Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

ISSN : 2087 - 3891 (PRINT)
ISSN : 2597 - 8918 (DIGITAL)

Jurnal Ilmiah Teknologi - Informasi & Sains
TEKNOIS
Universitas Binaniaga Indonesia

Change, Continuous Quality Improvement, De'bottlenecking, Enhancement... Do It!

Article DOI: 10.36350/jbs.v14i1.232
Received: December; Accepted: December; Published: January

Penerapan Algoritma Lempel Ziv Welch (LZW) Untuk Kompresi Data

Ichwan Arizki[1], Anggra Triawan[2*], Farhan Zayid[3]
[1,2,3]Teknik Informatika/Universitas Binaniaga Indonesia
Email: anggra@unbin.ac.id
) Corresponding Author

ABSTRACT

File compression is an important aspect in the development of information technology. The demand to provide information in a short time with a large number and size of files makes compression techniques very important. One of the problems faced when the number of files is large is the fulfillment of storage. The process of storing, sending data and bandwidth requirements is very important because most of the latest information comes from cyberspace, namely the internet. Given the above problems, the Lempel ziv Welch Algorithm (LZW) is a method that can overcome the series of problems above. The Lempel Ziv Welch (LZW) algorithm is a string compression algorithm created by three friends named Lempel, Ziv and Welch in 1977. LZW is an algorithm with a lossless compression method, meaning that it reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. The result of this final project is an analytical study of how the Lempel Ziv Welch Algorithm (LZW) obtains a feasibility value of 90%.

Keywords*: Compression, decompression, lossless compression, lzw algorithm*

ABSTRAK

Kompresi file merupakan salah satu aspek penting dalam perkembangan teknologi informasi. Tuntutan penyediaan informasi dalam waktu singkat dengan jumlah dan ukuran file yang banyak menjadikan teknik kompresi menjadi sangat penting. Salah satu masalah Yang dihadapi Ketika file berjumlah banyak maka akan terjadi pemenuhan pada penyimpanan. Proses penyimpanan, pengiriman data maupun kebutuhan bandwidth sangat penting karena informasi terkini sebagian besar berasal dari dunia maya yaitu internet. Dengan permasalahan diatas Algoritma Lempel ziv welch (LZW) merupakan salah satu metode yang dapat mengatasi rangkaian masalah diatas. Algoritma Lempel Ziv Welch (LZW) merupakan salah salu algoritma string compression yang diciptakan oleh tiga sekawan bernama Lempel, Ziv dan Welch pada tahun 1977. LZW termasuk algoritma dengen metode lossless compression, artinya mengurangi bit dengan mengidentifikasi dan menghilangkan redundansi statistik. Tidak ada informasi yang hilang dalam kompresi lossless. Hasil tugas akhir ini adalah studi analisis bagaimana Algoritma Lempel Ziv Welch (LZW) mendapatkan nilai kelayakan sebesar 90%.

Keywords*: Kompresi, dekompresi, lossless compression, algoritma lzw.*

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

A. PENDAHULUAN

Perkembangan teknologi informasi yang pesat sangat mempengaruhi dalam penyajian sebuah data atau informasi. Pada saat ini penyajian sebuah data atau informasi tidak hanya dalam bentuk teks, tetapi dapat berupa gambar, audio dan video. Keempat elemen data tersebut merupakan bagian dari objek digital. Pada umumnya representasi objek dalam bentuk digital membutuhkan memori. Semakin besar ukuran objek, maka semakin besar pula memori yang dibutuhkan Di samping itu komunikasi data sangat sering dilakukan. Komunikasi data ini berhubungan erat dengan pengiriman file menggunakan sistem transmisi elektronik dari satu terminal komputer ke terminal komputer yang lain. Besarnya ukuran file terkadang menjadi kendala dalam proses pengirimannya. File dengan ukuran besar akan memakan waktu pengiriman yang lebih lama dibandingkan dengan file yang memiliki ukuran lebih kecil, terkadang ada resiko tidak dapat tertampung file pada media penyimpanan, sehingga akan memperkecil kapasitas kosong dalam memori media penyimpanan. Oleh karena itu, manusia selalu berusaha untuk menemukan suatu cara alternatif untuk menangani permasalahan tersebut, salah satunya dengan cara kompresi. Secara spesifik, kompresi file bertujuan untuk mengubah data menjadi sekumpulan kode untuk menghemat tempat penyimpanan dengan atau tanpa mengurangi kualitasnya serta mempercepat waktu transmisi data. Dengan memanfaatkan teknik kompresi ini, maka proses pengiriman file akan menjadi lebih maksimal dan lebih cepat.

The Lempel-Ziv (LZ) metode kompresi adalah salah satu algoritma paling populer untuk penyimpanan lossless. mengempis adalah variasi LZ yang dioptimalkan untuk kecepatan dekompresi dan rasio kompresi, sehingga kompresi ini bisa lambat. Deflate digunakan dalam PkZip, gzip dan PNG. LZW (Lempel-Ziv-Welch) digunakan dalam gambar GIF. Juga patut diperhatikan adalah LZR (LZ-Renau) metode, yang melayani sebagai dasar dari metode Zip. metode LZ memanfaatkan model kompresi berbasis tabel di mana entri tabel diganti untuk string data yang diulang. Untuk metode yang paling LZ, tabel ini dihasilkan secara dinamis dari data sebelumnya dalam input. Tabel sendiri sering Huffman dikodekan (misalnya Shri, LZX). berdasarkan skema coding LZ arus yang baik adalah melakukan LZX, digunakan dalam Microsoft CAB format. Losseless algoritma kompresi memanfaatkan redundansi biasanya statistik sedemikian rupa untuk mewakili pengirim data lebih singkat tanpa kesalahan. kompresi Lossless dimungkinkan karena sebagian besar dunia nyata telah redundansi data statistik. Sebagai contoh, dalam teks bahasa Inggris, 'e' huruf jauh lebih umum daripada huruf 'z', dan probabilitas bahwa 'q' huruf akan diikuti oleh huruf 'z' sangat kecil. Kompresi jenis lain, disebut kompresi lossy data atau persepsi coding, adalah mungkin jika beberapa kehilangan kesetiaan diterima. Umumnya, sebuah kompresi data lossy akan dipandu oleh penelitian tentang bagaimana orang melihat data tersebut. Sebagai contoh, mata manusia lebih sensitif terhadap variasi halus dalam terang daripada variasi warna. JPEG kompresi gambar yang bekerja di sebagian oleh "pembulatan" beberapa informasi penting ini-kurang. Lossy kompresi data menyediakan cara untuk mendapatkan kesetiaan terbaik untuk jumlah yang diberikan kompresi.

B. METODE

1. Algoritma LZW

(Dzulhaq & Andayani, 2014, p. dua) menguraikan perihal prosedur pemecahan Lempel-ziv-Welch (LZW) adalah prosedur pemecahan kompresi lossless universal yg diciptakan Abraham Lempel, Jacob Ziv, serta Terry Welch. dari (Bakara, 2017, p. 42) algoritma LZW menggunakan teknik adaptif dan berbasis “kamus”. Pendahuluan LZW merupakan LZ77 dan LZ78 yang berkembang sang Jacob Ziv dan Abraham Lempel pada 1997 serta 1978. Terry Welch berbagi teknik tersebut pada 1984. prosedur pemecahan ini melakukan kompresi menggunakan memakai dictionary, dimana fragmen-fragmen teks diganti menggunakan indeks yang diperoleh oleh sebuah “kamus”. Prinsip homogen juga digunakan 22 pada kode Braille, dimana kode-kode khusus dipergunakan buat merepresentasikan istilah-kata yg terdapat. Pendekatan ini bersifat adaptif dan efektif karena banyak karakter dapat dikodekan dengan mengacu di string yg sudah timbul sebelumnya dalam teks. Prinsip kompresi tercapai Bila surat keterangan dalam bentuk pointer dapat disimpan pada jumlah bit yang lebih sedikit

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

dibandingkan string aslinya. Urutan langkah prosedur pemecahan kompresi LZW artinya sebagai berikut:

a. Dictionary diinisialisasikan pada jumlah bit yang lebih sedikit dibandingkan string aslinya.
b. P artinya karakter pertama dalam stream karakter.
c. Q adalah karakter berikutnya dalam stream karakter.
d. Apakah string (P + Q) ada pada dictionary ?
   1) Bila "ya" maka P = P + Q (adonan P serta Q menjadi string baru)
   2) Jika "tidak" maka: output sebuah kode buat menggantikan string P, lalu masukkan string (P + Q) ke pada dictionary serta berikan angka/kode berikutnya yg belum diganti dalam dictionary buat string tersebut. Dengan P = Q. e)
e. Apakah masih terdapat karakter berikutnya pada stream karakter?
   Bila "ya" maka pulang ke langka dua, dan Jika "tidak" maka output kode yg menggantikan string P, lalu terminasi proses (stop).

2. Kompresi Data

(Aji, Darwiyanto, & Septiana, 2016, p. 5198) Kompresi data ialah ilmu atau seni merepresentasikan gosip dalam bentuk yg lebih compact. David Salomon mengatakan bahwa data kompresi merupakan proses mengkonversikan sebuah input data stream (stream asal, atau 20 data mentah asli) menjadi data stream lainnya (bit stream hasil, atau stream yang telah terkompresi) yang ukuran lebih kecil. Pemampatan merupakan salah satu berasal bidang teori info yg bertujuan buat menghilangkan redundansi berasal sumber. Pemampatan berguna pada membantu mengurangi konsumsi asal daya yang mahal, mirip ruang hard disk atau perpindahan data melalui internet. Tujuan asal kompresi data artinya buat merepresentasikan suatu data digital menggunakan sesedikit mungkin bit, tetapi permanen mempertahankan kebutuhan minimum buat menghasilkan pulang data aslinya. Data digital ini bisa berupa text, gambar, bunyi, dan kombinasi asal ketiganya, mirip video.

buat membuat suatu data menjadi lebih mungil ukurannya daripada data orisinil, diharapkan tahapan-tahapan (prosedur pemecahan) buat memasak data tadi. pada prosedur pemecahan kompresi data, tidak terdapat prosedur pemecahan yang cocok untuk semua jenis data. Hal ini ditimbulkan sebab data yang akan dimampatkan harus dianalisis terlebih dahulu, dan berharap menemukan pola eksklusif yang bisa digunakan buat memperoleh data pada berukuran yang lebih kecil. sebab itu, timbul banyak prosedur pemecahan -prosedur pemecahan kompresi data.

Kompresi Lossy
Kompresi data yang menghasilkan file data akibat kompresi yang tidak bisa dikembalikan sebagai arsip data sebelum dikompresi secara utuh. saat data akibat kompresi pada-decode balik, data akibat decoding 21 tersebut tidak dapat dikembalikan menjadi sama menggunakan data asli tetapi terdapat bagian data yg hilang. b) Kompresi Lossless.

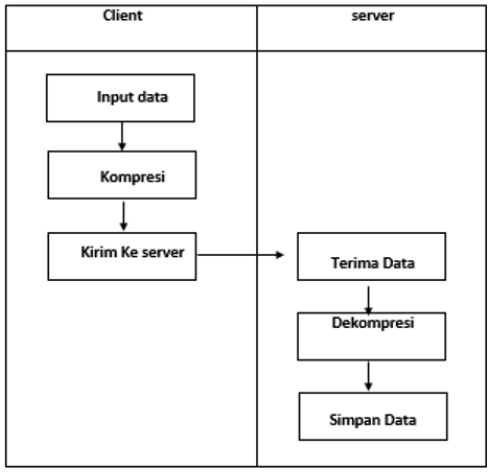
Kompresi data yang membentuk file data yang akan terjadi kompresi yang dapat dikembalikan menjadi file data asli sebelum dikompresi secara utuh tanpa perubahan apapun. Kompresi jenis ini ideal buat kompresi teks. prosedur pemecahan yang termasuk dalam metode kompresi lossless antara lain adalah dictionary coding serta huffman coding. berdasarkan Bakara (2017: p. 42) Rasio kompresi adalah berukuran persentase file yang sudah berhasil dimampatkan. Secara matematis rasio pemampatan dituliskan sebagai berikut. Rasio kompresi =100% - ((berukuran akibat kompresi)/(ukuran semula)) x 100% Misalkan rasio kompresi merupakan 25%, artinya 25% berasal file semula sudah berhasil dimampatkan.

C. HASIL DAN PEMBAHASAN

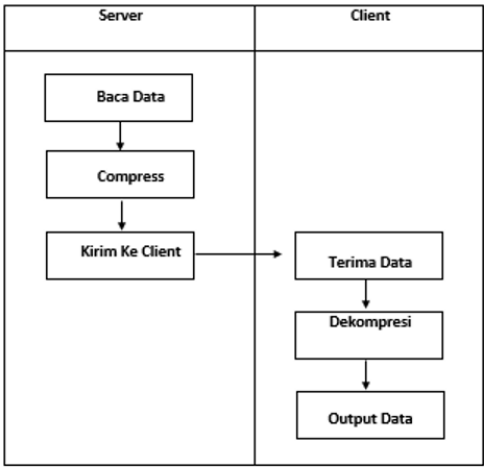
Untuk mengefektifkan kompresi data dan mengoptimalkan pada penyimpanan data, peneliti mencoba membuat dan menerapan Algoritma Lempel Ziv Welch (Lzw).

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

Proses kompresi dan dekompresi dengan algoritma LZW (Lempel Ziv Welch), dapat dijelaskan sebagai berikut, proses kompresi data secara sederhana mengganti string dari karakter dengan kode tunggal dan tidak melakukan analisa apapun pada teks yang masuk. Tetapi hanya menambahkan setiap string dari karakter baru yang ditemuinya ke tabel string (dictionary). cara kerja algoritma lemper ziv welch adalah mengecilkan ukuran file dengan cara kompresi dimana nantinya ukuran tersebut menjadi lebih kecil dari ukuran asli dan data tersebut akan di ambil atau di akses sesuai dengan keperluan aplikasi, sehingga dengan adanya LZW ini proses kompresi mejadi lebih efektif dan penyimpanan data lebih optimal.
Kompresi terjadi ketika ditemui string berulang, algoritma ini menghasilkan kode tunggal dan menggantikan string-string berulang yang ditemui selama proses kompresi. Panjang kode yang dihasilkan algoritma LZW dapat berubahubah, tetapi jumlah bit yang dimiliki kode tersebut harus lebih banyak dari pada sebuah karakter tunggal. Proses dekompresi pada LZW dilakukan dengan prinsip yang sama seperti proses kompresi.



Gambar 1. Alur Client ke Server



Gambar 2. Alur Server ke Client

1. Penerapan Algoritma dalam Source Code
   Pengkodean merupakan tahap di mana LZW diterapkan untuk sistem yang dikembangkan. Pengkodean yang dilakukan pada bahasa pemrograman yang dipakai yaitu adalah bahasa Python dan Html dan sedikit javascript untuk validasi. Pengkodean yang dilakukan ada 2 jenis yaitu pengkodean pada proses kompresi dan pengkodean pada proses dekompresi.
   a. Source Code Proses Compress
      Pada gambar 3 merupakan stuktur Coding ini berisi tentang proses kompresi data menggunakan algorima lzw dan dibantu oleh Bahasa program python dan juga html. Pada bagian ini terjadi sebuah proses kompresi data yang akan mendapatkan hasil .lzw sebagai output nya dan akan terjadi enkripsi data.

   b. Source Code Decompresed
      Pada gambar 4 merupakan stuktur Coding ini berisi tentang proses dekompresi data atau mengembalikan data ke ukuran semula serta mengembalikan semua informasi didalam nya menggunakan algorima lzw dan dibantu oleh Bahasa program python dan juga html. Pada pengkodean ini diterapkan sebuah konsep yang dimana cara kerja dari algoritma lzw itu bekerja. Merubah data yang sebelum nya berukuran original menjadi data yang terkompresi dan terenkripsi.

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

```
dictionary_size = 256
dictionary = {chr(i): i for i in range(dictionary_size)}
string = ""
compressed_data = []

for symbol in data:
    string_plus_symbol = string + symbol
    if string_plus_symbol in dictionary:
        string = string_plus_symbol
    else:
        compressed_data.append(dictionary[string])
        if(len(dictionary) <= maximum_table_size):
            dictionary[string_plus_symbol] = dictionary_size
            dictionary_size += 1
        string = symbol

if string in dictionary:
    compressed_data.append(dictionary[string])

output_file = open(fileName + ".lzw", "wb")
for data in compressed_data:
    output_file.write(pack('>H',int(data)))

output_file.close()
file.close()
```

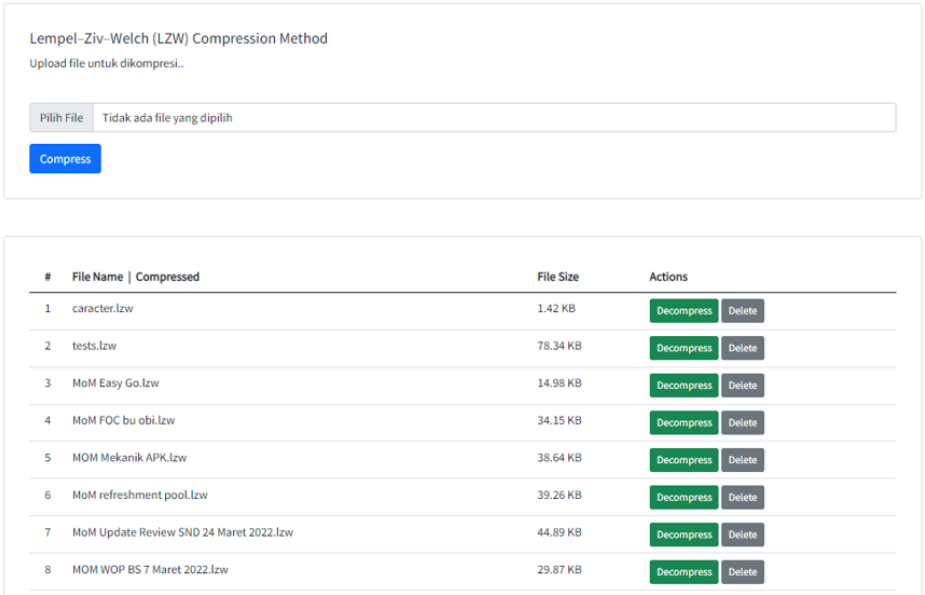
Gambar 3. Source Code Compresed

```
dictionary_size = 256
dictionary = dict([(x, chr(x)) for x in range(dictionary_size)])

for code in compressed_data:
    if not (code in dictionary):
        dictionary[code] = string + (string[0])
    decompressed_data += dictionary[code]
    if not(len(string) == 0):
        dictionary[next_code] = string + (dictionary[code][0])
        next_code += 1
    string = dictionary[code]

fileName = fileName.split(".")[0]
output_file = open(fileName + "_decoded.txt", "w")
for data in decompressed_data:
    output_file.write(data)

output_file.close()
file.close()
```

Gambar 4. Source Code Decompresed

2. Prototype Aplikasi

Tahap terakhir dari perancangan sistem adalah pembuatan prototype. Berikut adalah prototype dari sistem penyimpanan dengan dengan kompresi algoritma lzw.



Gambar 5. Prototype Aplikasi

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1

Pada gambar 5 terdapat proses input file / upload file yang didalam nya terdapat lzw untuk memperkecil nya dan terdapat beberapa file yang telah di perkecil, hasil yang dilihat pada list data diatas merupakan hasil ukuran yang telah di kompres, selain upload dihalaman ini juga terdapat menu download untuk mengunduh file hasil kompresi tadi tanpa mengurangi rahasia informasi didalam nya.

3. Hasil Penerapan Algoritma Lempel Ziv Welch

Data yang di gunakan dalam penelitian ini dalam data teks atau data yang ekstensi txt dapat dilihat pada tabel 1, alasan peneliti mengapa memilih data teks yang di gunakan adalah karena data teks untuk memprosesnya sederhana.

a. Ukuran Sebelum dilakukan kompresi data

Pada tabel 1 Terdapat beberapa file yang akan dilakukan pengujian dengan menggunakan algoritma lzw. Ukuran pada file disini merupakan ukuran default atau ukuran original pada file yang akan di ujicoba.

Tabel 1. Ukuran File Original

No	File	Ukuran	Ukuran Kompresi
1	Test1.txt	259 KB	1.42 KB
2	Test2.txt	73 KB	14.98 KB
3	Test3.txt	182 KB	34.15 KB
4	Test4.txt	569 KB	38.26 KB
5	Test5.txt	411 KB	39.26 KB
6	Test6.txt	550 KB	44.89 KB

b. Ukuran setelah di dekompresi

Pada tabel 2 Terdapat beberapa file yang akan dilakukan pengujian dengan menggunakan algoritma lzw. Ukuran pada file disini merupakan ukuran default atau ukuran original dan ukuran setelah di dekompresi oleh algoritma lzw. Pada proses dekompresi ini ukuran menjadi ke original.

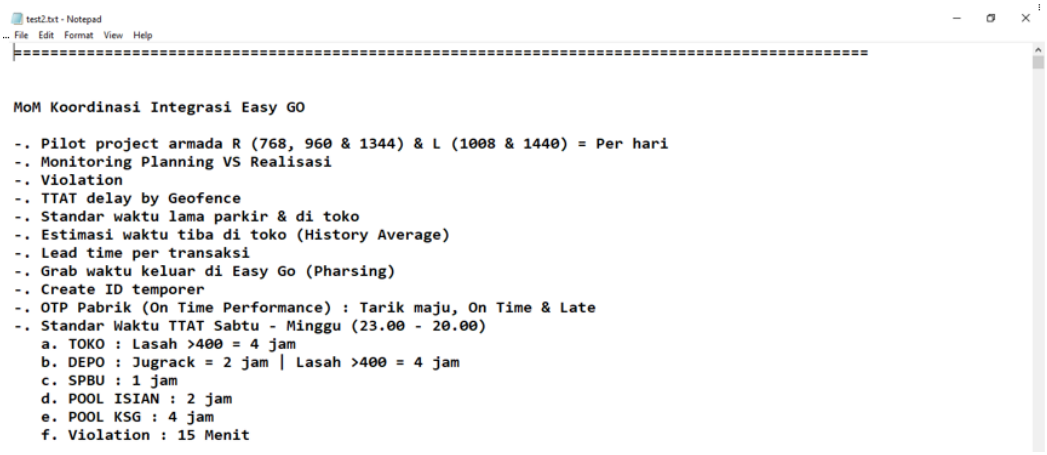
Tabel 2. Ukuran File Original dan Kompresi

No	File	Ukuran kompresi	Ukuran dekompesi
1	Test1.txt	1.42 KB	259 KB
2	Test2.txt	14.98 KB	73 KB
3	Test3.txt	34.15 KB	182 KB
4	Test4.txt	38.26 KB	569 KB
5	Test5.txt	39.26 KB	411 KB
6	Test6.txt	44.89 KB	550 KB

c. Isi file sebelum di lakukan kompresi

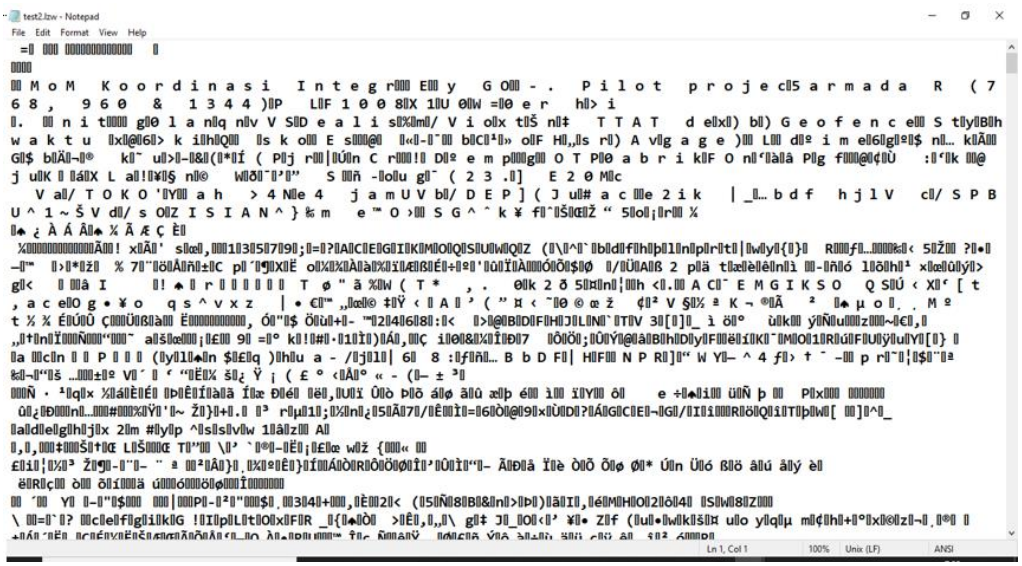
Pada gambar 6 merupakan isi dari dari file yang original, didalam file original ini terdapat informasi yang asli dari sebuah catatan hasil rapat. Dapat dilihat pada gambar catatan tersebut belum sama sekali terdapat enkkripsi pada isinya masih bisa dibaca dan tertata rapi.

Volume 14 Number 1, January 2024 Page. 66-73
Journal Homepage: http://teknois.unbin.ac.id/index.php/JBS
DOI Link: http://doi.org/10.36350/jbs.v14i1



Gambar 6. Isi File Original

d. Isi file setelah dilakukan kompresi
Pada gambar 7 merupakan hasil kompresi pada file yang dicontohkan pada gambar 6 dimana pada gambar 7 bisa dilihat isi dari file tersebut sudah menjadi karakter yang tidak beraturan atau bisa biasa disebut enkripsi pada data, dengan enkripsi ini menjadi manfaat dan keunggulan pada suatu data untuk mengamankan informasi didalam nya.



Gambar 7. Hasil Kompresi

4. Uji Ahli Sistem
Hasil pengujian untuk ahli sistem menggunakan kuesioner terbuka dan tertutup di mana pada kuesioner tersebut terdapat butir-butir pertanyaan terkait dengan Penerapan Algoritma Lempel Ziv Welch (LZW), Tabel 3 merupakan hasil kuesioner responden ahli sistem. Perhitungan skor yang digunakan untuk menghitung nilai pada kuesioner uji ahli menggunakan skala likert.

Tabel 3. Hasil Uji Sistem

No	Pertanyaan	R1	R2
1	Penerapan Algoritma Lempel Ziv Welch (Lzw) Untuk Kompresi Data sudah berjalan dengan baik	4	5
2	Algoritma Lempel Ziv Welch (Lzw) memiliki ke efektifan dalam Kompresi Data	4	4
3	Algoritma Lempel Ziv Welch (Lzw) Menghasilkan ukuran file menjadi lebih kecil	5	4

4	Algoritma Lempel Ziv Welch (Lzw) bisa merubah ke ukuran semula tanpa merusak data didalam nya	4	5
Jumlah		17	19
Nilai Tertinggi		20	20

$$\text{Persentase Kelayakan (\%)} = \frac{\text{skor yang di observasi}}{\text{skor yang diharapkan}} x\ 100$$

$$\text{Persentase Kelayakan (\%)} = \frac{17+19}{20+20} x\ 100$$

$$\text{Persentase Kelayakan (\%)} = \frac{36}{40} x\ 100$$

$$\text{Persentase Kelayakan (\%)} = 90\%$$

Persentase kelayakan yang didapat dari hasil kuesioner uji ahli adalah sebesar 90%, maka dapat dikategorikan “Sangat Layak”.

D. KESIMPULAN

Berdasarkan hasil analisis data kuesioner uji ahli sistem, Penerapan Algoritma Lempel Ziv Welch (Lzw) Untuk Kompresi Data, sangat efektif diterapkan pada sistem penyimpanan catatan rapat untuk mengatasi ketidak efektifan pada kompresi data dan tidak optimal pada penyimpanan dinyatakan sangat layak dengan nilai 90% persentase kelayakan dari total responden sebanyak 2 orang.

E. DAFTAR PUSTAKA

[1] Fahrizon, A., & Rony, M. A, “*Implementasi Algoritma Enkripsi Rsa Dan Kompresi Lzw Pada Database Nasabah*”, Jakarta : Pt. Central Capital Futures,2018.

[2] Hervindo C, " *Aplikasi Kompresi Data Dengan Algoritma Lzw Dan Pengamanan Data Dengan Algoritma Kriptografi Aes Pada Dropbox* ", Tangerang : Doctoral Dissertation, Universitas Buddhi Dharma, 2018

[3] Hidayat, H., Pamungkas, T., & Zarman, W," *Implementasi Algoritma Kompresi LZW Pada Database Server*", Komputa : Jurnal Ilmiah Komputer Dan Informatika , 2013

[4] Laia, Y , “*Optimasi Rasio Kompresi Dan Kompleksitas Waktu Kompresi File Teks Menggunakan Algoritma Lempel-ZIV-Welch Dengan Fibonacci Search*”, Sinkron: Jurnal Dan Penelitian Teknik Informatika., 2016.

[5] Purba, R. A., & Sitorus, L, “*Analisis Perbandingan Algoritma Arithmetic Coding Dengan Algoritma Lempel Ziv Welch (Lzw) Dalam Kompresi Teks*”, Medan : Jurnal Teknik Informatika UNIKA Santo Thomas, 2021

[6] Satyapratama, A., Widjianto, W., & Yunus, M, “*Analisis Perbandingan Algoritma Lzw Dan Huffman Pada Kompresi File Gambar Bmp Dan Png*”, Jurnal Teknologi Informasi:Teori, Konsep, Dan Implementasi, *6*(2), 69-81, 2015.

[7] Victor, A., & Herlambang, D. S., “*Implementasi Lempel Ziv Welch (Lzw) Untuk Kompresi Citra Digital Studi Kasus Di Museum Geologi Divisi Dokumentasi*”, Jurnal Komputer Bisnis, 13(1), 1-7, 2020.