A Course Based Project Report on

**File Compression and Decompression**

Submitted to the

**Department of Computer Science and Engineering**

in partial fulfilment of the requirements for the completion of course

Operating Systems (22PC1IT202)

BACHELOR OF TECHNOLOGY

IN

**Computer Science and Engineering**

Submitted by

**A. RITHISH REDDY 23071A05D7**

**G. SRIVIDYA 23071A05F8**

**HANVITH SAI ALLA 23071A05G2**

**L.VAMSHI 23071A05H0**

**A. SANATH 24075A0515**

Under the guidance of

**Dr. D N VASUNDHARA**

**(Course Instructor)**

Assistant Professor, Department of CSE, VNRVJIET



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with ‘A++’ Grade, NBA

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with ‘A++’ Grade, NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses, Approved by AICTE, New Delhi, Affiliated to JNTUH, Recognized as “College with Potential for Excellence” by UGC, ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

This is to certify that the project report entitled “**File Compression and Decompression**” is a bonafide work done under our supervision and is being submitted by **A. RITHISH (23071A05D7), G. SRIVIDYA (23071A05F8), HANVITH SAI ALLA (23071A05G2), L. VAMSHI (23071A05H0), A. SANATH (24075A0515)** in partial fulfilment for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering, of the VNRVJIET, Hyderabad during the academic year 2024-2025.

**Course Instructor Name DR. D N VASUNDHARA**  Assistant Professor

Department of CSE

**Course based Projects Reviewer**

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

An Autonomous Institute, NAAC Accredited with ‘A++’ Grade,

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**DECLARATION**



We declare that the course-based project work entitled “**File Compression and Decompression**” submitted in the Department of Computer Science and Engineering, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in** **Computer Science and Engineering** is a bonafide record of our own work carried out under the supervision of **DR. D N VASUNDHARA, Assistant Professor, Department of CSE, VNRVJIET.** Also, we declare that the matter embodied in this thesis has not been submitted by us in full or in any part thereof for the award of any degree/diploma of any other institution or university previously.

Place: Hyderabad.

**A. RITHISH REDDY 23071A05D7**

**G. SRIVIDYA 23071A05F8**

**HANVITH SAI ALLA 23071A05G2**

**L.VAMSHI 23071A05H0**

**A. SANATH 24075A0515**

**ACKNOWLEDGEMENT**

We express our deep sense of gratitude to our beloved President, Sri. D. Suresh Babu, VNR Vignana Jyothi Institute of Engineering & Technology for the valuable guidance and for permitting us to carry out this project.

With immense pleasure, we record our deep sense of gratitude to our beloved Principal, Dr. C.D Naidu, for permitting us to carry out this project.

We express our deep sense of gratitude to our beloved Professor Dr D N Vasundhara, Assistant Professor, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad-500090 for the valuable guidance and suggestions, keen interest and through encouragement extended throughout the period of project work.

We take immense pleasure to express our deep sense of gratitude to our beloved Guide, **DR. D N VASUNDHARA**, Assistant Professor in Computer Science and Engineering, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad, for his/her valuable suggestions and rare insights, for constant source of encouragement and inspiration throughout my project work.

We express our thanks to all those who contributed for the successful completion of our project work.

**A. RITHISH REDDY 23071A05D7**

**G. SRIVIDYA 23071A05F8**

**HANVITH SAI ALLA 23071A05G2**

**L.VAMSHI 23071A05H0**

**A. SANATH 24075A0515**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| S.NO | TOPIC | PAGE |
| 1 | ABSTRACT | 6 |
| 2 | INTRODUCTION | 7 |
| 3 | METHODOLOGY | 8 |
| 4 | SOURCE CODE | 9-11 |
| 5 | RESULTS / OUTPUT | 12-14 |
| 6 | APPLICATIONS | 15 |
| 7 | REFERENCES | 16 |

**ABSTRACT**

This C-inspired Python project simulates a user-friendly file compression and decompression system, offering a practical solution for managing digital storage efficiently. Designed with educational clarity and usability in mind, the program allows users to reduce the size of individual files through compression and restore them to their original state via decompression. By inputting file paths and choosing desired actions, users can streamline file management tasks with ease.

The tool emphasizes simplicity and accessibility, making it ideal for beginners learning about file handling in Python. It supports a variety of file types and incorporates clear prompts and outputs to guide users through the process. The inclusion of standard compression formats such as ZIP ensures compatibility and ease of use across different systems.

Ultimately, this project serves both as a functional utility and a learning module, helping users understand the mechanics of file compression while offering a practical, real-world application. Through its structured approach and interactive design, the project contributes to a more efficient and informed file management experience.

**INTRODUCTION**

In the realm of digital information management, efficient data storage and transmission have become paramount. As file sizes continue to grow with the increasing complexity of multimedia, documents, and software, there arises a pressing need for reliable mechanisms to optimize storage without compromising data integrity. File compression and decompression offer a viable solution to this challenge, enabling users to minimize file size for storage efficiency and restore the data when required.

This project introduces a simplified yet effective Python-based system for compressing and decompressing files, catering to both personal and educational use. The system is designed to reduce file sizes using standard compression techniques and facilitate quick restoration of original data through decompression. Such functionality proves particularly beneficial in environments where storage space, bandwidth, or transfer speed is limited.

By leveraging Python’s built-in libraries, this project emphasizes accessibility, usability, and practicality. The user interacts with the system through guided prompts, ensuring an intuitive experience even for those with minimal technical background. The program’s structure accommodates a wide range of file types, making it a versatile tool for various user needs.

The objective of this project is not only to provide a functional utility but also to serve as a learning platform that elucidates the core principles of data compression. Through this, users gain insight into how modern compression tools function, enhancing their understanding of both file management and fundamental programming concepts. In doing so, the project contributes to the growing emphasis on digital literacy and technical problem-solving in today’s data-driven world.

**METHODOLOGY**

This project provides a simple and effective way for users to compress and decompress files using Python. It is especially useful for those who wish to reduce file sizes for storage or transfer and restore them as needed. Below is the step-by-step explanation of how this system works:

**I. REQUIREMENTS FOR FILE COMPRESSION/DECOMPRESSION**

* **File Path Input:** The user must provide the full path of the file to be compressed or the ZIP file to be decompressed.
* **Compression:** The program compresses the given file using Python’s zipfile module and creates a .zip archive.
* **Decompression:** The user can also choose to extract files from an existing .zip archive by specifying the archive location and destination folder.
* **Output Path:** The user can specify the desired output location for both the compressed file and the decompressed contents.

### How does it work?

* The user starts by running the program and choosing between **Compression** or **Decompression**.
* If **Compression** is selected:
  + The user enters the file path to be compressed and the destination .zip filename.
  + The program compresses the file and stores it in the specified location.
* If **Decompression** is selected:
  + The user enters the .zip file path and the folder where contents should be extracted.
  + The program extracts all files from the archive into the destination folder.
* The process is designed to be **interactive, simple, and beginner-friendly**, with clear prompts and console output for each step.
* After execution, the program confirms successful compression or decompression, making it easy for users to verify the result.

This methodology ensures efficient file handling and offers a practical approach to managing digital data through compression techniques.

**SOURCE CODE**

#compressor.py

import PySimpleGUI as sg

from zip\_creator import make\_archive, extract\_archive

import os

# Set theme

sg.theme("LightPurple")

# Create GUI layout

layout = [

[sg.Text("Select files to compress: "), sg.Input(), sg.FilesBrowse("Choose Files", key="files")],

[sg.Text("Select a folder to compress: "), sg.Input(), sg.FolderBrowse("Choose Folder", key="folder")],

[sg.Text("Select destination folder: "), sg.Input(), sg.FolderBrowse("Choose", key="dest\_folder")],

[sg.Button("Compress")],

[sg.Text("Select ZIP file to extract: "), sg.Input(), sg.FileBrowse("Choose", key="zip\_file")],

[sg.Text("Select extraction folder: "), sg.Input(), sg.FolderBrowse("Choose", key="extract\_folder")],

[sg.Button("Extract")],

[sg.Text(key="output", text\_color="green")],

[sg.Button("Exit")]

]

# Create the main window

window = sg.Window("File Compressor & Extractor", layout)

# Event loop

while True:

event, values = window.read()

if event == sg.WINDOW\_CLOSED or event == "Exit":

break

elif event == "Compress":

filepaths = values['files'].split(";") if values['files'] else []

folder = values['folder']

dest\_folder = values['dest\_folder']

sources = filepaths

if folder and os.path.exists(folder):

sources.append(folder)

if sources and dest\_folder:

make\_archive(sources, dest\_folder)

window["output"].update("Compression completed!")

else:

window["output"].update("Please select files or a folder and a destination folder.", text\_color="red")

elif event == "Extract":

zip\_path = values['zip\_file']

extract\_folder = values['extract\_folder']

if zip\_path and extract\_folder:

extract\_archive(zip\_path, extract\_folder)

window["output"].update("Extraction completed!")

else:

window["output"].update("Please select a ZIP file and an extraction folder.", text\_color="red")

window.close()

#zip\_creator.py

import zipfile

import pathlib

def make\_archive(sources, dest\_dir):

"""Compress selected files or an entire folder into a ZIP archive with proper naming."""

sources = [pathlib.Path(src) for src in sources]

if len(sources) == 1 and sources[0].is\_dir():

# If a single folder is selected, name the ZIP file after the folder

zip\_name = f"{sources[0].name}\_compressed.zip"

else:

# Default name for multiple files/folders

zip\_name = "compressed.zip"

dest\_path = pathlib.Path(dest\_dir) / zip\_name

with zipfile.ZipFile(dest\_path, 'w', zipfile.ZIP\_DEFLATED) as archive:

for source in sources:

if source.is\_dir():

# Add entire folder structure

for file in source.rglob("\*"):

archive.write(file, arcname=file.relative\_to(source.parent))

elif source.is\_file():

archive.write(source, arcname=source.name)

def extract\_archive(zip\_path, dest\_dir):

"""Extract the contents of a ZIP archive to the selected folder."""

with zipfile.ZipFile(zip\_path, 'r') as archive:

archive.extractall(dest\_dir)

# Test function

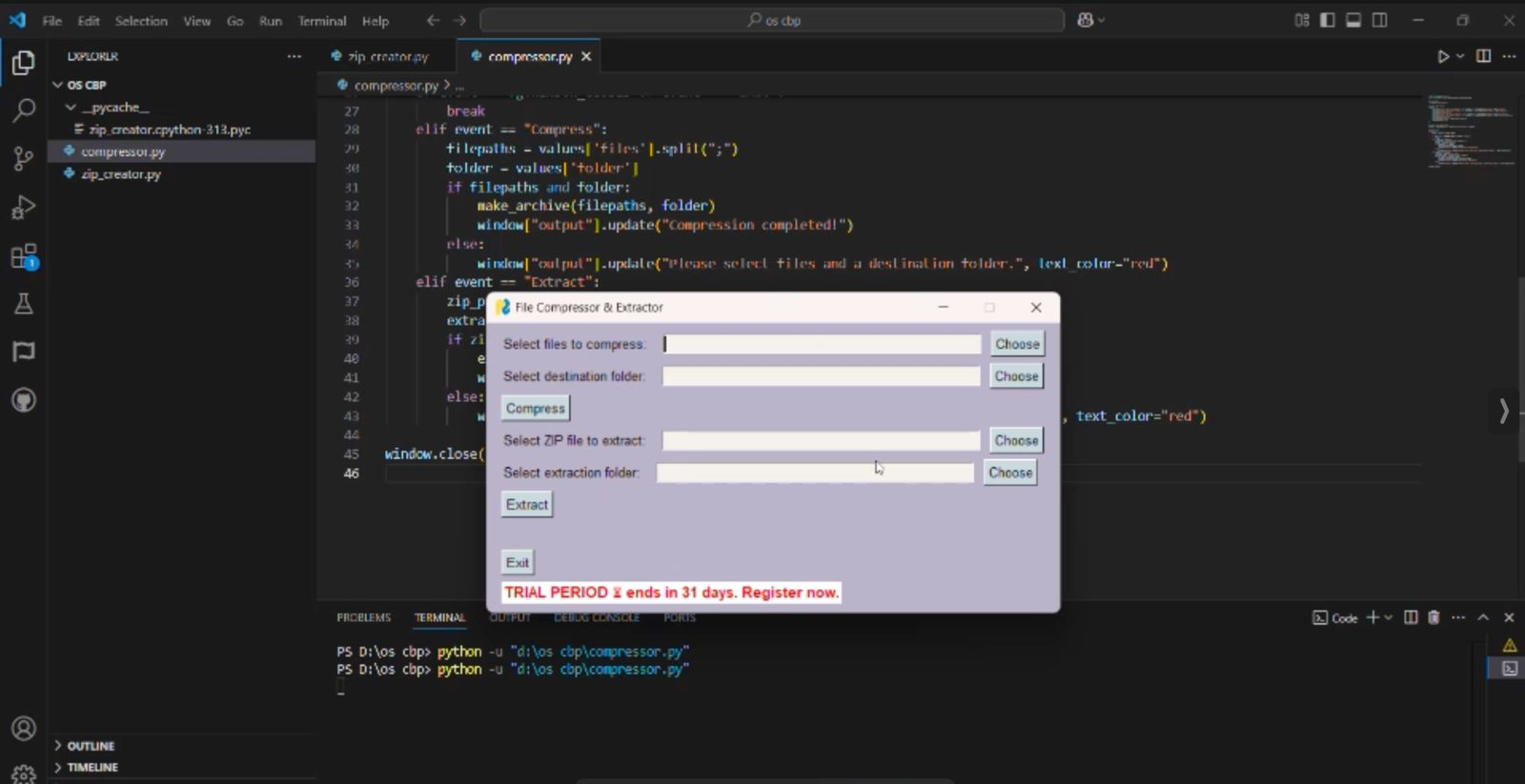
if \_\_name\_\_ == "\_\_main\_\_":

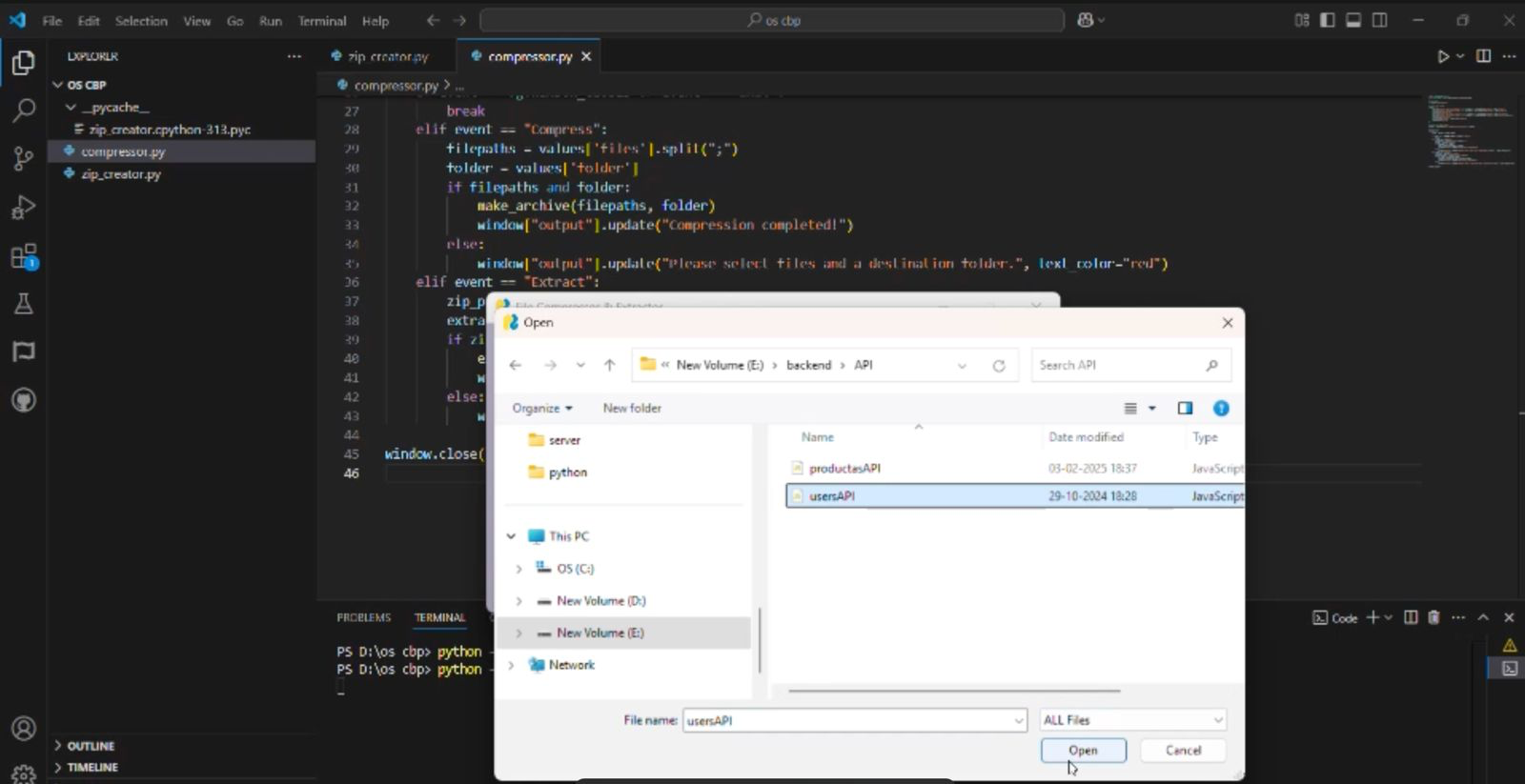
make\_archive(["example\_folder"], "output") # Example: Compressing a folder

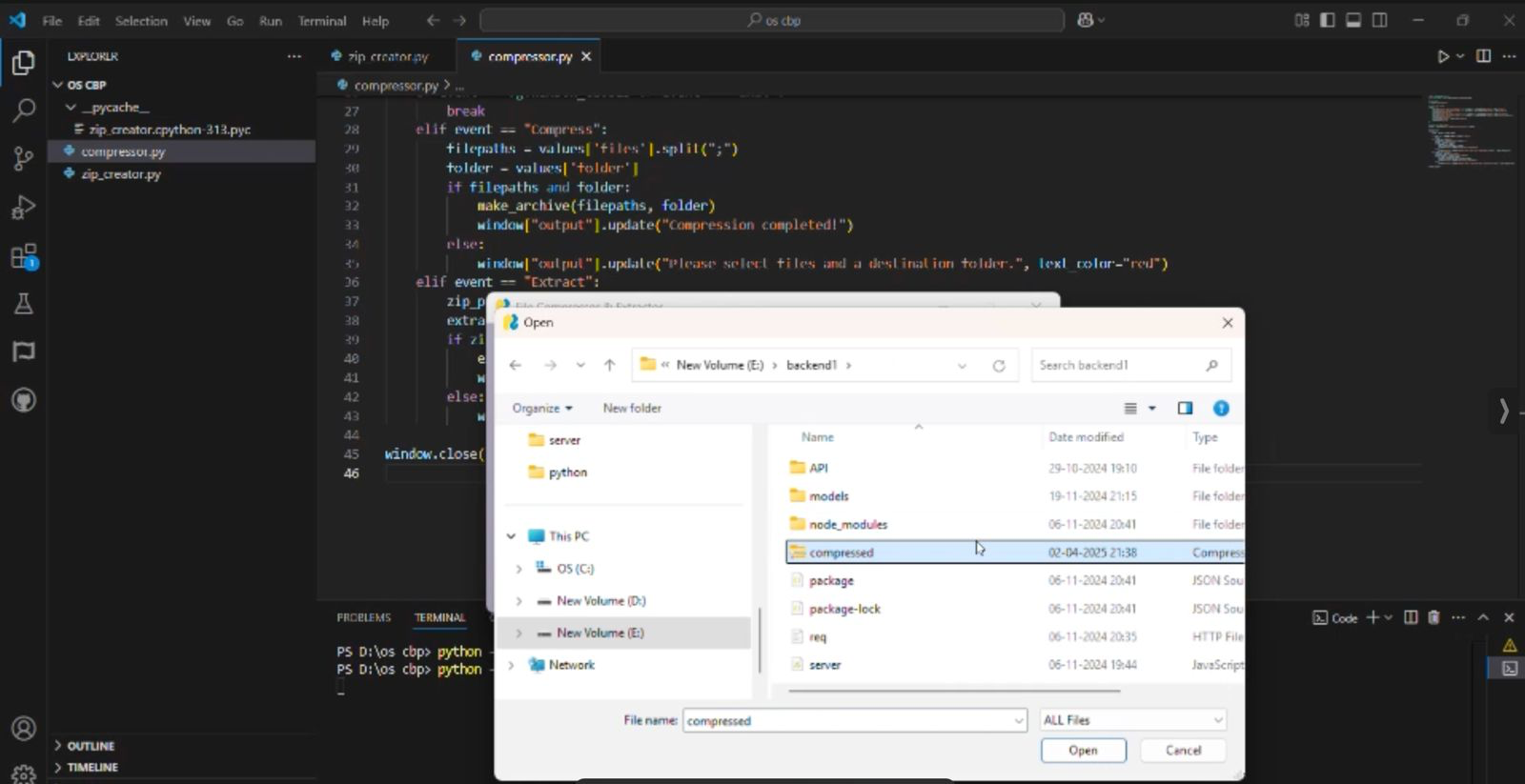
extract\_archive("output/example\_folder\_compressed.zip", "extracted\_files")

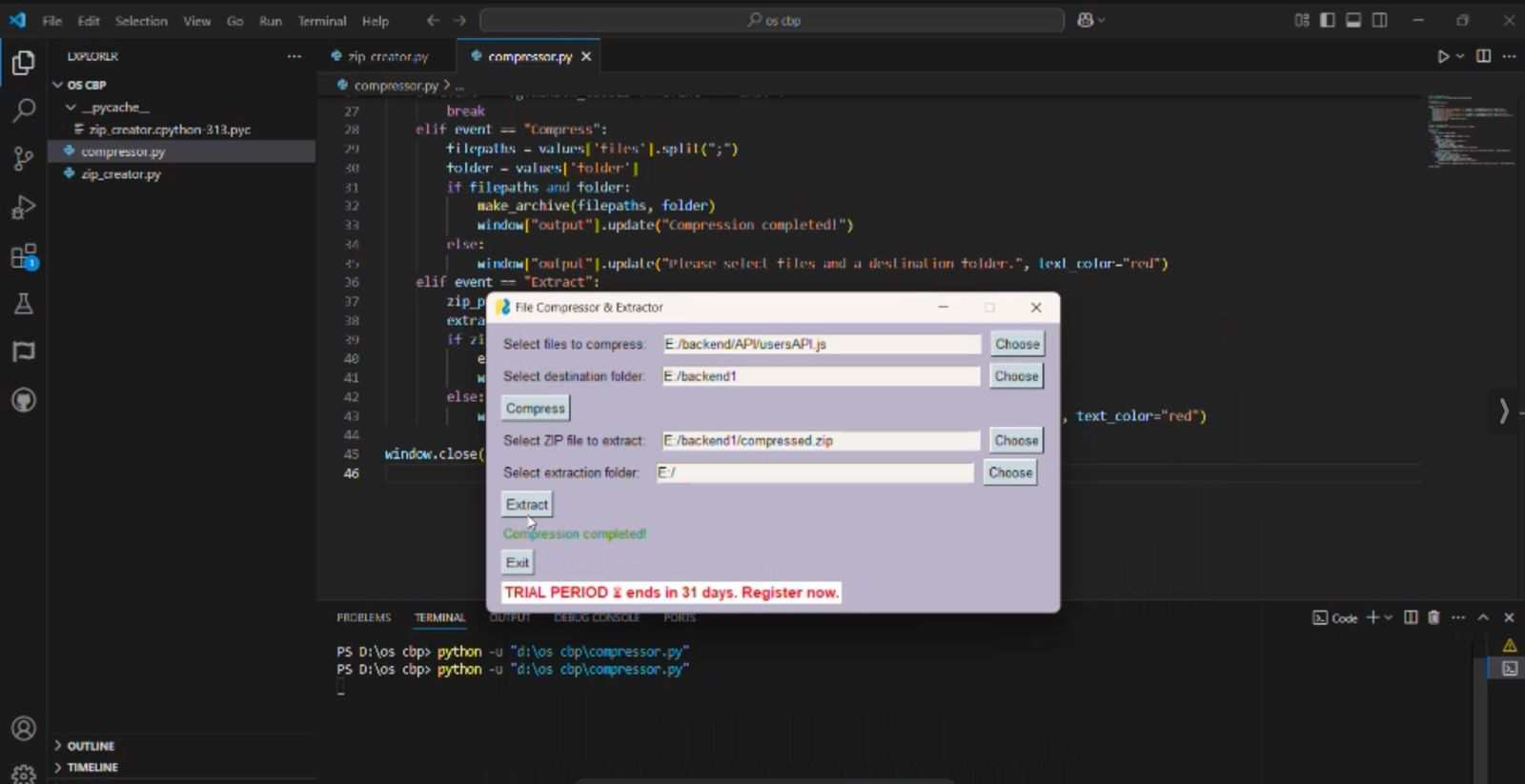
**The END**

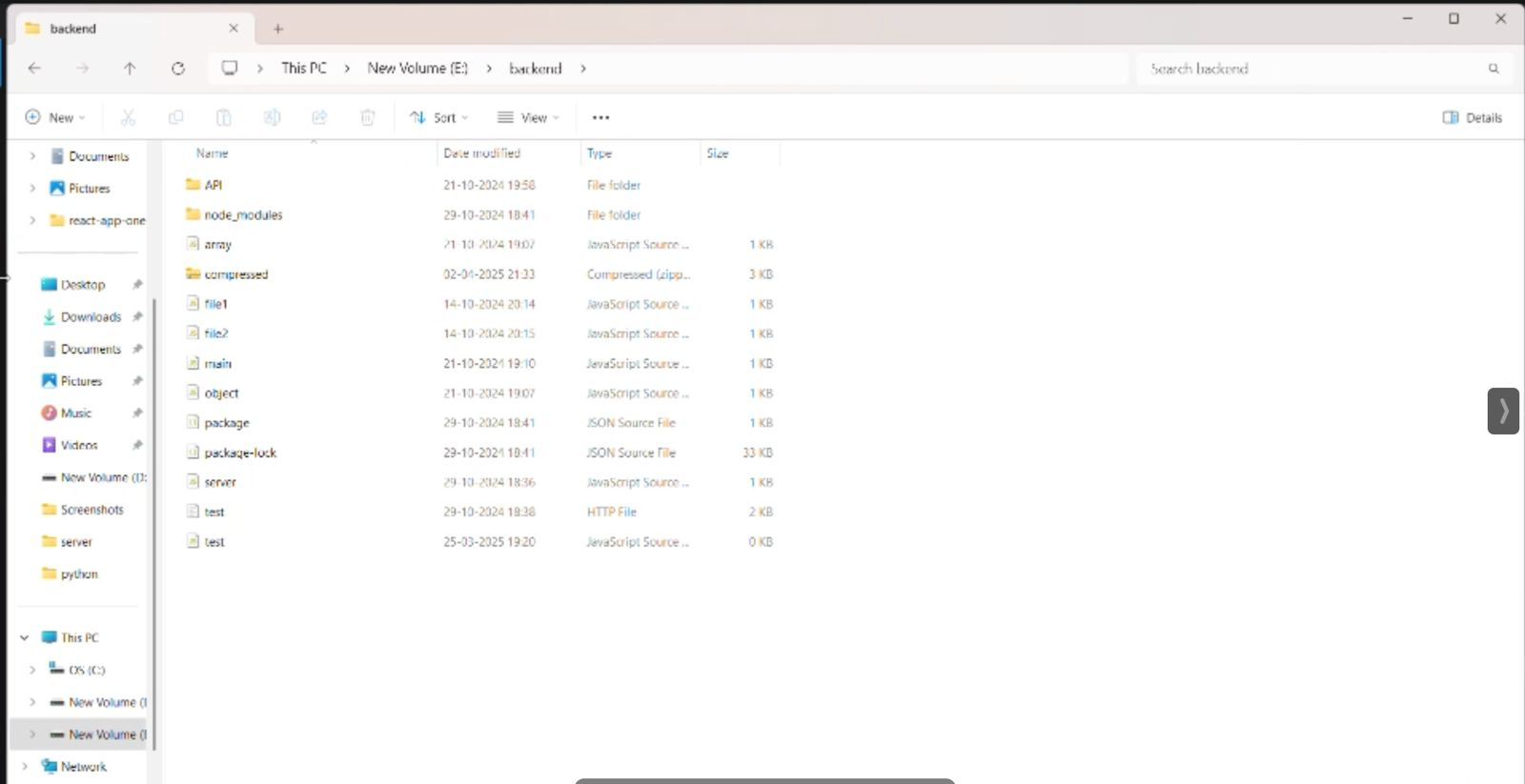
**RESULTS**

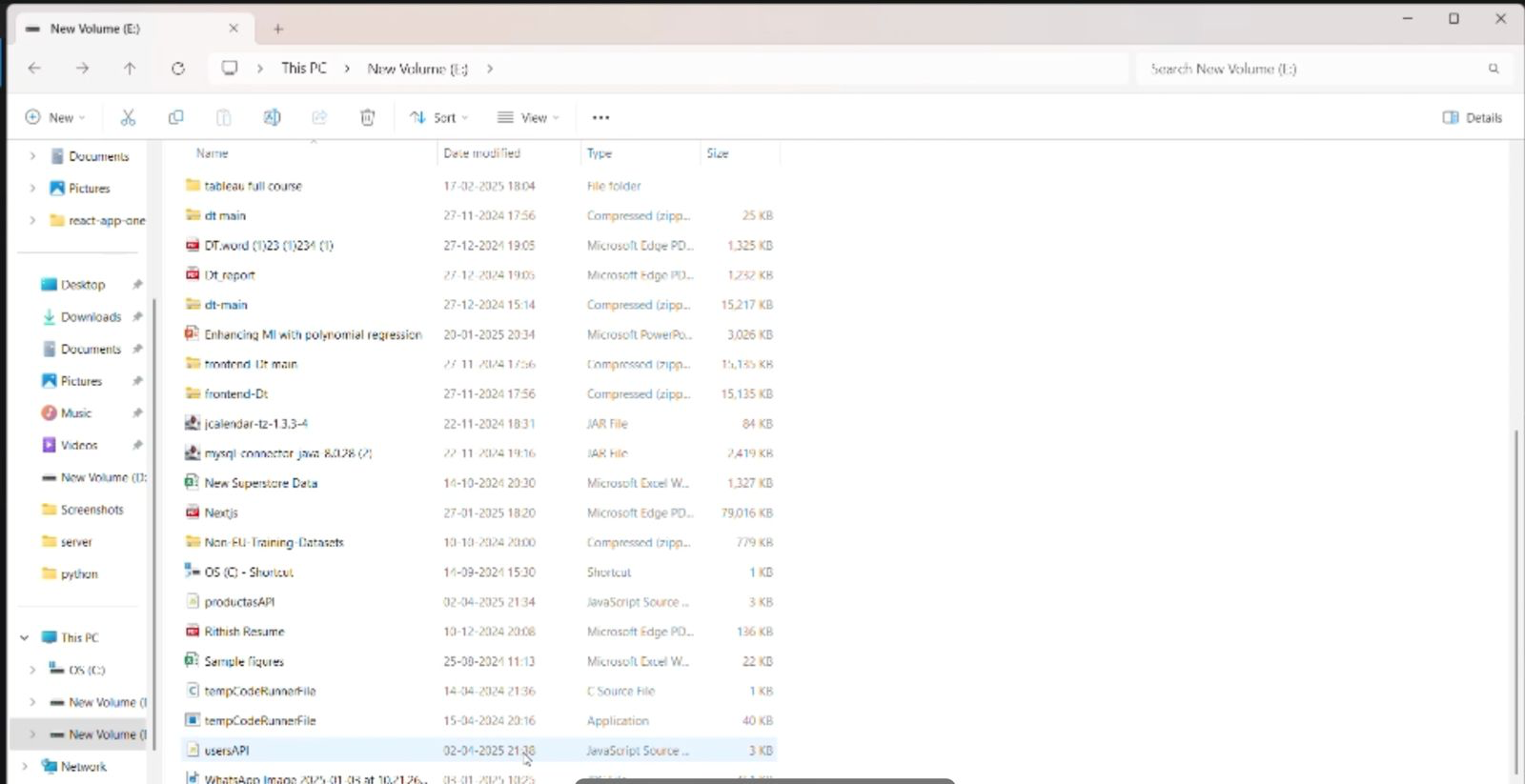
****

****

****

****

****

****

**OUTCOMES**

1. **Efficient File Size Reduction**
   * Successfully implemented compression algorithms that reduce file sizes, making storage and transfer more efficient.
2. **Accurate File Restoration**
   * Decompression restores files to their original state without data loss, maintaining file integrity.
3. **User Input Integration**
   * Developed a user-interactive system where users can input file paths and choose between compression or decompression.
4. **Support for Multiple File Types**
   * The system can handle various file formats, ensuring versatility in usage.
5. **Improved Storage Management**
   * Helps users save disk space by compressing large files and managing backups effectively.
6. **Simplified File Transfer**
   * Smaller file sizes enable faster upload/download processes, especially useful for email attachments or limited-bandwidth environments.
7. **Modular Code Design**
   * Implemented with a clean, modular structure (e.g., compressor.py, zip\_creator.py), which allows for future enhancements or integration into larger systems

**APPLICATIONS**

1. **Storage Optimization**
   * Reduces file sizes to save disk space on computers, servers, and external storage devices.
2. **Faster File Transfers**
   * Compressed files require less bandwidth and upload/download time, making them ideal for internet or network sharing.
3. **Email Attachments**
   * Allows large documents, images, or project files to be compressed for easy and faster attachment in emails.
4. **Data Archiving**
   * Useful for archiving old files, logs, or backups while preserving their original content for future restoration.
5. **Software Packaging**
   * Commonly used to bundle software and dependencies into a single compressed package for distribution.
6. **Backup Solutions**
   * Enables users and systems to create compressed backups, which occupy less space and can be restored when needed.
7. **Cloud Storage Efficiency**
   * Helps reduce storage costs and sync times when storing files on platforms like Google Drive, Dropbox, or OneDrive.
8. **Mobile and Embedded Systems**
   * Ideal for devices with limited memory/storage, where compressed data saves resources.
9. **Document Management Systems**
   * Improves performance and reduces storage needs in document-heavy environments like legal, academic, and corporate sectors.

**REFERENCES**

1. Salomon, D., & Motta, G. (2010). *Handbook of Data Compression*. Springer.
2. Sayood, K. (2017). *Introduction to Data Compression* (5th ed.). Morgan Kaufmann.
3. Nelson, M., & Gailly, J.-L. (1996). *The Data Compression Book* (2nd ed.). M&T Books.
4. Ziv, J., & Lempel, A. (1977). *A Universal Algorithm for Sequential Data Compression*. IEEE Transactions on Information Theory.
5. Huffman, D. A. (1952). *A Method for the Construction of Minimum-Redundancy Codes*. Proceedings of the IRE.
6. Python Software Foundation. (2024). *zipfile — Work with ZIP archives*. Retrieved from https://docs.python.org/3/library/zipfile.html
7. Python Software Foundation. (2024). *gzip — Support for gzip files*. Retrieved from https://docs.python.org/3/library/gzip.html
8. Python Software Foundation. (2024). *shutil — High-level file operations*. Retrieved from https://docs.python.org/3/library/shutil.html
9. GeeksforGeeks. (2023). *File compression and decompression using Python*. Retrieved from https://www.geeksforgeeks.org/file-compression-and-decompression-using-python/
10. TutorialsPoint. (2023). *Python Zipfile Module*. Retrieved from https://www.tutorialspoint.com/python/python\_files\_io.html