

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgavi-590018, Karnataka, India



## MINI PROJECT REPORT-21ISMP67

On

## “WEB BASED CROSS PLATFORM FILE SYSTEM RECOVERY TOOL”

Submitted in Partial Fulfillment of the requirements of 6<sup>th</sup> Semester

### BACHELOR OF ENGINEERING IN INFORMATION SCIENCE AND ENGINEERING

Submitted By

BABUREDDY B N	1SJ21IS009
IRFAN KHAN I	1SJ21IS034
MOHAMMED MAZZ	1SJ21IS059
SRIDHARA M	1SJ22IS401

Under the guidance of

**Prof. Aravinda Thejas Chandra**

**Associate Professor**

**Dept. of ISE**



DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING  
SJC INSTITUTE OF TECHNOLOGY

CHIKKABALLAPUR-562101

2024

||Jai Sri Gurudev||

**Sri Adichunchanagiri Shikshana Trust®**

**SJC INSTITUTE OF TECHNOLOGY, Chickballapur-562101**

**Department of Information Science and Engineering**



### **CERTIFICATE**

This is to certify that the mini project work entitled **“WEB BASED CROSS PLATFORM FILE SYSTEM RECOVERY TOOL”** carried out by **BABUREDDY B N (1SJ21IS009), IRFAN KHAN I (1SJ21IS034), MOHAMMED MAZZ (1SJ21IS059), SRIDHARA M (1SJ22IS401)** and a bonafide students of “6<sup>th</sup>” SEM “ISE” in partial fulfillment for the mini project of 6<sup>th</sup> Semester **Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi** during the year **2023-2024**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements with respect to mini project work prescribed for the Bachelor of Engineering degree.

.....  
**Prof. Aravinda Thejas Chandra**

Associate Professor,  
Dept. of ISE,SJCIT

.....  
**Prof. Sathesh Chandra Reddy S**

Prof. & Head ISE  
Dept. of ISE,SJCIT



## **DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

### **DEPARTMENT VISION**

Educating Students to Engineer Information Science and Technology for Advancing the Knowledge as to best serve the Real world

### **DEPARTMENT MISSION**

1. Focusing on Fundamentals and Applied aspects in both Information Science Theory and Programming practices.
2. Training comprehensively and encouraging R&D culture in trending areas of Information Technology.
3. Collaborating with premier Institutes and Industries to nurture Innovation and learning, in cutting edge Information Technology.
4. Preparing the students who are much Sought-after, Productive and Well- respected for their work culture having Lifelong Learning practice.
5. Promoting ethical and moral values among the students so as to enable them emerge as responsible professionals.

## DECLARATION

We **BABUREDDY B N (1SJ21IS009)**, **IRFAN KHAN I (1SJ21IS034)**, **MOHAMMED MAZZ (1SJ21IS059)**, **SRIDHARA M (1SJ22IS401)** Students of VI semester B.E in **Information Science and Engineering** at **S J C Institute of Technology, Chikballapur**, hereby declare that this work entitled “**WEB BASED CROSS PLATFORM FILE SYSTEM RECOVERY TOOL**” has been carried out for **Mini Project-21ISMP67** in **Dept. of ISE, SJCIT** under the guidance of guide **Prof. Aravinda Thejas chandra**, Associate Professor, **Dept. of ISE, SJC Institute of Technology, Chikballapur** and submitted during the academic year 2023-2024. We further declare that the report had not been submitted to another university for the award of any other degree.

**Place: Chikballapur**

**Date:**

**BABUREDDY B N**

**1SJ21IS009**

**IRFAN KHAN I**

**1SJ21IS034**

**MOHAMMED MAZZ**

**1SJ21IS059**

**SRIDHARA M**

**1SJ22IS401**

# **ABSTRACT**

Data integrity and accessibility are crucial in the digital age, yet data loss remains a significant issue due to accidental deletions, hardware failures, software corruption, and cyber threats. To address this challenge, we propose a web-based cross-platform file system recovery tool designed to provide an efficient, reliable, and user-friendly solution for retrieving lost or corrupted data. This tool will operate seamlessly across multiple operating systems, including Windows and Linux, ensuring broad accessibility and usability. Traditional data recovery methods often require specialized knowledge and are time-consuming and complex, limiting their accessibility to non-technical users. Moreover, existing solutions typically lack cross-platform compatibility, further restricting their utility.

The proposed tool addresses these shortcomings by offering a comprehensive solution that simplifies the data recovery process without sacrificing efficiency or reliability. Advanced recovery algorithms and real-time monitoring features will enable users to manage recovery tasks with ease and confidence, receiving continuous updates throughout the process. Robust security protocols will protect user data during recovery, addressing privacy and security concerns. This tool will be especially beneficial for small businesses, educational institutions, and individual users who may not have access to professional data recovery services.

## ACKNOWLEDGEMENT

With reverential pranam, we express my sincere gratitude and salutations to the feet of his holiness **Paramapoojya Jagadguru Byravaikya Padmabhushana Sri Sri Sri Dr. Balagangadharanatha Maha Swamiji**, his holiness **Paramapoojya Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Maha Swamiji**, and **Sri Sri Mangalnath Swamiji** , Sri Adichunchanagiri Mutt for their unlimited blessings.

First and foremost we wish to express our deep sincere feeling and gratitude to our institution, **Sri Jagadguru Chandrashekaranaatha Swamiji Institute of Technology**, for providing us an opportunity for completing the Mini-Project Work successfully.

We extend deep sense of sincere gratitude to our **Dr. G T Raju, Principal, SJC Institute of Technology, Chickballapur**, for providing an opportunity to complete the Mini-Project Work.

We extend special in-depth, heartfelt, and sincere gratitude to our HOD **Dr. Sathesh Chandra Reddy S, Head of the Department, Information Science and Engineering, SJC Institute of Technology, Chickballapur**, for his constant support and valuable guidance of the Mini-Project Work.

We convey our sincere thanks to our Guide **Prof. Aravinda Thejas Chandra, Associate Professor, Department of Information Science and Engineering, SJC Institute of Technology**, for his constant support, valuable guidance and suggestions of the Mini-Project Work.

Finally, we would like to thank all faculty members of Department of Information Science and Engineering, SJC Institute of Technology, Chickaballapur for their support.

We also thank all those who extended their support and co-operation while bringing out this Mini-Project Work.

<b>BABUREDDY B N</b>	<b>1SJ21IS009</b>
<b>IRFAN KAHAN I</b>	<b>1SJ21IS034</b>
<b>MOHAMMED MAZZ</b>	<b>1SJ21IS059</b>
<b>SRIDHARA M</b>	<b>1SJ22IS401</b>

# CONTENTS

Cover page	
Certificate	
Department Vision Mission	
Declaration	i
Abstract	ii
Acknowledgement	iii
Contents	iv-v
List of Figures	vi
List of Tables	vii

Chapter No	Chapter Title	Page No
<b>1</b>	<b>INTRODUCTION</b>	<b>1-4</b>
	1.1 Overview	
	1.2 Problem Statement	
	1.3 Significance and Relevance Work Objectives	
	1.4 Methodology	
	1.5 Organization of the Report	
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>5-6</b>
<b>3</b>	<b>SYSTEM REQUIREMENTS AND SPECIFICATION</b>	<b>7-9</b>
	3.1 System Requirement Specification	
	3.2 Specific Requirement	
	3.3 Hardware Specification	
	3.4 Software Specification	
	3.5 Functional Requirements	
	3.6 Non Functional Requirements	
	3.7 Performance Requirement	
<b>4</b>	<b>SYSTEM ANALYSIS</b>	<b>10-11</b>
	4.1 Existing System	
	4.1.1 Limitation	
	4.2 Proposed System	
	4.3 Advantages	

<b>5</b>	<b>SYSTEM DESIGN</b>	<b>12-16</b>
	5.1 Project Modules	
	5.2 Activity Diagram	
	5.3 UseCase Diagram	
	5.4 Dataflow Diagram	
	5.5 Sequence Diagram	
<b>6</b>	<b>IMPLEMENTATION</b>	<b>17-20</b>
	6.1 Algorithm	
	6.2 Pseudo-code module wise	
<b>7</b>	<b>TESTING</b>	<b>21-23</b>
	7.1 Methods of Testing	
	7.1.2 Unit Testing	
	7.1.2 Validation Testing	
	7.1.3 Functional Testing	
	7.1.4 Integration Testing	
	7.1.5 User Acceptance Testing	
	7.2 Test Cases	
<b>8</b>	<b>PERFORMANCE ANALYSIS</b>	<b>24-25</b>
<b>9</b>	<b>CONCLUSION AND FUTURE ENHANCEMENT</b>	<b>26-27</b>
	9.1 Conclusion	
	9.2 Future Enhancement	
	<b>BIBLIOGRAPHY</b>	<b>28</b>
	<b>APPENDIX</b>	<b>29-34</b>
	Appendix A:Screen Shots Appendix B:	
	Abbreviations	



## LIST OF FIGURES

<b>Figure No.</b>	<b>Name of the Figure</b>	<b>Page No.</b>
Figure 5.2	Activity Diagram	13
Figure 5.3	Use Case Diagram	14
Figure 5. 4	Data Flow Diagram	15
Figure 5.5	Sequence Diagram	16
Figure A.1	File Recovery Tool User Interface	29
Figure A.2	Select Drive	30
Figure A.3	Select Drive Directory	30
Figure A.4	Select File Type	31
Figure A.5	Scanning Process	31
Figure A.6	Deleted Files	32
Figure A.7	Recover Selected Files	32
Figure A.8	Recovered Files	33

## LIST OF TABLES

<b>Table No.</b>	<b>Table Name</b>	<b>Page No.</b>
Table 7.2.1	Browse_drive	22
Table 7.2.2	Browse_save	22
Table 7.2.3	Scan_for_deleted_files	22
Table 7.2.4	Recover_files	23
Table 7.2.5	Recover_selected_files	23

# **CHAPTER 1.**

## **INTRODUCTION**

# **CHAPTER-1:**

## **INTRODUCTION**

### **1.1 Overview**

Data integrity and accessibility are very crucial in the present digital world for every user and organization. Despite all the technological developments, it has been observed that data loss is a constant threat due to accidental deletion, hardware malfunctioning, software corruption, and cyber threats. The extent of loss from such data loss could be on a large financial and operational scale, impacting business operations, academic pursuits, and personal activities.

We intend to develop an online, multi-platform file system recovery tool to mitigate such challenges. It shall be efficient, reliable, and user-friendly in recovering lost or corrupted data. It will work across multiple operating systems-Windows and Linux-with ease, hence being accessible and usable for various categories of people. Conventionally, techniques to recover data are poorly accessible, user-unfriendly, and cross-platform incompatible. These are usually kinds of methods that involve special knowledge, and apart from being time-consuming, they are also comparatively complex to be used by people with little technical knowledge. Moreover, a number of already-existing data recovery solutions are developed to work within certain environments, something that, at times, reduces their usefulness to users who work across multiple platforms.

This tool will attempt to fill these gaps with a proposed all-in-one cross-platform solution designed to make the process of data recovery easier without loss of efficiency or reliability. The combination of advanced recovery algorithms with real-time monitoring functionality makes it easiest and most transparent for the users to launch and track the tasks of data recovery. Through real-time monitoring functionality, the recovery process will be constantly updated to the user in order to further instill confidence in the tool's performance. Furthermore, tight security protocols will be implemented in order to protect user data throughout the recovery process against possible violations of personal data privacy and security.

In other words, the proposed web-based cross-platform file system recovery tool will mark the momentous milestone of data recovery solutions by providing comprehensive, user-friendly, and reliable data retrieval for lost or corrupted data. The project thus highlights that recovery in the Digital Age matters a lot and that innovative solutions can make much difference toward improving habits of working with data and, hence, mitigating the negative impact caused by data loss.

## 1.2 Problem Statement

The problem is the frequent loss or accidental deletion of important files across various operating systems and file systems. Existing recovery tools are often platform-specific, limiting their usability and efficiency. A web-based cross-platform file system recovery tool is needed to provide a unified, accessible solution for users on Windows, macOS, and Linux, ensuring effective file recovery regardless of the file system in use.

## 1.3 Significance and Relevance

The development of a web-based cross-platform file system recovery tool is crucial in the digital age, where data integrity is essential for operations across various sectors such as healthcare, finance, education, and entertainment. Data loss can result in significant financial and operational disruptions, making efficient recovery solutions vital. This tool offers an intuitive and reliable solution for small businesses, educational institutions, and individual users, who often lack access to professional recovery services. By providing cross-platform compatibility and user-friendly design, it empowers users to independently recover lost data, minimizing downtime and potential losses. Furthermore, the tool aligns with current technological trends by emphasizing cross-platform interoperability, user-centric design, and robust data security. These features ensure its broad usability and protect user data during recovery, addressing critical needs in data management and recovery. Overall, this tool enhances resilience and efficiency in managing data loss in the digital era.

## 1.4 Work Objectives

- 1. User-Friendly Interface:** A web interface will be designed with a focus on allowing users to easily start and monitor tasks associated with data recovery.
- 2. Cross-Platform Compatibility:** The implemented tool, once executed, will be able to work in different operating systems, including Windows and Linux.
- 3. Advanced Recovery Algorithms:** Advanced recovery algorithms will be adopted by the implemented system to recover data with an increased possibility of success.
- 4. Provide for Real-Time Monitoring:** Progress updates and real-time logging detail the recovery process to the user.
- 5. Enhance Security Measures:** The recovery process is to be under a very tight security measure to protect the user data.

## 1.5 Methodology

The development of the web-based cross-platform file system recovery tool will follow a structured and systematic approach to ensure efficiency, reliability, and user-friendliness. The methodology comprises several key phases: requirement analysis, design, implementation, testing, and deployment. The requirement analysis phase involves gathering and documenting both functional and non-functional requirements. Stakeholders, including potential users, will be consulted to understand their needs and challenges. This step ensures that the tool addresses real-world data recovery issues across different operating systems. In the design phase, a scalable and modular architecture will be developed to support cross-platform compatibility. The tool's architecture will be based on a client-server model, where the client interface will be web-based, and the server will handle the core recovery processes. The design will also focus on creating an intuitive user interface that simplifies the recovery process.

The implementation phase will involve coding the core functionalities of the tool. Advanced recovery algorithms will be integrated to enhance the tool's effectiveness in retrieving lost or corrupted data. The web-based interface will be developed using HTML, CSS, and JavaScript, while the backend will be implemented using a suitable programming language and framework to ensure seamless operation across different operating systems. Rigorous testing will be conducted to validate the tool's performance, reliability, and security. The tool will be tested on various file systems (e.g., NTFS, HFS+, ext4) and operating systems (Windows, Linux) to ensure compatibility. Automated and manual test cases will be used to identify and rectify any bugs or issues. In the final phase, the tool will be deployed on a web server. Comprehensive user documentation and training materials will be provided to facilitate easy adoption. Continuous monitoring and maintenance will be conducted post-deployment to ensure the tool remains up-to-date and effective.

## 1.6 Organization of the Report

The mini project report is organized as follows:

- 1.6.1. **Chapter 1-Introduction**-This chapter tells about the problem statement, overview of the mini project, purpose, its objectives and its applicability with its theoretical outline.
- 1.6.2. **Chapter 2-Literature Survey**-Gives brief overview of the paper and the research sources that have been studied to establish through an understanding of the under consideration.
- 1.6.3. **Chapter 3-System Requirement Specification**-Discuss in detail about the different kind of requirement needed to successfully complete the project.
- 1.6.4. **Chapter 4-System Design**- It gives a detailed description of the methodology used in the study.
- 1.6.5. **Chapter 5-Implementation**-It will provide a detailed description of the software and hardware components used in the study.
- 1.6.6. **Chapter 6-Testing**-It will provide a detailed description of the testing and validation process used to evaluate the performance of the system.
- 1.6.7. **Chapter 7-System Analysis**- It will provide a detailed analysis of the performance of the system.
- 1.6.8. **Chapter 8-Performance Analysis**-It will provide a detailed analysis of the computational performance of the system.
- 1.6.9. **Chapter 9-Conclusion and Future Enhancement**-It will provide a summary of the key findings of the study and recommendations for future enhancements to the system.
- 1.6.10. **Bibliography** -contains the references to the referred paper that is mentioned in the literature survey.
- 1.6.11. **Appendix** contains screenshots of our model and supplementary information.

## **CHAPTER 2.**

### **LITERATURE SURVEY**



## CHAPTER-2:

### LITERATURE SURVEY

**[1]. Na Zhang, Ying Jiang, and Jia Wang, "The Research of Data Recovery on Windows File Systems,".**

This paper explores data recovery techniques specifically for Windows file systems, such as NTFS and FAT32. The authors analyze various methods used to recover lost or deleted data, including both software and hardware approaches. The study highlights the challenges associated with recovering data from corrupted or partially overwritten files and presents potential solutions to enhance recovery efficiency and accuracy. For the Recovery Tool, the insights from this research can be utilized to improve the robustness and reliability of file recovery processes on Windows platforms.( 2020)

**[2]. Rintaro Nagano, Ryuku Hisasue, Hiroshi Inamura, and Shigemi Ishida, "Recovery Method for Ransomware Encryption Attacks with File Extension Changing on File Server,".**

This study addresses the problem of ransomware attacks that change file extensions to encrypt data on file servers. The authors propose a novel recovery method that involves identifying and reversing the changes made to file extensions. They also discuss the challenges in distinguishing between benign file extension changes and those made by ransomware. The proposed method includes a detailed algorithm for detecting and recovering affected files. Implementing similar techniques in the Recovery Tool can enhance its ability to recover files affected by ransomware attacks, thereby providing an additional layer of security and data protection.( 2024)

**[3]. Dmitry Kuts, Sergey Porshnev, Maria Kuts, and Elena Popova, "The Peculiarities of Deleted Files Recovery in FAT32 File System,".**

This paper investigates the unique challenges of recovering deleted files from the FAT32 file system. The authors examine the structure of FAT32 and the implications it has on the recovery process. They discuss the limitations of traditional recovery tools when dealing with fragmented or partially overwritten files and propose improvements to enhance recovery rates. The findings from this research can be leveraged to optimize the Recovery Tool's file scanning and recovery algorithms, especially for devices using the FAT32 file system.( 2023)

**[4]. Prince Raj and Sapna Sinha,** "Enhancing File Recovery from Distributed File Systems (DFSs) Using Erasure Coding and Replication,".

The authors present methods to enhance file recovery in distributed file systems (DFSs) by utilizing erasure coding and replication techniques. They highlight the advantages of these methods in improving data redundancy and fault tolerance. The study includes detailed performance analyses of various erasure coding schemes and their impact on recovery times and success rates. Incorporating these techniques into the Recovery Tool can improve its ability to recover files from distributed environments, ensuring higher reliability and data availability.( 2024)

**[5]. Junghoon Oh, Sangjin Lee, and Hyunuk Hwang,** "Forensic Recovery of File System Metadata for Digital Forensic Investigation,".

This paper focuses on the forensic recovery of file system metadata, which is crucial for digital forensic investigations. The authors discuss methods for recovering and analyzing metadata from various file systems to reconstruct file activity and history. The research highlights the importance of metadata in understanding the context of file modifications, deletions, and access. The Recovery Tool can incorporate these forensic recovery techniques to provide more detailed information about recovered files, aiding in forensic investigations and data analysis.(2022)

**[6]. Piriform,** "Recuva: A Windows file recovery tool," Proceedings of the Windows Software Expo.

Recuva is a widely used file recovery tool for Windows, developed by Piriform. This paper discusses the features and capabilities of Recuva, including its ability to recover deleted files from various storage media. The study includes a performance evaluation of Recuva, highlighting its strengths and limitations. Insights from this evaluation can be used to benchmark and improve the Recovery Tool, ensuring it provides comparable or superior file recovery capabilities. Additionally, understanding Recuva's user interface and usability aspects can guide the design improvements for the Recovery Tool's user experience.( 2007)

## **CHAPTER 3.**

# **SYSTEM REQUIREMENTS AND SPECIFICATION**

## CHAPTER-3:

# SYSTEM REQUIREMENT AND SPECIFICATION

### 3.1 System Requirement And Specification

The file recovery tool must be a desktop application compatible with Windows and Linux. It should provide a user-friendly interface using Python's Tkinter library for browsing drives, selecting save directories, and displaying scanning and recovery progress. The tool must support various file types (.txt, .docx, .xlsx, .pptx, .pdf, .jpg, .png) and handle directory browsing through file dialogs. It should allow users to select and recover multiple files, with progress updates and status messages. Performance requirements include responsiveness during scanning and recovery, with multithreading to ensure smooth operation. Hardware requirements include a modern multi-core CPU, 4 GB of RAM, and sufficient storage. The application must be reliable, user-friendly, and ensure data integrity during recovery. The software requirements include Python 3.x, Tkinter, and relevant libraries for file operations and multithreading.

### 3.2 Specific Requirements

- **User Interface:** The tool provides a graphical interface using Tkinter for selecting drives, specifying save directories, and displaying recovery progress.
- **File Types:** Supports recovery for various file types such as .txt, .docx, .xlsx, .pptx, .pdf, .jpg, .png.
- **Directory Browsing:** Users can browse and select source drives and destination directories using file dialog windows.
- **File Selection:** Users can select multiple files from a list to be recovered.
- **Progress Indication:** Displays progress bars and status messages to inform users of the current operation status.

### 3.3 Hardware Requirements

#### 1. Client Machine:

- **Processor:** Modern multi-core CPU (e.g., Intel Core i5 or equivalent).
- **RAM:** Minimum 4 GB of RAM for smooth operation.
- **Storage:** Adequate storage for installation and recovery operations; SSD preferred for speed.

- **Network:** Basic internet connection for potential updates or support.

### 3.4 Software Requirements

#### 1. Operating System:

- **Windows:** Windows 10 or later.
- **Linux:** Ubuntu 20.04 or later.

#### 2. Programming Language and Libraries:

- **Language:** Python 3.x.
- **Libraries:** Tkinter for GUI, shutil for file operations, os for directory and file handling, threading for concurrent operations.

#### 3. Development Tools:

- **IDE:** Any Python-compatible IDE (e.g., PyCharm, Visual Studio Code).
- Version Control:** Git for source code management.

### 3.5 Functional Requirements

#### 1. Browse Directories

- **Drive Selection:** Users can select the drive to scan for deleted files.
- **Save Directory Selection:** Users can choose a directory where recovered files will be saved.

#### 2. File Scanning

- **File Type Selection:** Users can choose the type of files to scan for.
- **Scanning Operation:** The tool scans the selected drive for files matching the specified type.

#### 3. File Recovery

- **Select Files:** Users can select files from the scan results to recover.
- **Recover Files:** The selected files are copied to the specified save directory.

#### 4. Progress and Status Updates

- **Scanning Progress:** Displays progress and status of the scanning process.
- **Recovery Progress:** Shows the progress of file recovery operations.

#### 5. Error Handling

- **Directory Validation:** Ensures selected directories are valid and accessible.
- **File Existence Check:** Validates the existence of files before attempting recovery.

### 3.6 Non-Functional Requirements

#### 1. Performance

- **Responsiveness:** The application should remain responsive during scanning and recovery operations due to multithreading.

#### 2. Usability

- **User-Friendly Interface:** Simple and intuitive interface with clear navigation and instructions.
- **Error Messages:** Informative error messages to guide users in case of issues.

#### 3. Reliability

- **Consistent Operation:** Reliable performance in scanning and recovering files with minimal downtime.

#### 4. Security

- **Data Integrity:** Ensures that files are not corrupted during the recovery process.

### 3.7 Performance Requirements

#### 1. UI Responsiveness:

- **Real-Time Interaction:** The user interface should remain responsive during scanning and recovery operations, allowing users to interact with other parts of the application without lag. This includes browsing directories, selecting files, and viewing progress updates.

#### 2. Scanning Efficiency:

- **Speed:** The scanning process should be efficient, completing within a reasonable time based on the size of the selected drive and the number of files. The tool should handle large volumes of files and directories without significant delays.

#### 3. Recovery Speed:

- **File Transfer:** File recovery should be completed swiftly, with minimal delay from the time of selection to the completion of the recovery. The performance should scale with the number of files selected for recovery.

#### 4. Resource Management:

- **CPU and Memory Usage:** The application should optimize CPU and memory usage, avoiding excessive resource consumption. Multithreading should be used effectively to ensure that scanning and recovery processes do not overload the system.

## **CHAPTER 4.**

## **SYSTEM ANALYSIS**

## CHAPTER-4:

### SYSTEM ANALYSIS

#### 4.1 Existing System

The existing system appears to be a recovery tool, possibly a web application, given the presence of a Python script (app.py) and directories typically used in web development (static, templates).

##### Components:

app.py: Likely the main application script.

static: Contains static files such as CSS, JavaScript, and images.

templates: Contains HTML templates.

.vscode: Configuration files for the Visual Studio Code editor.

##### 4.1.1 Limitations

Based on the provided structure and typical issues in similar systems, potential limitations might include:

- **Scalability:** If the application is not designed to handle high traffic, it may face performance issues.
- **Security:** Potential vulnerabilities if best practices are not followed, such as proper input validation and secure data handling.
- **Maintenance:** Lack of documentation or modular design might make future updates and maintenance challenging.
- **User Interface:** The usability and design might not be optimal if not properly tested and designed.

#### 4.2 Proposed System

To address the potential limitations, the proposed system improvements might include:

- **Enhanced Scalability:** Implement load balancing, database optimization, and efficient resource management.
- **Improved Security:** Ensure proper input validation, use secure communication protocols, and implement authentication and authorization mechanisms.
- **Better Maintainability:** Refactor code to be more modular, add comprehensive documentation, and implement automated testing.
- **User Interface Enhancement:** Improve the UI/UX design, conduct user testing, and make the interface more intuitive and accessible.



### 4.3 Advantages

Implementing the proposed improvements can offer several advantages:

- **Increased Performance:** Enhanced scalability measures can ensure the application performs well under high traffic.
- **Enhanced Security:** Improved security measures can protect user data and prevent malicious attacks.
- **Easier Maintenance:** Modular code and proper documentation can simplify updates and troubleshooting.
- **Better User Experience:** An improved user interface can increase user satisfaction and engagement.

## **CHAPTER 5.**

## **SYSTEM DESIGN**

## CHAPTER-5:

### SYSTEM DESIGN

#### 5.1 Project Modules

1. **User Interface Module:** Is responsible for rendering the interface by loading and displaying the HTML, CSS, and JavaScript files. It handles user interactions through event listeners for actions like button clicks and file selections. This module also provides feedback to users via alerts, pop-ups, or messages to indicate the status of operations such as file recovery success or errors.
2. **File Browser Module:** It enables users to navigate the file system. It enumerates all available drives, displays them to the user, and allows for dynamic directory navigation. Additionally, it includes file filtering capabilities based on user-defined criteria like file type, size, or deletion date, enhancing the user's ability to locate specific files.
3. **File Scanning Module:** Is tasked with performing an initial scan to detect and list deleted files. It continuously updates the progress bar and scanning status, providing real-time feedback to the user. This module also includes error handling to manage issues such as inaccessible drives or directories, ensuring the user is informed of any problems.
4. **File Recovery Module:** It allows users to select which deleted files to recover. It then executes the recovery process, restoring the selected files to a specified directory. An integrity check is performed to verify that the recovered files are not corrupted, ensuring data quality and reliability.
5. **Progress Tracking Module:** It manages the progress bar, reflecting the current status of scanning and recovery operations. It displays detailed information such as the number of files scanned and recovered, as well as any errors encountered. Upon completion of the processes, it notifies the user, providing a comprehensive overview of the operation's outcome.

## 5.2 Activity Diagram

The activity diagram illustrates the workflow of the web-based file recovery tool, detailing user interactions and system processes. It starts with the user launching the application and displaying the interface, allowing them to select a drive and save directory. The user initiates a scan, triggering the file scanning module to detect deleted files, which are then listed for user review. Once files are selected, the recovery process begins, copying files to the specified directory. The diagram concludes with feedback on the recovery status, ensuring a clear and structured overview of the tool's operations.

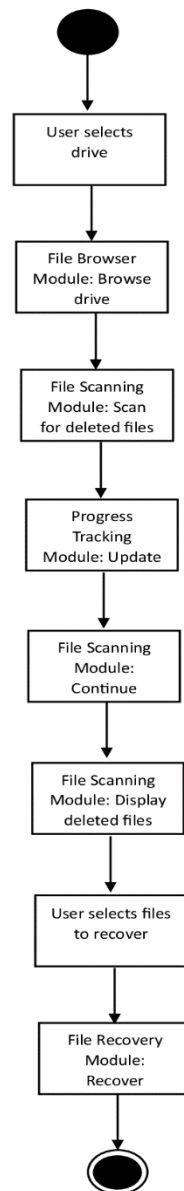
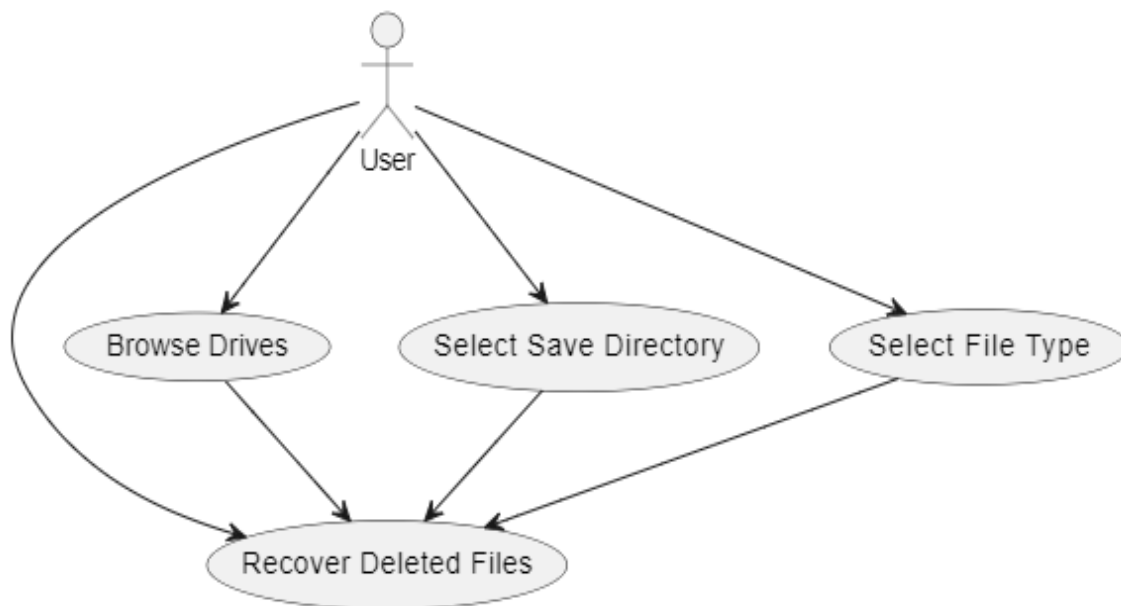


Figure 5.2 : Activity Diagram

### 5.3 Use Case Diagram

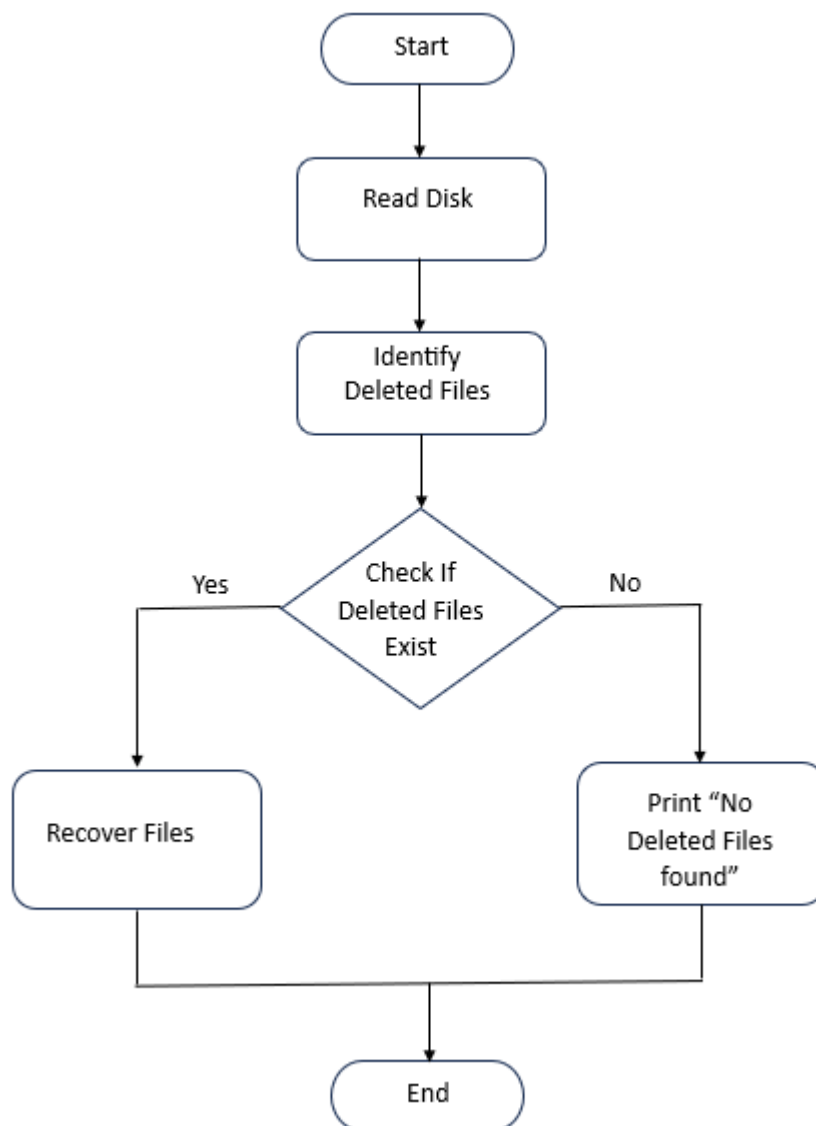
The use case diagram serves as a visual representation of how users interact with the file recovery tool, highlighting the primary tasks and functionalities available within the application. It outlines a range of actions that users can perform, such as launching the tool, selecting specific drives or partitions to scan, specifying directories where files were lost, and choosing the types of files they wish to recover. The diagram also details the processes initiated by the user, including starting scans to detect lost or deleted files, recovering selected files, and viewing the results of these operations. Additionally, it depicts the feedback mechanisms provided by the tool, such as notifications on the status of scans and recovery attempts. By mapping out these interactions, the use case diagram clarifies how the tool facilitates file recovery tasks, helping to ensure that both the user interface and system responses align with user needs and expectations. This comprehensive overview aids in identifying key functionalities and improving the overall user experience by clearly defining the sequence of actions and the expected outcomes within the tool.



**Figure 5.3 : Use Case Diagram**

## 5.4 Dataflow Diagram

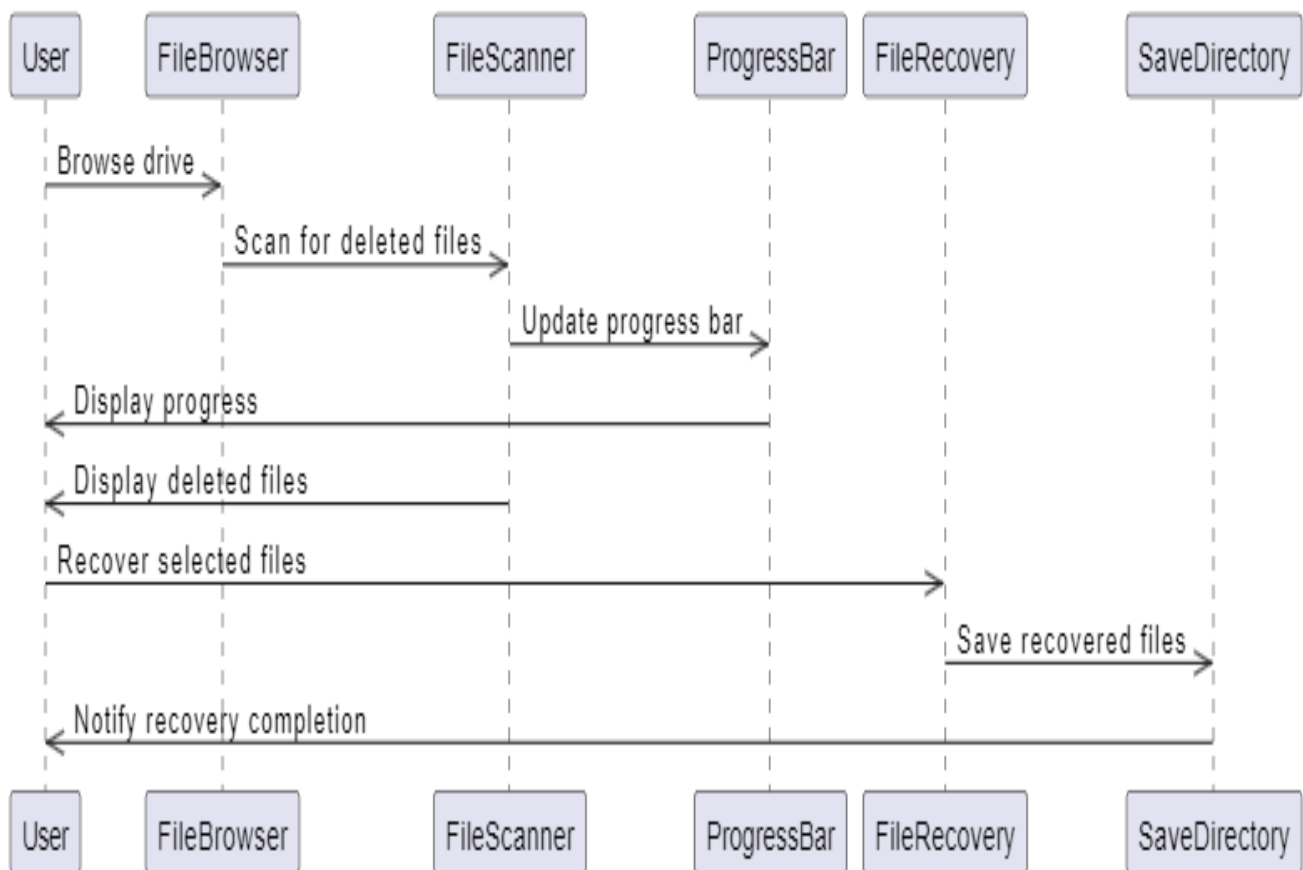
The data flow diagram (DFD) depicts the flow of information within the file recovery tool, illustrating how data moves between different components. It shows how user inputs are processed through various modules, including file browsing, scanning, and recovery. The diagram highlights data sources, such as user selections and system responses, and data stores, like the list of deleted files and recovered files. It provides a visual representation of how data is transformed and handled, helping to understand system processes and interactions. This overview is crucial for designing and optimizing the application's data handling and workflow.



**Figure 5.4 : Data Flow Diagram**

## 5.5 Sequence Diagram

The sequence diagram illustrates the chronological sequence of interactions between the user and the file recovery tool. It details the order of messages exchanged between objects or components, such as the user interface, file scanning module, and file recovery module. The diagram captures the flow of events from initiating a scan to recovering files and provides a clear visualization of how each component responds to user actions over time. This helps in understanding the dynamic behavior of the system and ensuring that processes are executed in the correct sequence. It is essential for designing and debugging the tool's functional interactions.



**Figure 5.5 : Sequence Diagram**

## **CHAPTER 6.**

## **IMPLEMENTATION**



## CHAPTER-6:

## IMPLEMENTATION

### 6.1 Algorithm

#### Step1 : Initialize Application:

- Create a Tkinter main window (root).
- Initialize RecoveryTool class with root.

#### Step2 : Setup User Interface:

- Define main UI components, including labels, entry fields, buttons, listbox, and progress bar.
- Arrange components using Tkinter layout methods.

#### Step3 : User Actions:

- **Browse Drive:**
  - Open a file dialog to select a drive directory.
  - Set the selected path in the drive entry field.
- **Browse Save Directory:**
  - Open a file dialog to select a save directory.
  - Set the selected path in the save entry field.
- **Scan for Deleted Files:**
  - Validate if a drive path is provided.
  - Display scanning status and initiate a new thread to scan for files.
- **Recover Selected Files:**
  - Validate if a save path is provided and files are selected.
  - Display recovery status and initiate a new thread to recover selected files.

#### Step 4 : Scanning for Files:

- Traverse directories and files in the selected drive.
- Filter files based on the selected file type.
- Collect file paths and modified times.
- Sort files by modified time in descending order.
- Update the file listbox with scanned files and their timestamps.

#### Step 5 : Recovering Files:

- For each selected file, get the file path and name.
- Copy the file to the specified save directory.
- Handle cases where files might not be found.

**Step 6 : Update UI:**

- During scanning and recovery, update the status label and progress bar.
- Once scanning or recovery is completed, update the status and progress to indicate completion.

**Step 7 : Exit Application:**

- Close the application window and terminate the program when finished.

**6.2 Pseudo-code**

```
# Import necessary libraries and modules
Import Flask, request, jsonify, render_template from flask
Import CORS from flask_cors
Import os, shutil
Import datetime
Import Tk, filedialog from tkinter

# Initialize Flask application
app = Flask(__name__)

# Enable Cross-Origin Resource Sharing (CORS) for all routes
CORS(app)

# Function to browse and select a folder using Tkinter
Function browse_folder():
    Initialize Tkinter root
    Hide the main window
    Get folder path using filedialog.askdirectory()
    Destroy the Tkinter instance
    Return folder path

# Route for index page
@app.route('/')
Function index():
    Return rendered 'index.html' template
```

# Route to browse drive

```
@app.route('/browse_drive', methods=['GET'])
```

Function browse\_drive():

    Get folder path by calling browse\_folder()

    Return JSON response with folder path

# Route to browse save directory

```
@app.route('/browse_save', methods=['GET'])
```

Function browse\_save():

    Get folder path by calling browse\_folder()

    Return JSON response with folder path

# Route to scan files in a specified drive

```
@app.route('/scan_files', methods=['GET'])
```

Function scan\_files():

    Get 'drive' and 'fileType' parameters from request

    Initialize empty list for deleted\_files

    # Walk through the directory structure

    For each root, dirs, and files in os.walk(drive):

        For each file in files:

            If file ends with file\_type:

                Get file path

                Get modified time of the file

                Append file information (path and modified time) to deleted\_files list

    Sort deleted\_files by modified time in descending order

    Return JSON response with sorted deleted\_files

# Route to recover files

```
@app.route('/recover_files', methods=['POST'])
```

Function recover\_files():

    Get JSON data from request

    Extract 'saveDir' and 'files' from data

    # Loop through each file information

    For each file\_info in files:

Split timestamp and file path from file\_info

Normalize and get absolute file path

If file exists at the path:

    Get file name

    Construct save file path by joining saveDir and file\_name

    Copy file to save file path

Return JSON response with status 'success'

# Run the Flask application in debug mode

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

## **CHAPTER 7.**

### **TESTING**

## **CHAPTER-7:**

### **TESTING**

#### **7.1 Methods of Testing**

##### **7.1.1 Unit Testing**

Unit testing involves testing individual components or methods of a software application to ensure they perform as expected in isolation. Each function or method is tested with a range of inputs to confirm its behaviour matches the expected output. In the context of the Recovery Tool, unit testing would involve testing methods like `scan_for_files` and `recover_files` independently to verify they correctly handle file scanning, sorting, and recovery processes. This ensures that each piece of functionality works correctly on its own before integrating it with other parts of the application.

##### **7.1.2 Validation Testing**

Validation testing assesses whether the software meets the requirements and expectations of the end users. This involves comparing the actual outcomes of the application to the specified requirements. For the Recovery Tool, validation testing would involve checking that the tool correctly scans for and lists deleted files according to the chosen file type and allows users to recover selected files to a specified directory. This ensures the application fulfills its intended purpose and delivers a satisfactory user experience.

##### **7.1.3 Functional Testing**

Functional testing evaluates the software's functionality against its requirements. This type of testing focuses on user interactions and ensures that all features operate correctly. For the Recovery Tool, functional testing would include verifying that the "Browse" buttons correctly open file dialogs, the "Scan for Deleted Files" button initiates the scanning process, and the "Recover Selected Files" button correctly recovers files to the specified directory. This ensures that each feature performs its designated function as expected.

##### **7.1.4 Integration Testing**

Integration testing examines how different components of the software work together. This testing is critical for identifying issues that may arise when combining multiple units or modules. In the case of the Recovery Tool, integration testing would involve

checking that the file scanning results are correctly displayed in the listbox and that file recovery processes integrate properly with the file system operations. This ensures that the individual components work together seamlessly to provide a cohesive user experience.

### 7.1.5 User Acceptance Testing

User acceptance testing (UAT) involves testing the software with real users to ensure it meets their needs and expectations. This type of testing is conducted in a real-world environment and focuses on the overall user experience. For the Recovery Tool, UAT would involve having actual users interact with the application to ensure it is intuitive, easy to use, and effectively performs file recovery tasks. Feedback from this testing phase helps identify any areas for improvement before the final release.

## 7.2 Test Cases

**Table 7.2.1: Browse\_drive**

Test case	TC01
Description	Open file dialog and set drive path
Input	User selects "C:\\" or "D:\\"
Expected Output	Entry field updated to "C:\\" or "D:\\"
Actual Output	Entry field updated to "C:\\" or "D:\\"
Status	Pass

**Table 7.2.2 Browse\_save**

Test case	TC02
Description	Open file dialog and set save path
Input	User selects "C:\\" or "D:\\"
Expected Output	Entry field updated to "C:\\" or "D:\\"
Actual Output	Entry field updated to "C:\\" or "D:\\"
Status	Pass

**Table 7.2.3 : Scan\_for\_deleted\_files**

Test case	TC03
Description	Scan files and sort by modification time
Input	Drive path "C:\", file type ".txt", ".jpg", ".docx", ".pdf", ".pptx"
Expected Output	List of files sorted by modification time
Actual Output	List of files sorted by modification time
Status	Pass

**Table 7.2.4 : Recover\_files:**

Test case	TC04
Description	Recover files from list to "D:\" or "C:\"
Input	Save path "D:\" or "C:\", list of files
Expected Output	Files copied to the specified save directory
Actual Output	Files copied to the specified save directory
Status	Pass

**Table 7.2.5 : Recover\_selected\_files:**

Test case	TC05
Description	Recover selected files to "D:\" or "C:\"
Input	Save path "D:\" or "C:\", selected files list
Expected Output	Status label updated to "Recovery completed"; Progress bar at 100%
Actual Output	Status label updated to "Recovery completed"; Progress bar at 100%
Status	Pass



**CHAPTER 8.**  
**PERFORMANCE ANALYSIS**

## **CHAPTER-8:**

### **PERFORMANCE ANALYSIS**

#### **1. Efficiency of File Scanning:**

The efficiency of file scanning in the RecoveryTool depends on several factors, including the number of files and directories in the selected drive and the performance of the underlying file system. The `scan_for_files` method traverses the directory structure and checks file extensions, which can be time-consuming for large drives with many files. Performance optimizations such as multi-threading or asynchronous processing could be considered to handle large volumes of data more efficiently. Monitoring the time taken for scanning operations and optimizing the algorithm can significantly enhance the tool's performance.

#### **2. Resource Utilization:**

The RecoveryTool's resource utilization, including CPU and memory usage, is influenced by the file scanning and recovery processes. During intensive operations, such as scanning large directories or recovering multiple files simultaneously, the tool may consume significant CPU and memory resources. Implementing efficient data structures and minimizing memory overhead can help manage resource usage effectively. Profiling tools can be used to analyze and optimize resource consumption to ensure smooth operation without excessive strain on the system.

#### **3. User Interface Responsiveness:**

The responsiveness of the user interface (UI) is crucial for maintaining a positive user experience. Long-running operations, such as scanning or recovery, should not cause the UI to become unresponsive. Using threading for background operations, as implemented in the RecoveryTool, helps keep the UI responsive. However, additional measures such as updating the UI with progress indicators and ensuring that long-running tasks do not block the main thread are essential for maintaining responsiveness and preventing user frustration.

#### **4. Scalability :**

Scalability refers to the tool's ability to handle increasing amounts of data or user load. As the size of the scanned drive or the number of files to recover grows, the RecoveryTool should scale efficiently. Evaluating the tool's performance with varying data sizes and configurations helps identify potential bottlenecks. Optimization techniques such as indexing, caching, or using more efficient algorithms can enhance scalability. Ensuring the tool performs well under different load conditions is crucial for its long-term usability and effectiveness.

**6. Disk I/O Performance :**

The performance of disk input/output (I/O) operations directly affects the Recovery Tool's overall speed and efficiency. File reading, scanning, and writing processes are dependent on disk I/O performance. Monitoring and optimizing disk I/O operations, such as minimizing unnecessary read/write operations or using buffered I/O, can improve performance. Understanding the impact of different file systems and storage media on I/O performance is also essential for ensuring optimal tool performance.

**CHAPTER 9.**  
**CONCLUSION AND FUTURE ENHANCEMENT**

## CHAPTER-9:

### CONCLUSION AND FUTURE ENHANCEMENT

#### 9.1 Conclusion:

The RecoveryTool has been designed to provide an efficient and user-friendly solution for recovering deleted files from various file systems. The tool incorporates a comprehensive set of features, including browsing drives and save directories, selecting file types, scanning for deleted files, and recovering selected files. The use of threading ensures that the user interface remains responsive during long-running operations, enhancing the overall user experience. The integration of various error-handling mechanisms ensures that the tool operates reliably under different conditions, providing clear feedback to users in case of issues.

The literature surveys and performance analysis provide valuable insights into optimizing the RecoveryTool further. Techniques and algorithms from contemporary research can be integrated to improve the efficiency, reliability, and scalability of the tool. By addressing the identified performance and usability challenges, the RecoveryTool can be made more robust and effective in handling large data sets and complex recovery scenarios.

#### 9.2 Future Enhancement:

##### 1. Advanced File Recovery Algorithms:

**Enhancement:** Implement advanced file recovery algorithms, such as those used in contemporary research on Windows file systems and ransomware recovery methods.

**Benefit:** This would improve the accuracy and success rate of recovering files, even in challenging scenarios involving corrupted or partially overwritten data.

##### 2. Support for Additional File Systems:

**Enhancement:** Extend support to include more file systems such as ext4, HFS+, APFS, and others.

**Benefit:** Expanding the range of supported file systems will make the RecoveryTool versatile and useful for a broader audience, including users of Linux.

##### 3. Improved Performance Optimization:

**Enhancement:** Optimize the file scanning and recovery processes using techniques such as multi-threading, asynchronous processing, caching, and efficient data structures.

**Benefit:** Enhancements in performance will reduce the time taken for scanning and recovery, especially on large drives, and improve overall responsiveness.

#### 4. Enhanced User Interface:

**Enhancement:** Redesign the user interface to be more intuitive and user-friendly, incorporating feedback mechanisms, better progress indicators, and clear error messages.

**Benefit:** A more intuitive UI will enhance user satisfaction and make the tool easier to use, especially for non-technical users.

#### 5. Scalability Improvements:

**Enhancement:** Implement scalability improvements such as load balancing, efficient algorithms, and resource management techniques to handle large data volumes and user loads.

**Benefit:** These improvements will ensure the tool remains efficient and responsive even when dealing with extensive data sets or multiple concurrent users.

#### 6. Cloud Storage Integration:

**Enhancement:** Add support for recovering files from cloud storage services such as Google Drive, Dropbox, and OneDrive.

**Benefit:** Integration with cloud storage services will expand the tool's capabilities, allowing users to recover files stored in the cloud, making it more versatile and comprehensive.

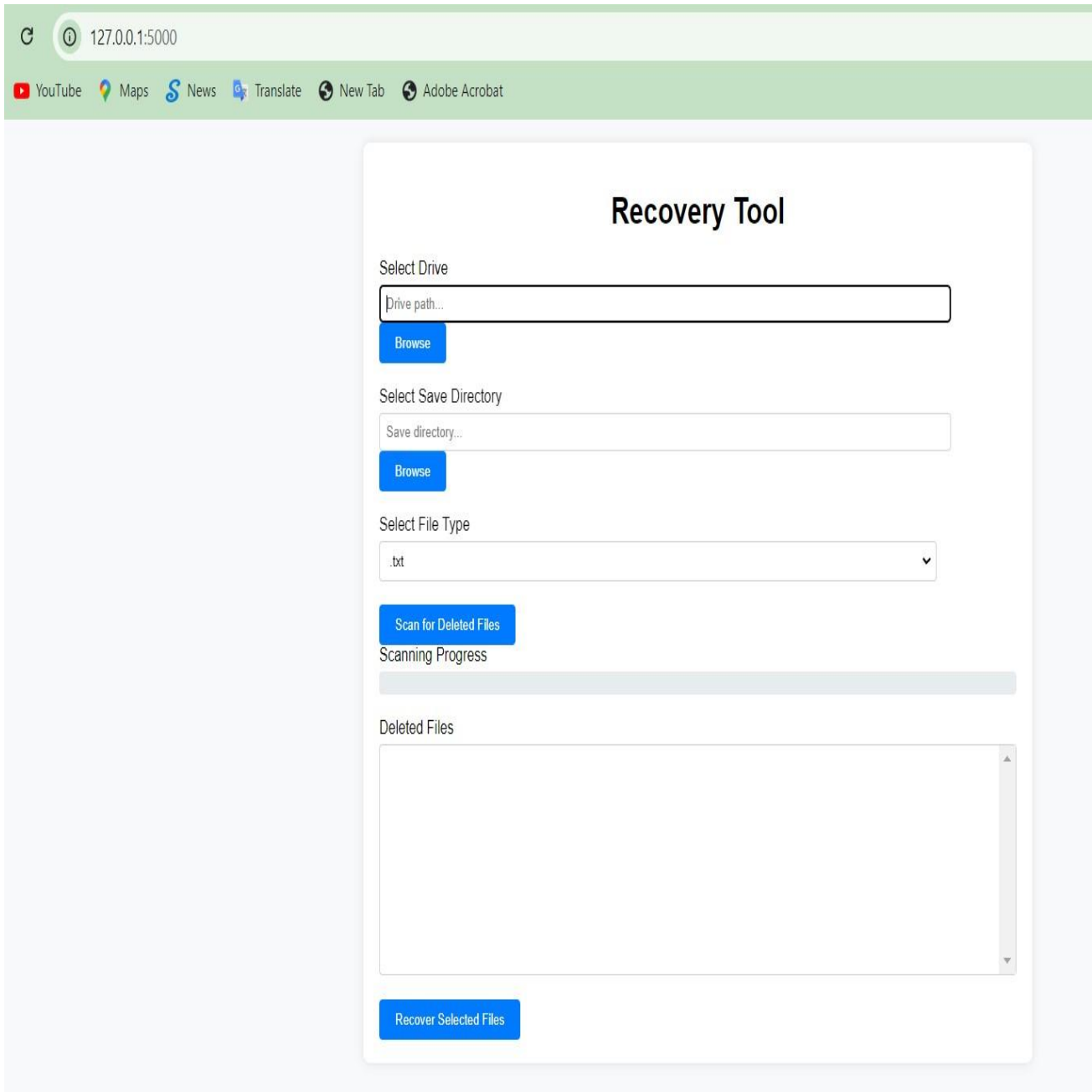
## **BIBLIOGRAPHY**

- [1].Na Zhang; Ying Jiang; Jia Wang. "The Research of Data Recovery on Windows File Systems." 2020.
- [2]. Rintaro Nagano; Ryuku Hisasue; Hiroshi Inamura; Shigemi Ishida. "Recovery Method for Ransomware Encryption Attacks with File Extension Changing on File Server." 2024.
- [3]. Dmitry Kuts; Sergey Porshnev; Maria Kuts; Elena Popova. "The Peculiarities of Deleted Files Recovery in FAT32 File System." 2023.
- [4]. Prince Raj; Sapna Sinha. "Enhancing File Recovery from Distributed File Systems (DFSs) Using Erasure Coding and Replication." 2024.
- [5]. Junghoon Oh; Sangjin Lee; Hyunuk Hwang. "Forensic Recovery of File System Metadata for Digital Forensic Investigation." 2022.
- [6] Piriform. Recuva: A Windows file recovery tool. Proceedings of the Windows Software Expo,2007.

## APPENDIX

### Appendix A:

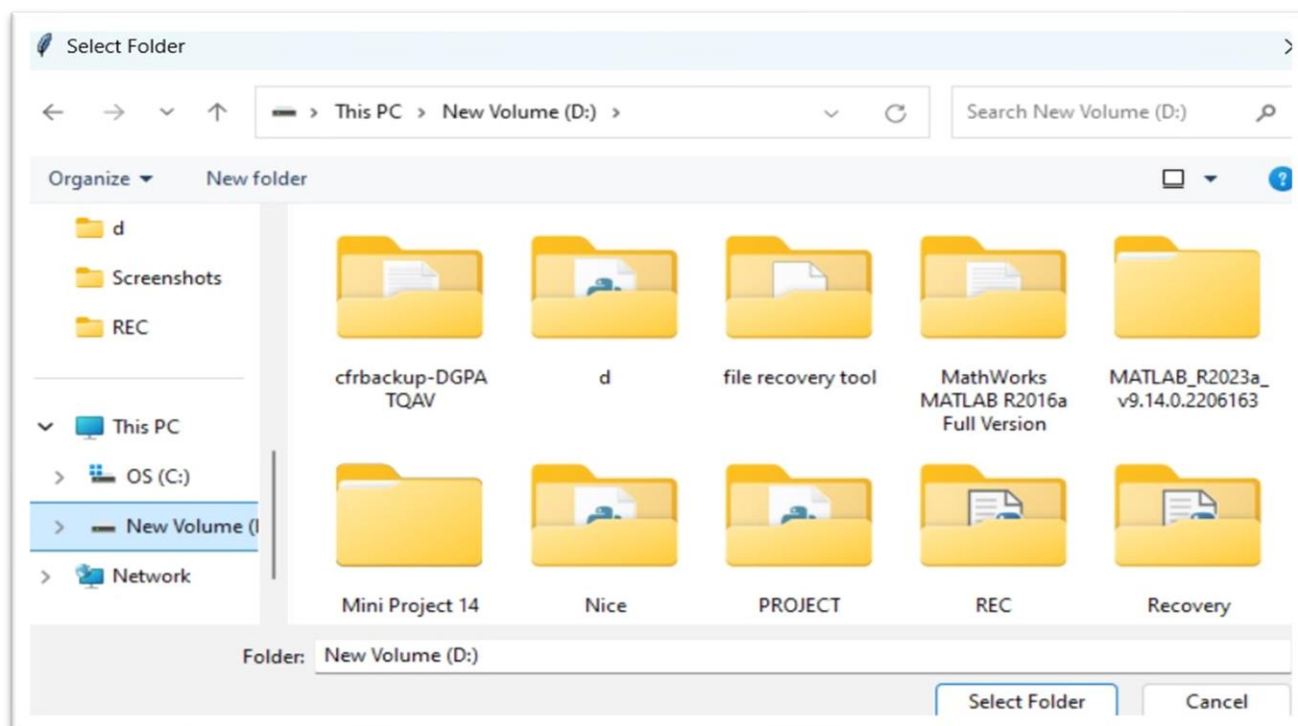
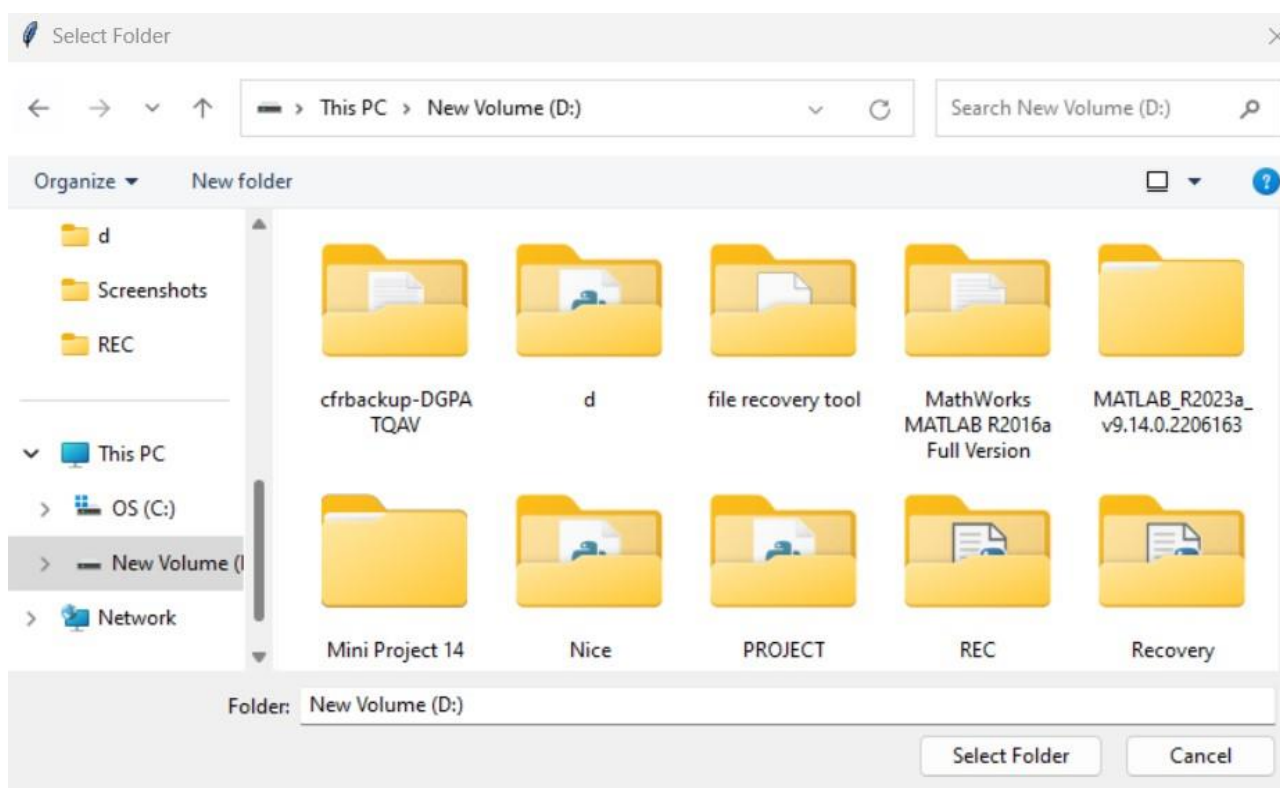
#### Snap Shots

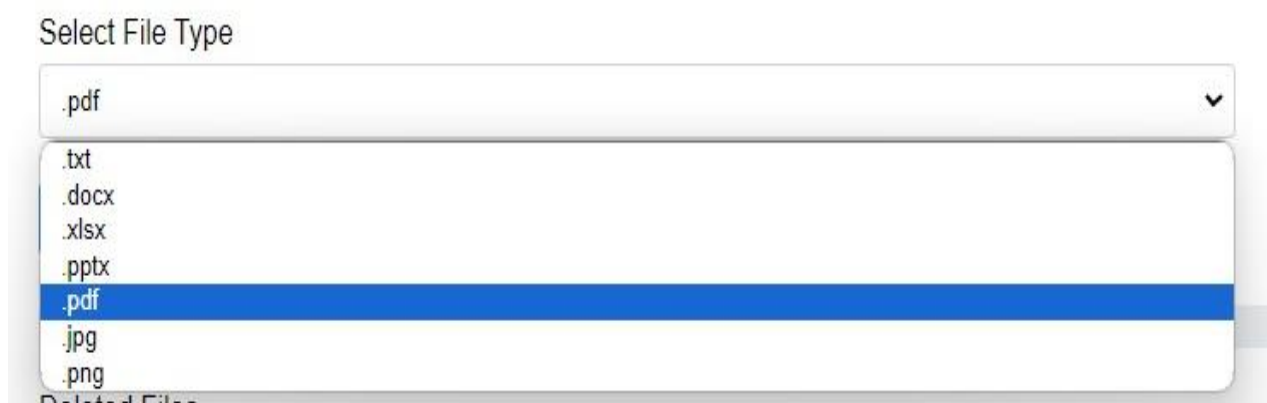


The screenshot displays a web browser window with the address bar showing '127.0.0.1:5000'. The browser's tab bar includes links to YouTube, Maps, News, Translate, New Tab, and Adobe Acrobat. The main content area features a 'Recovery Tool' interface. This interface includes three input fields: 'Select Drive' with a 'Drive path...' placeholder and a 'Browse' button; 'Select Save Directory' with a 'Save directory...' placeholder and a 'Browse' button; and 'Select File Type' with a dropdown menu currently set to '.txt'. Below these fields is a blue 'Scan for Deleted Files' button. Underneath the button is a 'Scanning Progress' section with a horizontal progress bar. At the bottom of the interface is a 'Deleted Files' section with a large, empty list box and a 'Recover Selected Files' button.

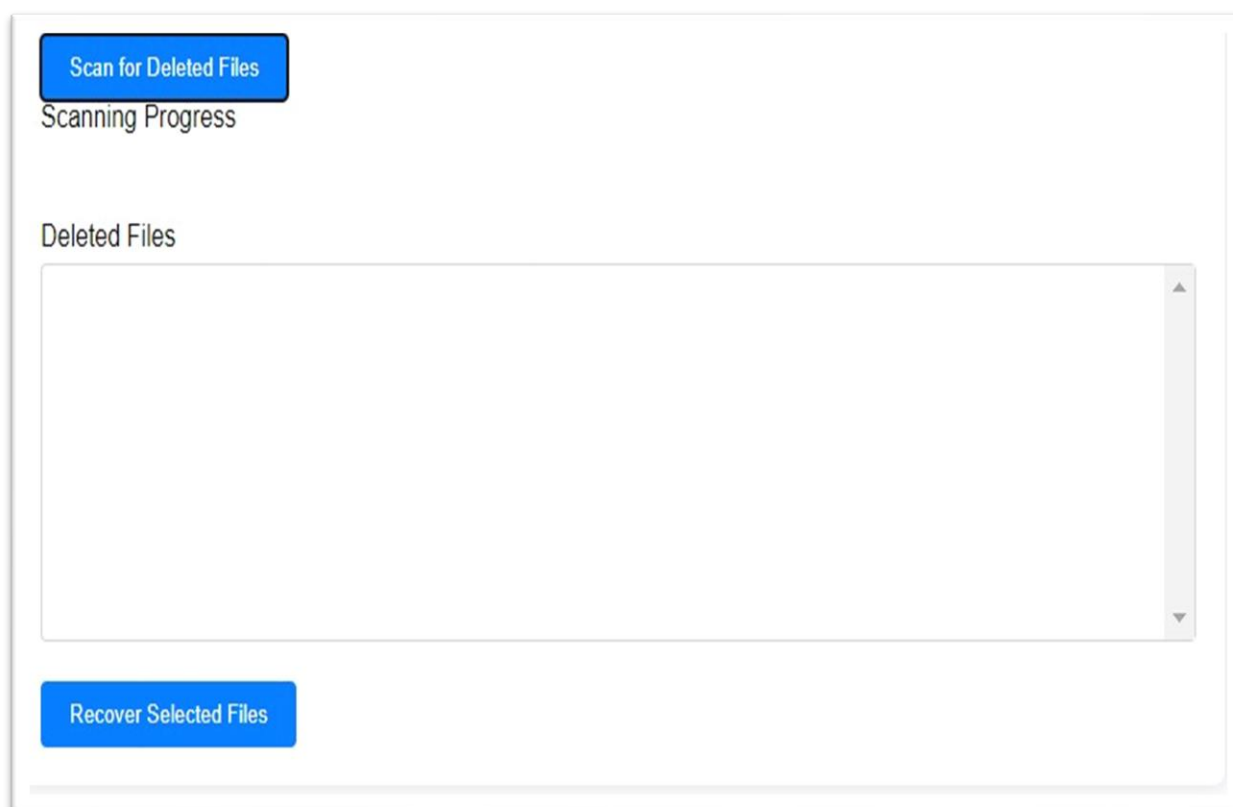
**Figure A.1 :File Recovery Tool User Interface**



**Figure A.2 : Select Drive****Figure A.3 : Select Drive Directory**



**Figure A.4 : Select File Type**



**Figure A.5 : Scanning Process**

Scan for Deleted Files

Scanning Progress

Scanning completed

Deleted Files

2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IFH4QF6.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$INASCL6.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IP2FAJY.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IWVP1FO.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RFH4QF6.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RNASCL6.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RP2FAJY.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RWVP1FO.pdf  
 2024-07-28 01:10:58 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IP2LNLA.pdf  
 2024-07-28 01:10:58 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IZ48UQ8.pdf  
 2024-07-28 01:10:42 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RP2LNLA.pdf

Recover Selected Files

**Figure A.6 : Deleted Files**

Scan for Deleted Files

Scanning Progress

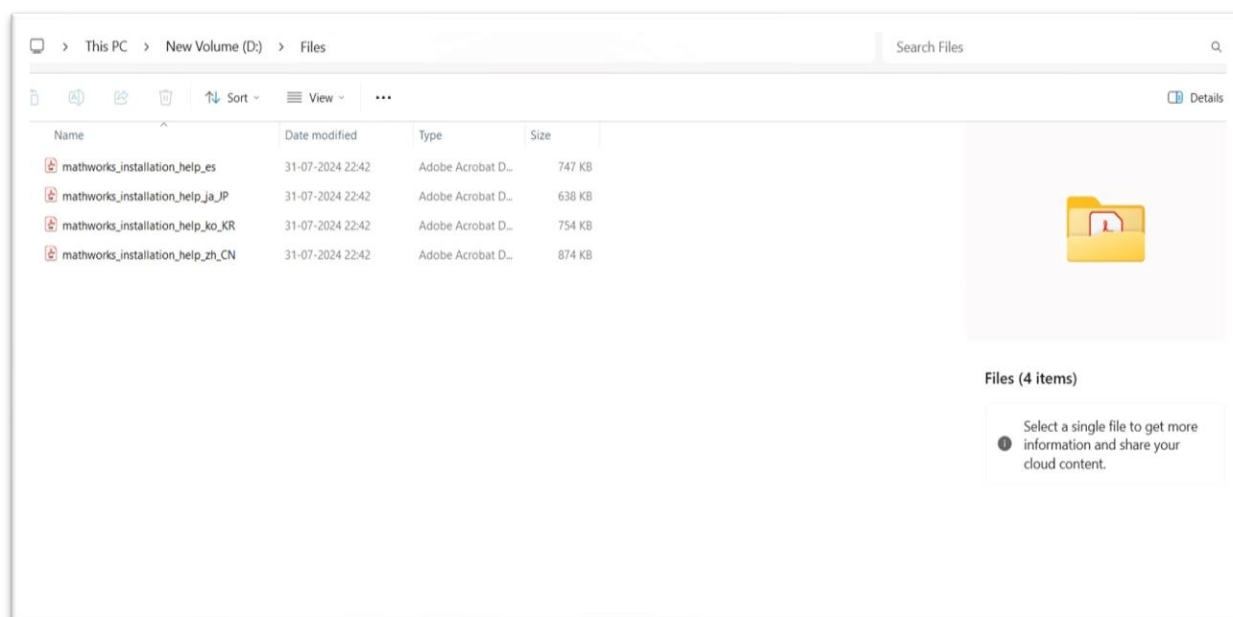
Recovery completed

Deleted Files

2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IFH4QF6.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$INASCL6.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IP2FAJY.pdf  
 2024-07-31 00:43:11 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IWVP1FO.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RFH4QF6.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RNASCL6.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RP2FAJY.pdf  
 2024-07-31 00:42:40 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RWVP1FO.pdf  
 2024-07-28 01:10:58 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IP2LNLA.pdf  
 2024-07-28 01:10:58 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$IZ48UQ8.pdf  
 2024-07-28 01:10:42 - D:/RECYCLE.BIN/S-1-5-21-2355863938-832309245-1826106129-1001/\$RP2LNLA.pdf

Recover Selected Files

**Figure A.7 : Recover Selected Files**



**Figure A.8 : Recovered Files**

## APPENDIX B:

### Abbreviations

The list of abbreviations frequently used in the chapters are mentioned in the table below:

SL NO.	NAME	ABBREVIATIONS
1	HTML	HyperText Markup Language
2	CSS	Cascading Style Sheets
3	JS	JavaScript
4	HTTPS	Hypertext Transfer Protocol Secure
5	HTTP	Hypertext Transfer Protocol
6	SSD	Solid State Drive
7	HDD	Hard Disk Drive
8	GUI	Graphical User Interface
9	EXT	Extensible File System
10	JSON	JavaScript Object Notation