



Graphic Era
deemed to be **University**
DEHRADUN


PROJECT AND TEAM INFORMATION

Project Title

(Try to choose a catchy title. Max 20 words).

User-Driven File Compression System

Student/Team Information

Team Name: Team # (Mentor needs to assign)	Minimizers
Team member 1 (Team Lead) (Last Name, name: student ID: email, picture):	Bhandari,Adarsh-23021079 adarshsinghbhandari@gmail.com 

Team member 2

(Last Name, name: student ID: email, picture):

Mehta,Kaavya-23021324

Kaavyavinaymehta22@gmail.com



Team member 3

(Last Name, name: student ID: email, picture):

Dhondiyal,Chitransh- 230213734

chitranshdhoundiyal00@gmail.com



Team member 4 (Last Name, name: student ID: email, picture):

Sirola, Himanshu- 23021326 Email-
sirolahimanshu440@gmail.com



PROJECT PROGRESS DESCRIPTION (35 pts)

Project Abstract

(Brief restatement of your project's main goal. Max 300 words).

1. In today's data-centric era, efficient storage and faster transmission are crucial. Large files occupy excessive space and slow down communication systems. The User-Driven File Compression System is a Python-based project designed to address this challenge by implementing multiple compression techniques—Huffman Coding, LZW, and Run-Length Encoding (RLE) based on key Design and Analysis of Algorithms (DAA) strategies like Greedy, Divide-and-Conquer, and Dynamic Programming. It enables compression of both text and binary files, allowing algorithm selection based on data characteristics for optimal efficiency.
2. The system features an intuitive graphical interface that showcases file sizes before and after compression, providing users with a visual understanding of each algorithm's performance. This unique blend of utility and education distinguishes our tool from conventional compressors by offering transparency, adaptability, and insight. It promotes deeper algorithmic understanding while delivering a practical solution to modern storage and transmission needs.

Updated Project Approach and Architecture

(Describe your current approach, including system design, communication protocols, libraries used, etc. Max 300 words).

1. Modular System Architecture with GUI: The User-Driven File Compression System implements three algorithms—Run-Length Encoding (RLE), Huffman Coding, and Lempel-Ziv-Welch (LZW)—in a modular backend. Each algorithm is encapsulated in separate Python functions for clarity and reusability. The GUI, built using Tkinter, allows users to select files, choose an algorithm, and start compression. File paths are handled using filedialog, and user choices are passed to the backend. Output is saved to a user-defined location, with status alerts shown via messagebox.

2. Technologies Used and Future Plans: Tkinter is used for the GUI, while os and filedialog support file operations. Custom Python modules handle the compression logic. The modular design ensures easy maintenance and expansion. Future enhancements include adding decompression support to the GUI, showing compression ratios and execution time, previewing files before/after compression, and developing a web-based interface using Flask for broader access.

Tasks Completed

(Describe the main tasks that have been assigned and already completed. Max 250 words).

Task Completed	Team Member
1. Huffman Coding Algorithm – Structure and Implementation.	Chitransh Dhondiyal
2. Developed the RLE algorithm and tested it successfully on sample text.	Kaavya Mehta
3. Implemented the LZW dictionary algorithm logic and ensured it performs correctly on test data.	Himanshu Sirola
4. Modular Integration of All Algorithms (Initial Phase).	Adarsh Bhandari & Himanshu Sirola
5. Research and Blueprint Design for binary Compression.	Kaavya Mehta & Chitransh Dhondiyal
6. Conducted preliminary end-to-end testing of the integrated Huffman, LZW, and RLE algorithms on various text files to ensure correct functionality and data flow within the application	Adarsh Bhandari
	Entire team

Challenges/Roadblocks

(Describe the challenges that you have faced or are facing so far and how you plan to solve them. Max 300 words).

1) **Algorithm Implementation Differences**

Implementing three different compression algorithms (RLE, Huffman, and LZW) posed difficulties due to their distinct logic and output formats. Ensuring each algorithm functioned independently and correctly compressed files without errors took time and iterative debugging.

2) **GUI Functionality**

Building an intuitive GUI with Tkinter presented issues in capturing user selections and file paths accurately. Initially, file paths weren't passed correctly to backend functions. These issues were resolved by restructuring the control flow between GUI components and backend logic.

3) **File Format and Error Handling**

We faced several file path errors (especially on Windows) and Unicode encoding issues. These were resolved by using raw strings (r"path") and specifying encoding formats explicitly during file operations.

4) **User Feedback and Interface Limitations**

At this stage, the GUI offers only basic compression functionality. Providing feedback messages and maintaining responsiveness during long operations posed challenges, which were addressed using message boxes and proper status updates.

Tasks Pending

(Describe the main tasks that you still need to complete. Max 250 words).

Task Pending	Team Member (to complete the task)
1. Basic GUI development completed; requires further refinement to enhance usability and functionality.	Adarsh Bhandari
2. Integration of all algorithms -Efficiently and dynamically choosing the correct algo for compression and integrating all of them into a single program.	Kaavya Mehta
3. Binary compression using RLE algorithm.	Kaavya Mehta
4. Binary compression using Huffman encoding algorithm.	Chitransh Dhondiyal
5. Binary compression using LZW algorithm.	Himanshu Sirola
6. Testing and Debugging - Conduct thorough testing of the GUI and image compression features to identify and resolve any bugs or usability issues	Adarsh Bhandari & chitransh Dhondiyal
7. Development and integration of binary file compression functionality.	Himanshu Sirola

Project Outcome/Deliverables

(Describe what are the key outcomes / deliverables of the project. Max 200 words).

1. Our approach offers a more **transparent, efficient, and user-friendly** file compression solution for various data types, including binary, text. Unlike conventional compression applications, our system empowers users to directly compare the performance of different compression algorithms, enabling them to observe firsthand how algorithmic choices impact file size reduction.
2. From a technical standpoint, the project delivers a **functional algorithm selection mechanism**. This mechanism, driven by user-defined compression levels ('Best,' 'Normal,' 'Optimal'), effectively maps these levels to the appropriate compression algorithm. The implementation showcases conditional logic and control flow techniques to dynamically choose and execute the selected algorithm, demonstrating a key aspect of system design. We are yet to implement binary compression algorithm, but we have planned everything beforehand, and it will be executed in a timely manner.

Progress Overview

(Summarize how much of the project is done, what's behind schedule, what's ahead of schedule. Max 200 words.)

1. Our team has **successfully structured and implemented the core logic** for the Huffman Coding algorithm, developed and tested the Run-Length Encoding (RLE) algorithm on sample text data, and implemented the Lempel-Ziv-Welch (LZW) dictionary algorithm, ensuring its correct performance on test data. Furthermore, we have completed the **initial modular integration of all three algorithms**, laying a solid foundation for the system's core functionality. We've also **drafted the GUI flow** and identified key components for file upload and result display and conducted preliminary research and blueprint design for the future implementation of binary compression.
2. While the GUI and binary compression features are planned for subsequent phases, our phased development approach ensures that the project remains manageable, and **our current status is positive** and on track with the planned milestones.

Codebase Information

(Repository link, branch, and information about important commits.)

- **Repository Link:** <https://github.com/Adarsh-bhandari1/User-Driven-File-Compression-System.git>
- **Primary Branch:** main

Important Commits:

- **Initial commit** – Set up project structure and added base algorithm templates.
- **Added RLE, Huffman, and LZW compression logic** – Implemented core logic for all three compression algorithms.
- **Basic GUI implemented** – Integrated Tkinter GUI for file selection and compression initiation.
- **Modularized code and improved structure** – Separated compression logic into modules and improved code maintainability.
- **Connected GUI to backend algorithms** – Enabled functional compression via GUI with file input/output handling

Testing and Validation Status

(Provide information about any tests conducted)

Test Type	Status(Pass/Fail)	Notes
RLE Compression	Pass	Successfully compressed input text files using Run-Length Encoding.
Huffman Compression	Pass	Generated compressed output using frequency-based encoding.
LZW Compression	Pass	Properly converted input text into integer codes using LZW.

Deliverables Progress

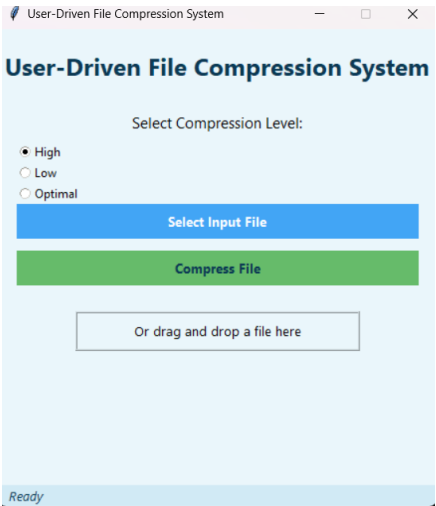
(Summarize the current status of all key project deliverables mentioned earlier. Indicate whether each deliverable is completed, in progress, or pending.)

1. Python-based compression-only application:

The core compression algorithms—Huffman, LZW, and RLE—have been implemented and subjected to testing, yielding positive results on various text files. Research and development for binary compression integration are ongoing, with implementation scheduled for a subsequent phase.








2. Graphical User Interface (GUI) Development:

Recognizing that a user-centric design is crucial for transparency and ease of use, we are actively developing the GUI. We have identified a viable solution for efficient algorithm integration within the interface. The implementation of the GUI is currently underway.



3.Compressed Files / Output Generation:

Completed – Output files for all three compression algorithms (Huffman .huff, LZW .lzw, and RLE .rle) have been successfully generated from the input file, confirming the correct implementation and execution of the compression module.

 compressed.huff		20-05-2025 23:07	HUFF File	1,136 KB
 compressed.lzw		20-05-2025 23:07	LZW File	163 KB
 compressed		20-05-2025 23:07	RLE File	586 KB
 input		17-05-2025 10:52	Text Document	3,047 KB

