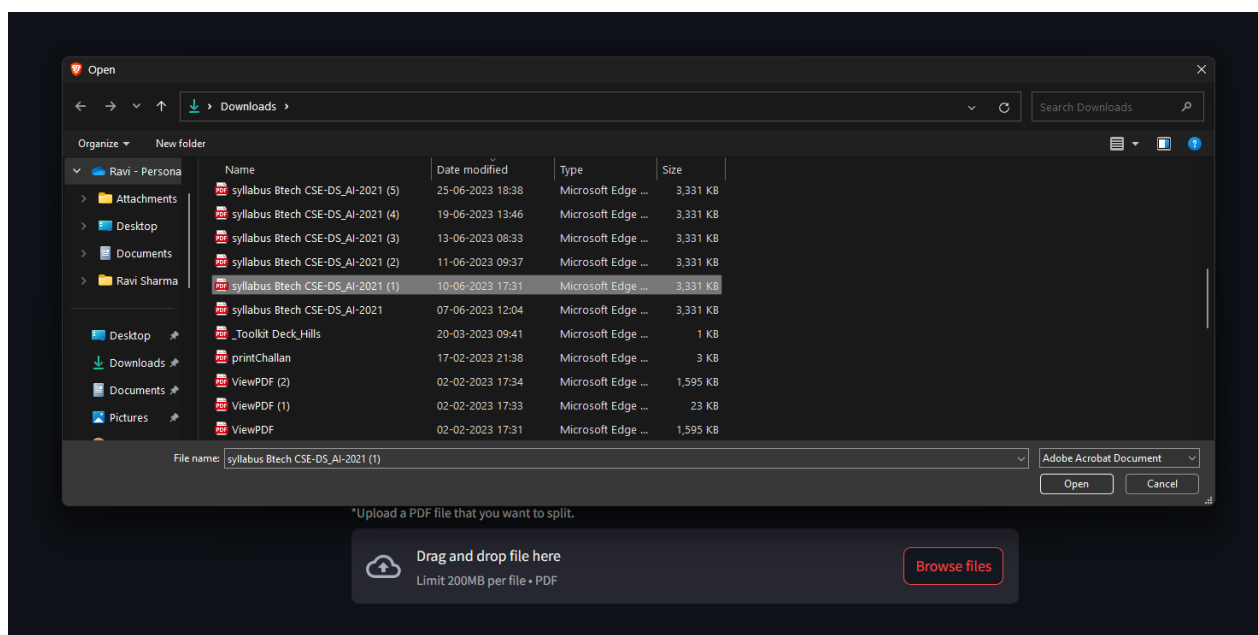


Split PDF

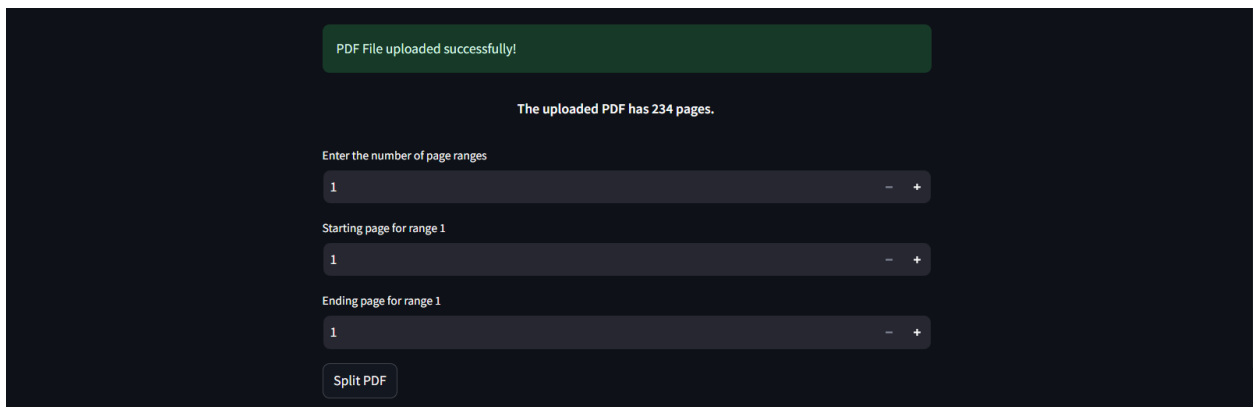
1. On navigating through the side bar to the “Split PDF” Page users will see this interface.



2. On clicking “Browse Files” users are prompted to select a PDF from their system that they want to split.



3. After successfully uploading the files users are asked to enter the number of page ranges as well “starting page for split 1” and “Ending page for split 1” and so on.



PDF File uploaded successfully!

The uploaded PDF has 234 pages.

Enter the number of page ranges

1

Starting page for range 1

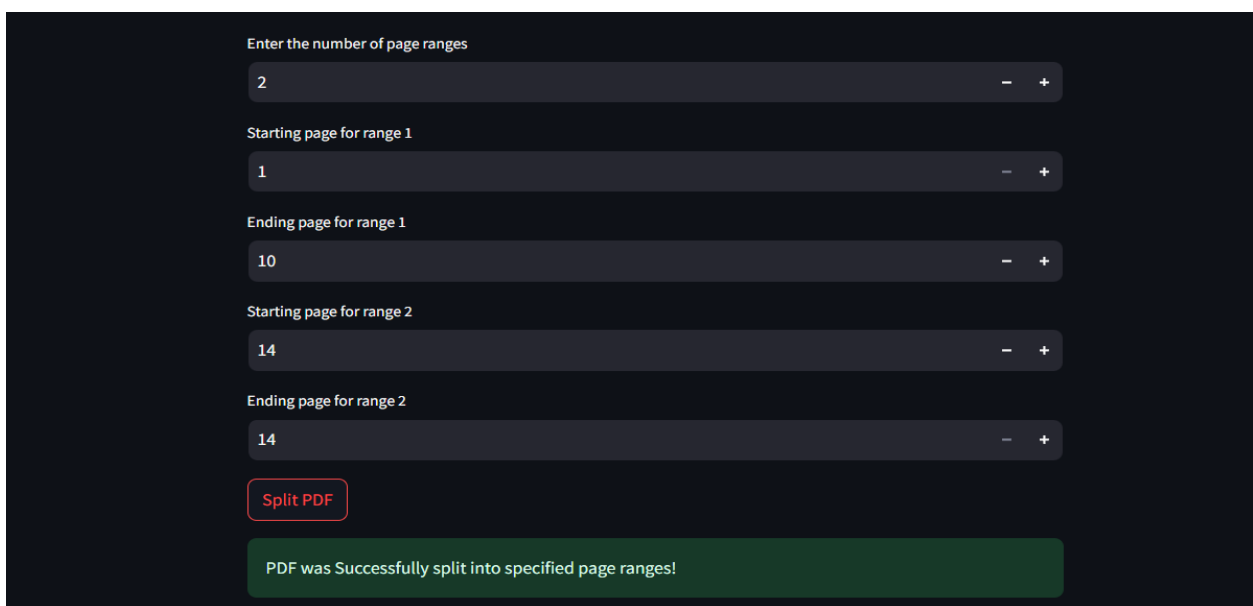
1

Ending page for range 1

1

Split PDF

3. After giving proper input ranges user may now select the “Split PDF” button to split the PDF into new PDF’s



Enter the number of page ranges

2

Starting page for range 1

1

Ending page for range 1

10

Starting page for range 2

14

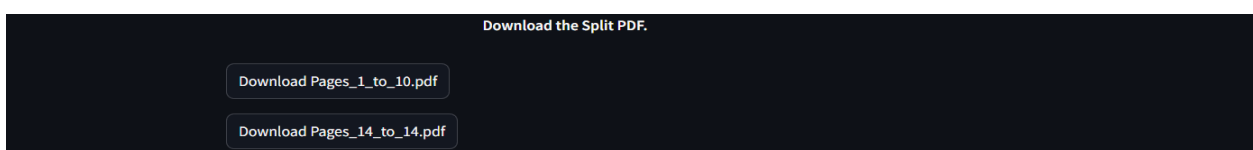
Ending page for range 2

14

Split PDF

PDF was Successfully split into specified page ranges!

4. The user may download the individual splits of the PDF by using “Download Pages” button.



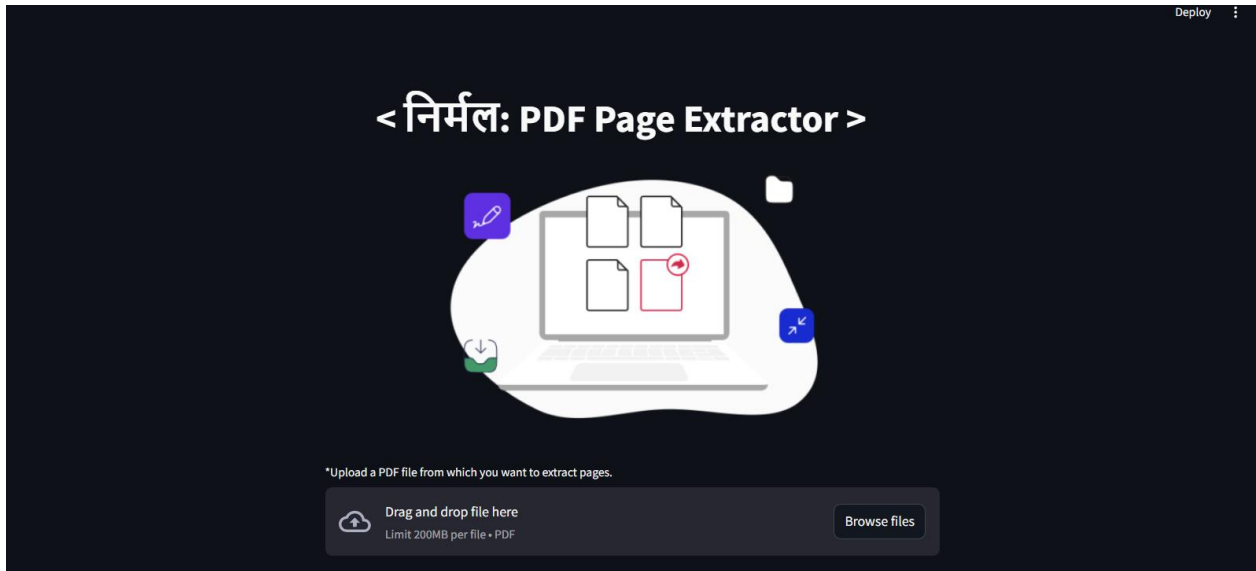
Download the Split PDF.

Download Pages_1_to_10.pdf

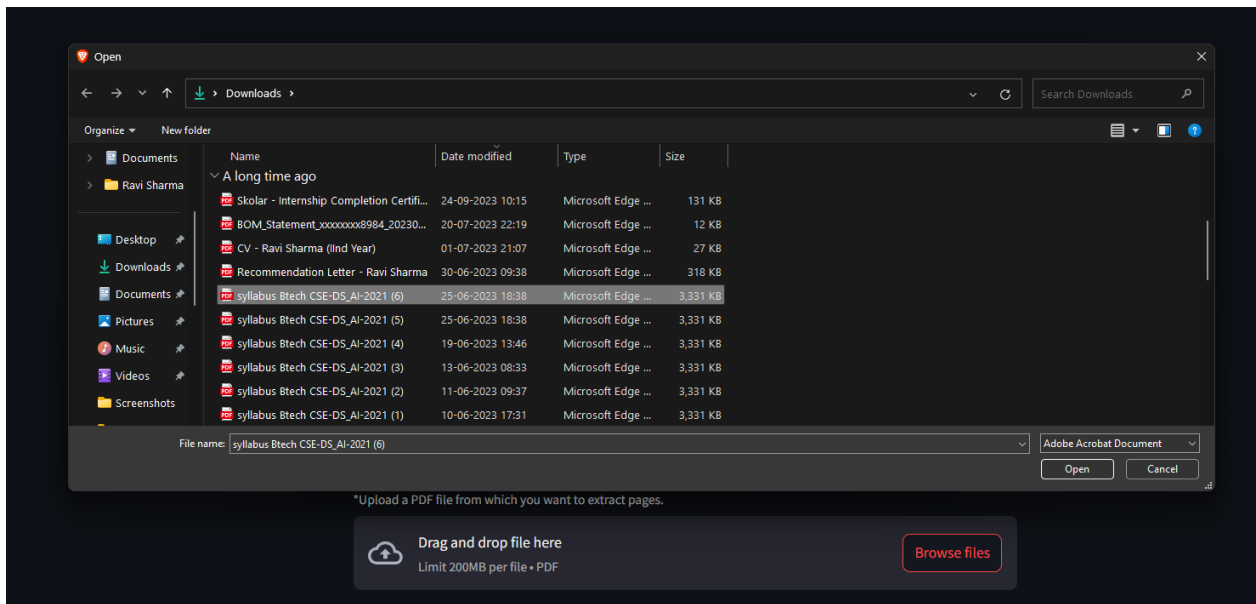
Download Pages_14_to_14.pdf

Extract PDF

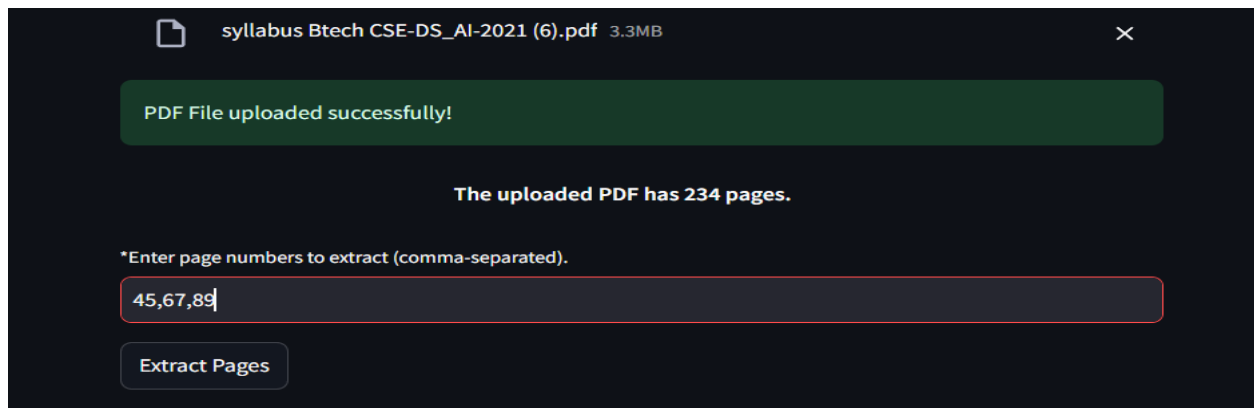
1. On navigating through the side bar to the “Extract PDF” Page users will see this interface.



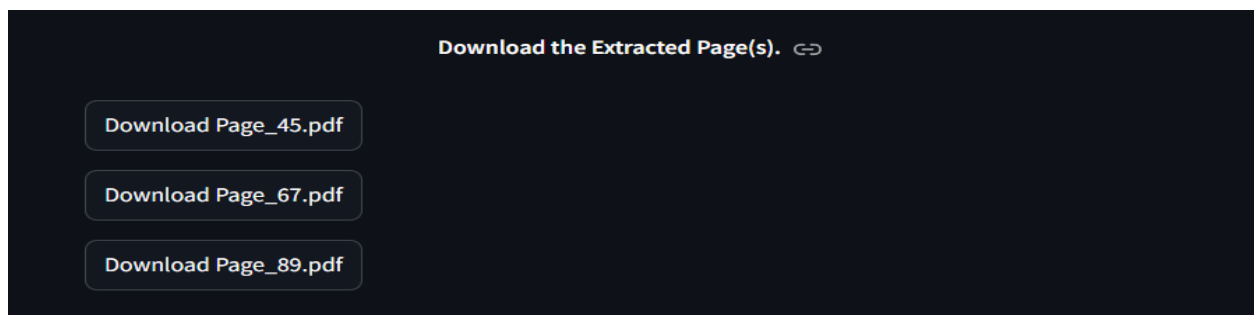
2. On clicking “Browse Files” users are prompted to select a PDF from their system from which they want to extract the pages from.



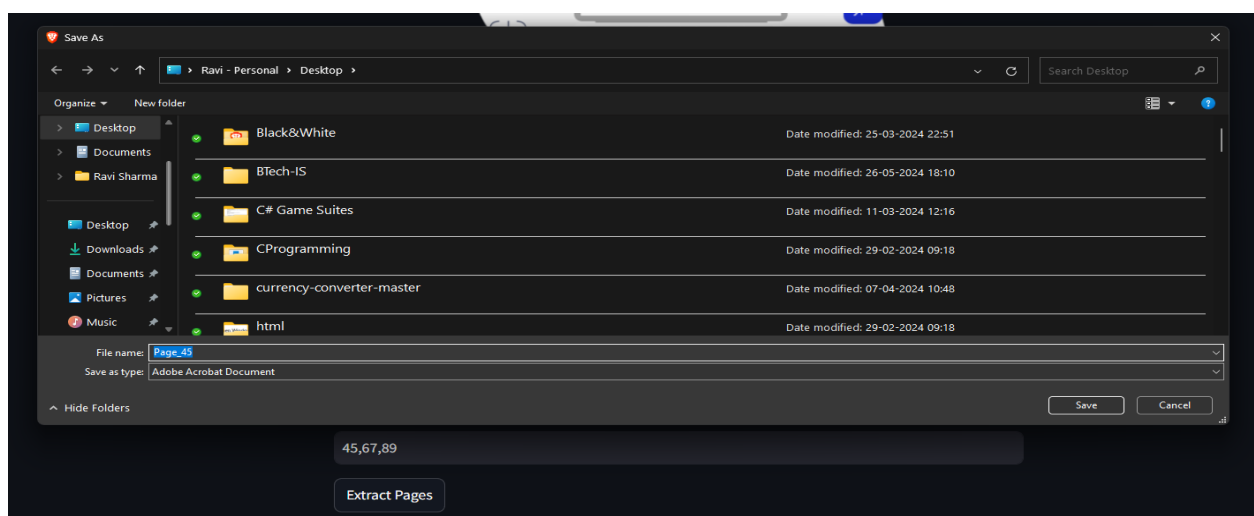
3. Now the users should enter the pages (comma separated) that they want to extract from the PDF.



4. Users should not click on the “Extract Pages” button to extract the respective pages from the PDF.



5. Users can download them using the “Download Pages” button

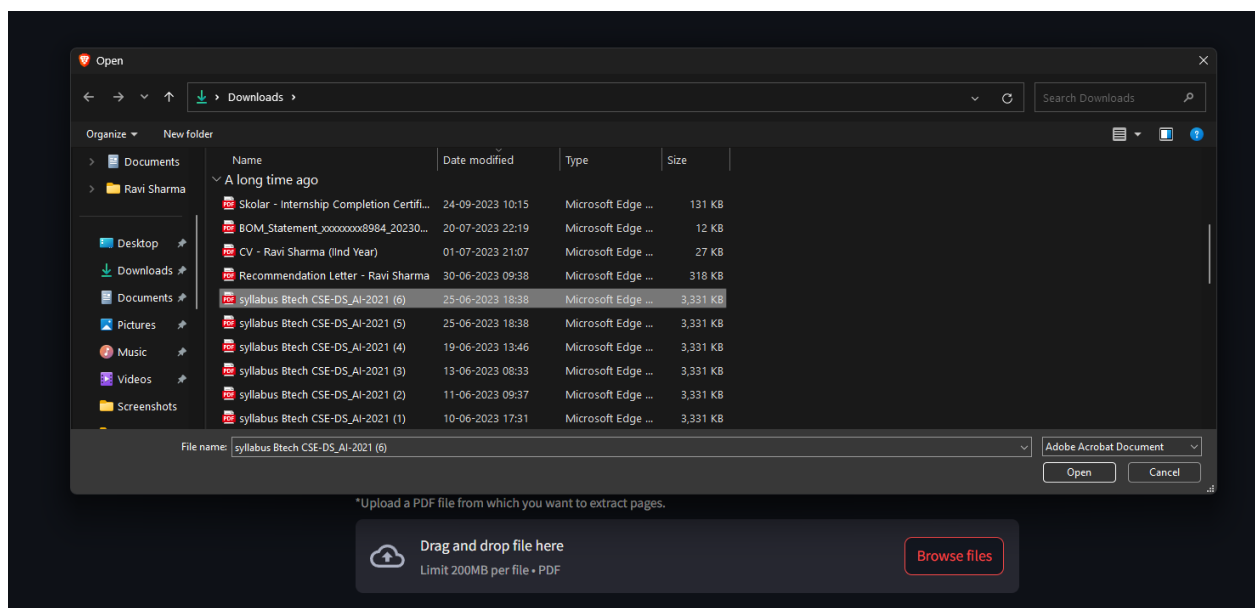


Compress PDF

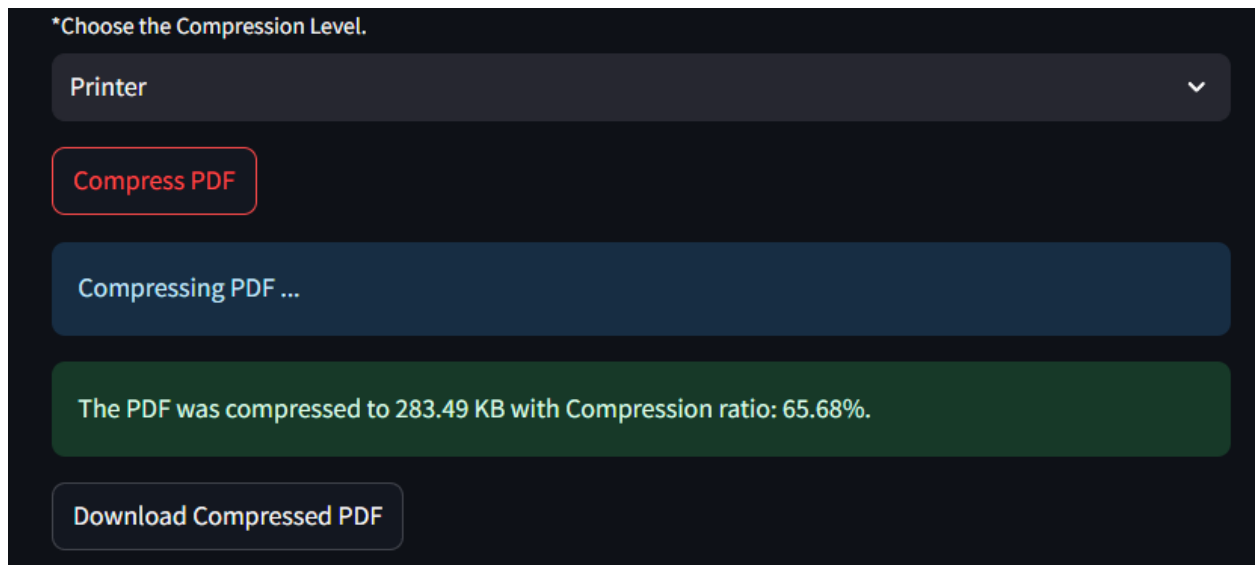
1. On navigating through the side bar to the “Compress PDF” Page users will see this interface.



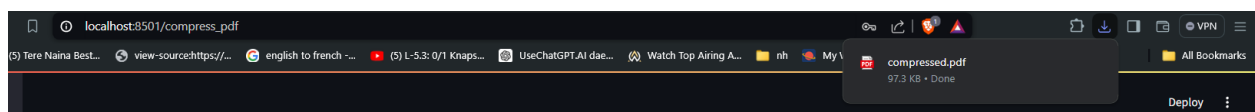
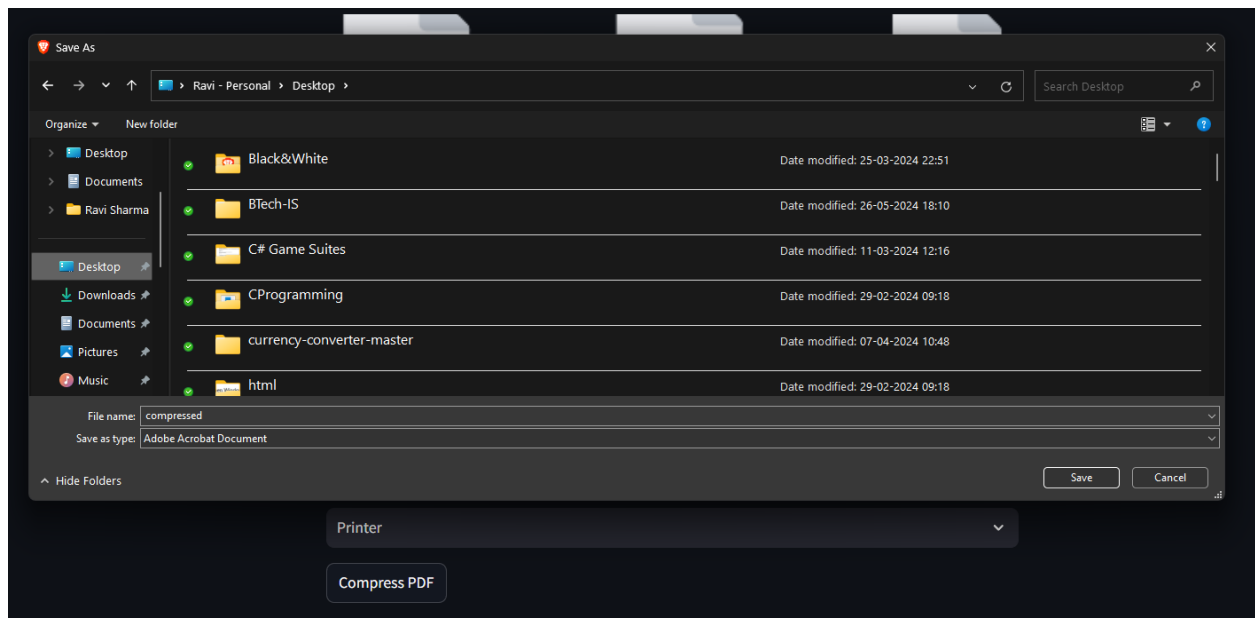
2. On clicking “Browse Files” users are prompted to select a PDF from their system from which they want to compress.



3. After selecting the compression level from the expandable bar the user now has to click the “compress PDF” button.



4. The PDF has been successfully compressed and the user can download the compressed PDF by clicking on the “Download Compressed PDF” button.



Exiting the Application

1. To exit the application the user has to navigate to “Logout” page under the Account section of the application.



2. Users are shown the message that “They are logged in” and are provided “Logout” button which allows the user to exit the application.



3. Clicking on the “Logout” button takes user to the “Login” page of the application.



Conclusion

This project aimed to develop a comprehensive and user-friendly application for managing various file conversion and PDF operations. The primary objective was to facilitate seamless conversion between different file formats, ensure document security, and provide efficient management tools for handling PDF documents. Throughout this project, several libraries and technologies were utilized to achieve these goals.

Key Features of the application that were successfully implemented are –

1. Excel to PDF
2. Word to PDF
3. PNG to PDF
4. JPG to PDF
5. PDF to JPG

PDF Operations:

1. Merging PDFs
2. Splitting PDFs by page range
3. Extracting specific pages from PDFs
4. Protecting PDFs with passwords
5. Unlocking encrypted PDFs
6. Compressing PDFs
7. Adding watermarks to PDFs
8. Digitally signing PDFs

Other Features:

1. Login And Authentication
2. Home Page
3. Management Information System
4. Dashboard

The project successfully implemented a wide range of functionalities that cater to the needs of users dealing with various document formats. The use of multiple libraries and tools ensured that the application is robust and capable of handling complex operations efficiently. Additionally, security measures such as password protection and digital signatures were integrated to ensure document integrity and confidentiality.

References

1. **Li, J., Wang, P., He, Y., Sun, L., Jia, L., Li, Q. & Sun, Y. (2023, December).** Design and implementation of a drawing encryption tool based on image processing and PDF manipulation. In Fourth International Conference on Signal Processing and Computer Science (SPCS 2023) (Vol. 12970, pp. 547-551). SPIE.
2. **Khorasani, M., Abdou, M., & Hernández Fernández, J. (2022).** Web Application Development with Streamlit. *Software Development*, 498-507.
3. **Siahaan, V., & Sianipar, R. H. (2019).** Python GUI with SQL Server for Absolute Beginners: Building Responsive, Powerful Cross-platform, and Database-Driven Applications with PyQt. SPARTA PUBLISHING.
4. **Umesh, P. (2012).** Image processing in python. *CSI Communications*, 23(2), 23-24.
5. **Dewson, R. (2008).** Beginning SQL Server 2008 Express for Developers: From Novice to Professional. Apress.
6. **Koning, B. (2022).** Extracting Sections From PDF-Formatted CTI Reports (Bachelor's thesis, University of Twente).
7. **Hunt, J., & Hunt, J. (2019).** Working with excel files. *Advanced Guide to Python 3 Programming*, 249-255.