METADATA FORENSIK UNTUK ANALISIS KORELASI BUKTI DIGITAL

TESIS

ZAENUDIN

15917124

Tesis diajukan sebagai syarat untuk meraih gelar Magister Komputer

Konsentrasi Forensika Digital

Program Studi Megister Teknik Informatika

Program Pascasarjana Fakultas Teknologi Industri

Universitas Islam Indonesia

2018

Lembar Pengesahan Pembimbing

Metadata Forensik Untuk Analisis Korelasi Bukti Digital

Zaenudin
15917127

Yogyakarta, Juli 2018

Pembimbing I	Pembimbing II
Dr. Bambang Sugiantoro, MT	Yudi Prayudi, S.Si., M.Kom

Lembar Pengesahan Penguji

Metadata Forensik Untuk Analisis Korelasi Bukti Digital

Zaenudin
15917127

Yogyakarta, Juli 2018

Tim Penguji,

Dr. Bambang Sugiantoro, MT
Ketua

Yudi Prayudi, S.Si., M.Kom
Anggota I

Dr. Imam Riadi M.Kom.
Anggota II

Mengetahui,
Ketua Program Pascasarjana
Fakultas Teknologi Industri Universitas Islam Indonesia

Ir. R. Teduh Dirgahayu, S.T., M.Sc.

Abstrak

Metadata Forensik Untuk Analisis Korelasi Bukti Digital

Metadata merupakan informasi yang tertanam pada sebuah file yang isinya berupa penjelasan tetang file tersebut. Penanganan barang bukti utama dengan pendekatan berbasis metadata masih banyak secara manual mencari korelasi file yang terkait untuk mengungkap berbagai kasus kejahatan komputer. Ketika file yang dikorelasikan berada dilokasi (folder) yang terpisah dan banyaknya file tentu akan menjadi tantangan yang berat bagi para invesitigator forensik dalam menganalisis barang bukti tersebut.

Penelitian ini, akan membangun sebuah prototife analisis menggunakan pendekatan berbasis metadata untuk menganalisis korelasi file bukti utama dengan file yang terkait atau di anggap relevan dalam konteks penyelidikan secara otomatis berdasarkan parameter metadata yaitu *Author, Size, File Type* dan *Date*. Penelitian ini dilakukan analisis terkait membaca karakteristik metadata file yaitu file type *Jpg, Docx, Pdf, Mp3 dan Mp4* dan analisis korelasi bukti digital dengan menggunakan parameter yang ditentukan, sehingga dapat memperbanyak temuan barang bukti dan mempermudah analisis barang bukti digital tersebut.

Hasil analisis korelasi bukti digital didapatkan bahwa menggunakan parameter *Author, Size, File Type* dan *Date* ditemukan file yang terkorelasi lebih sedikit yaitu sebanyak 2 file sedangkan dengan menggunakan parameter tanpa *Size* dan *File Type* ditemukan file yang terkorelasi lebih banyak yaitu 8 file karena beragam ekstensi dan ukuran file.

Kata kunci

Metadata, Forensik, File, Bukti Digital

Abstract

Correlation Analysis Of Forensic Metadata For Digital Evidence

Metadata is the information that is embedded in a file whose contents are the explanation of the file. Handling major evidence with a metadata-based approach still manually searches for correlation of related files to uncover various computer crime cases. When the correlated files are located in separate folders and the number of files will certainly be a formidable challenge for forensic investigators in analyzing the evidence.

This research will build an analysis prototype using a metadata-based approach to analyze the correlation of key evidence files with related files or deemed relevant in the context of investigation automatically based on metadata parameters ie Author, Size, File Type, and Date. This study analyzed the reading of the characteristics of the metadata file, namely file type Jpg, Docx, Pdf, Mp3 and Mp4 and analysis of digital evidence correlation using the specified parameters, so as to proliferate the evidence and simplify the analysis of digital evidence.

The result of analysis of digital evidence correlation found that using a parameter of Author, Size, File Type and Date found the less correlated file that is as much as 2 file whereas by using parameter without Size and File Type found the more correlated file that is 8 file because of various extension and size files

Keywords

Metadata, Forensic, File, Digital Evidence

Pernyataan Keaslian Tulisan

Dengan ini saya menyatakan bahwa tesis ini merupakan tulisan asli dari penulis, dan tidak berisi material yang telah diterbitkan sebelumnya atau tulisan dari penulis lain terkecuali referensi atas material tersebut telah disebutkan dalam tesis. Apabila ada kontribusi dari penulis lain dalam tesis ini, maka penulis lain tersebut secara eksplisit telah disebutkan dalam tesis ini.

Dengan ini saya juga menyatakan bahwa segala kontribusi dari pihak lain terhadap tesis ini, termasuk bantuan analisis statistik, desain survei, analisis data, prosedur teknis yang bersifat signifikan, dan segala bentuk aktivitas penelitian yang dipergunakan atau dilaporkan dalam tesis ini telah secara eksplisit disebutkan dalam tesis ini.

Segala bentuk hak cipta yang terdapat dalam material dokumen tesis ini berada dalam kepemilikan pemilik hak cipta masing-masing. Apabila dibutuhkan, penulis juga telah mendapatkan izin dari pemilik hak cipta untuk menggunakan ulang materialnya dalam tesis ini.

Yogyakarta, Juli 2018

Zaenudin, S.ST

Daftar Publikasi

Zaenudin, Sugiantoro.,B & Prayudi,Y. (2018). Correlation analysis of forensic metadata for digital evidence. International Journal of Computer Science and Information Security (IJCSIS), Vol. 16, No. 3, March 2018

Publikasi yang menjadi bagian dari tesis

Sitasi publikasi 1

Kontributor	Jenis Kontribusi
Author Zaenudin	Mendesain eksperimen (60%) Menulis *paper* (70%)
Author Yudi Prayudi	Mendesain eksperimen (40%) Menulis dan mengedit *paper* (30%)
Author Bambang Sugiantoro	Melakukan analisis statistik dari data di tabel 2 dan tabel 3

Halaman Kontribusi

Ada beberapa pihak terkait yang punya kontribusi dalam penyelesaian penulisan tesis ini:

- Bapak Dr. Bambang Sugiantoro, MT selaku Pembimbing I dan Bapak Yudi Prayudi, S.Si., M.Kom selaku Pembimbing II yang telah memberikan arahan-arahannya kepada penulis, sehingga penulisan tesis ini bisa selesai dengan baik dan tepat waktu.
- Bapak Ir. H. Lalu Darmawan Bakti, M.Sc., M.Kom selaku Ketua Sekolah Tinggi Manajemen Informatika Komputer (STMIK) Mataram yang telah memberikan motivasi dan biaya-biaya selama masa studi sampai dengan tahap pembuatan tesis ini.

Halaman Persembahan

Alhamdulillah Rasa syukur kehadirat Allah SWT atas limpahan Rahmat dan Hidayah-Nya.

Sholawat dan salam selalu tercurahkan keharibaan Baginda Nabi Besar Muhammad SAW.

Untuk ALM. Bapak dan Ibu tercinta,

Terima kasih atas segala yang diberikan, jerih payah dan dukungan, nasihat dan pengertiannya serta do'a dan kasih sayang-mu yang tiada terhingga yang tidak mungkin dapat ku-balas dengan selembar kertas bertuliskan kata cinta dan persembahan.

Untuk Kakak-kakak ku,

Terima kasih atas doa dan bantuan kalian selama ini.

Untuk Istri dan Anakku Muhammad Iqbal,

Terima kasih atas dukungan, Kasih Sayang dan pengertiannya selama ini,

Bapak Direktur STMIK-ASM Mataram yang telah memberikan kepercayaannya kepada Saya dalam melanjutkan Studi ini, Terima kasih banyak atas biaya dan bantuan-bantuan lainnya yang diberikan, semoga Saya berguna dan dapat memberikan sumbangsih dan bermanfaat bagi perkembangan STMIK-ASM Mataram kedepan.

dan

Untuk Sahabatku yang tidak bisa ku sebutkan satu-persatu namanya,

Terima kasih banyak atas dukungan dan masukannya selama pembuatan laporan ini.

Kata Pengantar

Assalamu'alaikum Wr. Wb.

Alhamdulillah, segala puji bagi Allah SWT atas segala rahmat, hidayah, dan inayah-Nya, sehingga penulisan laporan tesis sebagai salah satu syarat memperoleh gelar Pascasarjana Magister Informatika Fakultas Teknologi Industri Universitas Islam Indonesia yang berjudul "METADATA FORENSIK UNTUK ANALISIS KORELASI BUKTI DIGITAL" dapat penulis selesaikan dengan baik. Shalawat serta salam semoga senantiasa tercurah atas Nabi Muhammad SAW, para sahabat, serta pengikutnya hingga hari kiamat nanti.

Penyusunan laporan tesis ini tidak lepas dari bimbingan, dukungan, dan bantuan dari berbagai pihak. Oleh karena itu dalam kesempatan ini dengan segala kerendahan hati, penulis ingin menyampaikan ucapan terima kasih yang setulus-tulusnya kepada:

1. Allah SWT, yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis selalu diberi kesehatan dan kemudahan selama masa pengerjaan tesis ini.
2. Ibu, saudara, sahabat beserta keluarga besar yang telah memberikan do'a restu dan dukungannya.
3. Bapak Rektor dan seluruh jajaran Rektorat Universitas Islam Indonesia.
4. Dr. R. Teduh Dirgahayu, S.T., M.Sc. selaku Direktur Program Pascasarjana Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Dr. Bambang Sugiantoro, M.T selaku dosen pembimbing pertama yang telah memberikan pengarahan, bimbingan, masukan, serta dorongan semangat selama pelaksanaan tesis dan penulisan laporan.
6. Yusuf Yudi Prayudi, S.Si., M.Kom. selaku dosen pembimbing kedua yang telah memberikan pengarahan, bimbingan, masukan, serta dorongan semangat selama pelaksanaan tesis dan penulisan laporan.
7. Dosen-dosen Magister Informatika dan seluruh jajaran staf program Pascasarjana. Terima kasih atas semua ilmu pengetahuan, saran, motivasi, serta bantuannya.
8. Bapak Direktur STMIK-ASM Mataram yang telah memberikan kepercayaannya kepada Saya dalam melanjutkan Studi ini, Terima kasih banyak atas biaya dan bantuan-bantuan

lainnya yang diberikan, semoga Saya berguna dan dapat memberikan sumbangsih dan bermanfaat bagi perkembangan STMIK-ASM Mataram kedepan.

9. Rekan-rekan Angkatan XII. Terima kasih atas semua dukungan dan kerja samanya selama ini. Selamat berjuang.
10. Keluarga besar Magister Informatika. Terima kasih atas kerja samanya.
11. Teman-teman yang jauh di sana dan selalu mendoakan, terima kasih atas semuanya.
12. Semua pihak yang telah memberikan bantuan dan dorongan yang tidak dapat penulis sebutkan satu persatu.

Semoga segala kebaikan yang telah diberikan kepada penulis, akan dibalas Allah dengan yang lebih baik. Amin.

Penulis menyadari bahwa dalam penyusunan laporan tesis ini masih banyak terdapat kekurangan. Untuk itu penulis menyampaikan permohonan maaf serta sangat mengharapkan kritik dan saran yang membangun untuk penyempurnaan di masa mendatang.

Akhir kata semoga laporan ini dapat bermanfaat bagi kita semua. Amin.

Wassalamu'alaikum Wr. Wb.

Yogyakarta, Juli 2018

Zaenudin

Daftar Isi

Daftar Gambar

Daftar Tabel

BAB 1
Pendahuluan

1.1 Latar Belakang

Seiring *heterogenitas* bukti digital dalam penyelidikan terus berkembang dengan kemajuan teknologi, kita dihadapkan pada perangkat digital yang lebih baru, artefak yang lebih banyak dan berbagai macam format file, perkembangan tersebut membawa keuntungan, dan pada saat yang bersamaan memberikan peluang baru bagi kejahatan di bidang teknologi informasi (Raghavan & Raghavan, 2014a). Berbagai jenis pelanggaran dan tindak kejahatan dalam dunia komputer khususnya masalah data atau *file*, seperti pencurian data, penggandaan data, penghapusan data sampai dengan masalah memanipulasikan data yang semakin sering terjadi pada saat ini.

Berdasarkan data dari Jakarta Urban hosting kasus pencurian data memiliki tren yang semakin meningkat. Sejak tahun 2013 hingga saat ini kasus pencurian data telah mencapai 9 milyar akun. Setiap hari tercatat ada kasus pencurian data, sampai memanipulasi data. Di tahun 2017, terdapat 1.9 milyar kasus pencurian data sensitif melalui 918 insiden jika dibandingkan dengan tahun 2016 di priode yang sama, kasus pelanggaran data di tahun 2017 terdapat kenaikan 13 %.

Sesuai dengan Undang-undang Republik Indonesia Nomor 11 Tahun 2008 tentang Informasi dan Transaksi Elektronik bahwa informasi elektronik dan/atau dokumen elektronik dan/atau hasil cetaknya merupakan alat bukti hukum yang sah, maka peran digital forensik sebagai metode pembuktian suatu kasus kejahatan secara digital menjadi sangat penting. Sebagaimana tertuang dalam Penjelasan atas Undang-undang Republik Indonesia Nomor 11 Tahun 2008 tentang Informasi dan Transaksi Elektronik:

"..... pembuktian merupakan faktor yang sangat penting, mengingat informasi elektronik bukan saja belum terakomodasi dalam sistem hukum acara Indonesia secara komprehensif, melainkan juga ternyata sangat rentan untuk diubah, disadap, dipalsukan, dan dikirim ke berbagai penjuru dunia dalam waktu hitungan detik. Dengan demikian, dampak yang diakibatkannya pun bisa demikian kompleks dan rumit."

Dalam berbagai kasus yang terjadi saat ini, adanya barang bukti digital yang dapat membantu petugas dalam mengungkap suatu kasus tindak pidana. Salah satunya melalui informasi mengenai isi dari sebuah data atau file yang di sebut dengan metadata file.

Metadata merupakan informasi yang tertanam pada sebuah file berupa penjelasan tetang file tersebut. Metadata mengandung informasi mengenai isi dari suatu data yang dipakai untuk keperluan manajemen file atau data itu nantinya dalam suatu basis data (Riley, 2017). Metadata sering disebut "informasi tentang informasi" atau "data tentang data" (Riley, 2017).

Selama ini investigator analisis forensik dalam penanganan barang bukti utama dengan pendekatan berbasis metadata masih banyak secara manual dalam mencari korelasi file yang terkait. Namun, ketika file yang dikorelasikan berada dilokasi (folder) yang terpisah dan banyaknya file tentu akan menjadi tantangan yang berat bagi para invesitigator forensik dalam menganalisis barang bukti digital tersebut (Raghavan & Raghavan, 2014a).

Penelitian berbasis metadata yang pernah dilakukan antara lain dilakukan oleh (Spore, 2016) yaitu mengaitkan data dengan informasi lain, pengguna yang mengaksesnya, direktori file tempat penyimpanannya, terakhir kali dicopy, dan sebagainya. Penelitan berikutnya Melakukan analisis untuk memverifikasi metadata yang terkait dengan gambar dan melacak menggunakan fitur GPS (Kumar, Srikanth, & Sailaja, 2016).

Untuk mempermudah dalam proses analisis korelasi, Dalam penelitiannya membangun sebuah sistem analisis AssocGEN menggunakan metadata untuk menentukan asosiasi antara artefak file user, log dan pembuangan paket jaringan dan mengidentifikasi metadata untuk mengelompokkan dan menentukan korelasi antara artefak dan kelompok artefak yang terkait (Raghavan & Raghavan, 2013). Metadata forensik pernah dilakukan oleh penelitian sebelumnya tetapi dengan membangun tools dan parameter yang berbeda. Penelitian dengan berbasis metadata forensik pernah dilakukan oleh (Subli, Sugiantoro & Prayudi, 2017). Dalam penelitiannya membuat sebuah sistem metadata forensik untuk membaca karakteristik metadata secara umum dan mencari file-file korelasi metadata dengan salah satu parameter yaitu *file owner, file size,* f*ile date* dan *file type*. Menurut (Raghavan & Raghavan, 2013) Dengan menggunakan tools metadata forensik tentu akan sangat mempermudah investigator dalam menganalisis korelas bukti digital.

Sehingga dalam penelitian ini akan membangun sebuah prototipe untuk memahami dan membaca karakteristik metadata secara umum dan detail metadata yang spesifik dan mengidentifikasi, menganalisis korelasi metadata untuk mengelompokkan file yang terkait atau hubungan yang di anggap relevan dalam konteks penyelidikan secara otomatis berdasarkan parameter metadata yaitu *Author, Size*, *File Type* dan *Date*. Dengan menggunakan beberapa dan seluruh parameter yang sudah ditentukan, sehingga dapat memperbanyak temuan barang bukti dan mempermudah analisis barang bukti digital

tersebut. Dengan adanya penelitian ini diharapkan dapat memberikan konstribusi pada analis forensik dalam menganalisis korelasi bukti digital dengan pendekatan berbasis metadata.

2.1 Rumusan Masalah

Adapun rumusan masalah pada penelitian ini sebagai berikut.

a. Bagaimanakah karakteristik metadata file?
b. Bagaimana menentukan korelasi antara file Bukti Digital?
c. Bagaimana menguji kinerja sistem yang dibungun untuk melakukan analisis metadata file dalam proses investigasi?

3.1 Batasan Masalah

Batasan masalah pada penelitian metadata forensik ini sebagai berikut:

a. Membaca karakteristik metadata secara umum dan detail metadata.
b. Parameter korelasi metadata yang berdasarkan *Author*, file type *(type file)*, tanggal file *(file date)*, ukuran file *(file size)*.
c. File yang diuji dan dipahami ada beberapa jenis diantaranya file dengan ektensi *Docx, Pdf, Jpg, Mp3, Mp4* yang berada pada komputer bukan berdasarkan hasil *imaging/akuisisi*.

4.1 Tujuan Penelitian

Tujuan dalam penelitian metadata forensik ini antara lain:

a. Melakukan pembacaan metadata untuk memahami karakteristik metadata setiap file
b. Melakukan perancangan sistem untuk melakukan korelasi metadata file yang bisa mencari file-file yang terkait.
c. Melakukan pengujian kinerja sistem yang dibangun untuk melakukan analisis metadata file dalam sebuah proses investigasi barang bukti.

5.1 Manfaat Penelitian

Adapun manfaat yang diharapkan dalam penelitian metadata ini dapat memberikan konstribusi antara lain:

a. Mempermudah seorang analisa/investigator dalam memahami karakteristik metadata setiap file.
b. Mempermudah seorang analisa/investigator dalam menemukan file-file yang terkait dengan korelasi metadata file.

6.1 *Review* Penelitian

Berikut ini akan dibahas ulasan tentang penelitian yang telah dilakukan sebelumnya yang berkaitan dengan metadata antara lain.

(Salama, Varadharajan & Hitchens, 2012) menganalisis berbagai jenis metadata yang umum tersedia dengan objek digital seperti foto dan dokumen yang tersedia di Internet. Dan menganalisis berbagai jenis informasi metadata yang dihasilkan oleh kamera dan perangkat smartphone yang digunakan untuk menangkap dan menyimpan foto digital di web. Mengembangkan peraturan heuristik yang dapat digunakan untuk meningkatkan kualitas pengambilan keputusan dalam investigasi forensik. Dan menekankan perlunya melindungi metadata.

(Crossley, Asimakopoulou, Sotiriadis, & Bessis, 2013) dalam penelitiannya mengatakan penyimpanan data komputasi *cloud* dan dampak penandaan metadata sebagai metode potensial untuk melacak informasi file asli. untuk mengatasi masalah yang diajukan ke pemeriksa forensik yang mencoba menganalisis format file gambar yang dapat dipertukarkan dari ekstensi file nama jpg untuk gambar yang telah disimpan di dalam *cloud* untuk mengidentifikasi informasi.

Penelitian yang sama dilakukan oleh (Raghavan & Raghavan, 2013) dalam penelitiannya menyajikan mesin analisis AssocGEN yang menggunakan metadata untuk menentukan asosiasi antara artefak *file user*, *log* dan pembuangan paket jaringan dan mengidentifikasi metadata untuk mengelompokkan dan menentukan korelasi antara artefak dan kelompok artefak yang terkait.

Sementara itu penelitian yang dilakukan oleh (Woods, Cassanhoff & A Lee 2013) berfokus pada metadata yang dihasilkan oleh *tools open-source* yang mendukung Digital Forensik XML (DFXML). Bagaimana bagian-bagian dari metadata ini dapat digunakan saat merekam peristiwa PREMIS untuk menggambarkan kegiatan yang relevan dengan pelestarian dan akses dari metadata tersebut.

Penelitian lain juga dilakukan oleh (Raghavan & Raghavan, 2014) Melakukan analisis metode untuk secara otomatis mengidentifikasi asosiasi di antara bukti digital pada sintaksis dan tingkat semantik menggunakan metadata. penerapan metode ini untuk mengidentifikasi asosiasi metadata dari koleksi gambar dokumen-dokumen, dokumen pengolah kata dan menghasilkan interkom korelasi untuk tujuan mengidentifikasi atau file yang relevan dari kumpulan file besar dalam bukti digital. menunjukkan bahwa hubungan file yang diidentifikasikan dengan menggunakan bantuan metadata dalam mengidentifikasi foto-foto dan dokumen yang dicopy.

Selanjutnya (Alanazi & Jones, 2015) mengatakan bagaimana menggunakan berbagai format dan jenis metadata untuk mempalidasi berbagai jenis dokumen dan file yang memiliki sejumlah format dan jenis metadata, yang dapat digunakan untuk menemukan properti dari file, dokumen atau aktivitas sebuah jaringan. Selain itu pula, metadata banyak digunakan di kondisi apapun, dimana metadata dapat memberikan beragam bukti antara sekelompok orang, karena sebagian diantaranya tidak mengetahui jenis informasi yang tersimpan dalam dokumen mereka.

(Spore & Andy, 2016) mengatakan tujuan pemeriksaan forensik terhadap metadata yaitu mengaitkan data dengan informasi lain, pengguna yang mengaksesnya, direktori file tempat penyimpanannya, terakhir kali dicopy, dan sebagainya. Dalam sebuah kasus Metadata dapat menghasilkan bukti tidak langsung untuk mendukung barang bukti. Anda bisa melihat bagaimana file diakses, sesuai urutan, dan oleh siapa. Hampir semua tindakan yang Anda lakukan dengan sebuah file mengubah beberapa aspek metadatanya. Dengan analisis forensik yang tepat, metadata dapat membantu menyoroti pola, menetapkan *timelines*, dan menunjukkan kesenjangan dalam data.

Penelitian berikutnya dilakukan oleh (Kumar et al., 2016) mengatakan suatu hari sorang secara langsung dan tidak langsung sudah banyak sekali perangkat pintar yang melekat. Seseorang dapat menemukan keberadaannya jika kita memantau perangkat yang mereka gunakan dengan mengumpulkan metadata foto yang diposkan oleh mereka di media sosial. Beberapa situs sosial media memiliki fitur untuk memposting tempat masa lalu mereka. Untuk menyediakan aplikasi Android sederhana, ini akan menggunakan fitur *Geo tagging* yang tersedia dengan sebagian besar smartphone. Dan dengan menggunakan data berbasis lokasi ini bisa melacak orang berdasarkan garis bujur dan garis lintang dari *Global Positioning System* (GPS). bisa menggunakan ini untuk mengumpulkan foto yang diposkan oleh seseorang dan menganalisisnya untuk mengetahui posisi mereka saat ini. memverifikasi metadata yang terkait dengan gambar dan melacak di suatu negara, kota, *rute*, dan jalan negara berdasarkan Ketinggian GPS, GPS Latitude, GPS Bujur dan posisi GPS.

(Subli, Sugiantoro & Prayudi, 2017) penelitiannya dengan melakukan pendekatan metadata, maka diharapkan proses ini bisa melihat langsung metadata file secara umum dan juga dapat menemukan file-file berdasarkan korelasi file dengan parameter dari metadata file tersebut. Menggunakan salah satu parameter yang sudah ditentukan.

Pada penelitian ini konsep yang akan diusulkan dan yang membedakan dengan penelitian sebelumnya adalah pada segi teknik pencarian korelasi dengan memanfaatkan

metadata file dengan beberapa atau keseluruhan parameter yang sudah ditentukan bisa mengkorelasikan file yang satu dengan yang lainnya apabila file tersebut terkait.

Untuk lebih jelasnya tentang penelitian-penelitian sebelumnya dapat dilihat pada tabel 1.1 dibawah ini:

Tabel 1.1 *Literatur review* terhadap penelitian sebelumnya

No	Peper Utama	Metadata Object	Problem Penelitian	Pemecahan Masalah
1	(Alanazi & Jones, 2015)	- dokumen - jaringan	Informasi elektronik sering mengandung metadata yang tidak dapat dilihat saat melihat informasi menggunakan aplikasi dan alat yang secara konvensional terkait dengan jenis file.	mempalidasi berbagai jenis dokumen dan file yang memiliki sejumlah format dan jenis metadata, yang dapat digunakan untuk menemukan properti dari file, dokumen atau aktivitas sebuah jaringan.
2	(Crossley et al., 2013)	- .Jpg yang di simpan di *cloud*	Penyimpanan data dalam *cloud* dan dampak penandaan metadata sebagai metode potensial untuk melacak informasi keaslian file.	menganalisis format file gambar yang dapat dipertukarkan dari ekstensi file nama jpg untuk gambar yang telah disimpan di dalam *cloud* untuk mengidentifikasi keaslian.
3	(Raghavan & Raghavan, 2013)	- file user - log - paket jaringan	Sumber bukti digital di analisis dengan secara individual memeriksa berbagai artefak secara manual, pemeriksaan artefak dan metadata secara terpisah dan untuk mengkorelasikan artefak terkait membuat tanatangan yang sangat berat.	Menyajikan tools analisis AssocGen untuk mengidentifikasi metadata untuk mengelompokkan dan menentukan korelasi antara artefak dan kelompok artefak yang terkait. Untuk mempermudah pencarian korelasi metadata.

Tabel 1.1 *Literatur review* terhadap penelitian sebelumnya (Lanjutan)

No	Peper Utama	Metadata Object	Problem Penelitian	Pemecahan Masalah
4	(Woods, Cassanhoff & A Lee 2013)	- XML (DFXML)	Bagaimana bagian-bagian dari metadata ini dapat digunakan saat merekam peristiwa PREMIS untuk menggambarkan kegiatan yang relevan dengan pelestarian dan akses dari metadata tersebut menggunakan tools open-souce	*BitCurator project* untuk mengembangkan strategi *extensible* dalam mengubah dan menggabungkan metadata digital forensik ke dalam skema metadata arsip dan fokus pada metadata yang dihasilkan oleh *tools open-source* Digital Forensik (DFXML).
5	(Raghavan & Raghavan, 2014)	- Gambar - Dokumen-dokumen - Dokumen pengolahan kata - Foto	sistem analisis konvensional yang berkaitan dengan forensik digital, konten dianalisis untuk menggambarkan keadaan file dalam bukti digital dan memastikan relevansinya.	Melakukan analisis metode secara otomatis mengidentifikasi asosiasi di antara bukti digital pada sintaksis dan tingkat semantik menggunakan metadata, menghasilkan interkom korelasi file yang relevan dari kumpulan file besar dalam bukti digital. dilakukan dengan menggunakan " searching ". Saat mencari file-file, penggunaan kata kunci adalah norma. Bila kata-kata yang tepat tidak diketahui, seseorang dapat menggunakan pencarian ekspresi reguler yang menggunakan bahasa yang lebih fleksibel untuk menggambarkan sekumpulan kata kunci yang sesuai dengan sebuah pola

Tabel 1.1 *Literatur review* terhadap penelitian sebelumnya (Lanjutan)

No	Peper Utama	Metadata Object	Problem Penelitian	Pemecahan Masalah
6	(Salama, Varadharajan & Hitchens, 2012)	- foto dan dokumen yang berada di internet - camera dan Smartphone yang digunakan untuk di simpan di web	Meningkatnya aktifitas di media sosial dan online, masalah keamanan dan privasi menjadi sangat signifikan. Khususnya pada media sosial dan berbagi informasi dengan pengguna lainya.	menganalisis metadata yang bersumber dari internet yaitu foto, dokumen, camera, smartphone dan mengembangkan peraturan hauristik untuk pengambilan keputusan dan menekankan perlunya melindungi metadata.
7	(Spore & Andy, 2016)	- Direktori	Bagaimana mencari barang bukti yang berupa dokumen yang telah di hapus dan membuktikan salinan di servernya dengan tanggal yang berbeda. Dan Bagaimana melihat peranan metadata yang lebih besar pada proses pengadilan	Dengan cara mengaitkan data dengan informasi lain, pengguna yang mengaksesnya, direktori file tempat penyimpanannya, terakhir kali dicopy, dan sebagainya. metadata dapat membantu menyoroti pola, menetapkan *timelines*, dan menunjukkan kesenjangan dalam data.
8	(Kumar et al., 2016)	- foto di upload di sosial media - smartphone	Platform komunikasi di mana rata-rata individu terhubung ke enam perangkat yang mengaktifkan komputer. (Teks, Gambar, Audio dan Video). Siapa yang terlibat dengan gambar ini? (Siapa yang mengambilnya, siapa pemiliknya, siapa di dalamnya?) Apa yang menarik dari gambar ini? Dimana gambar ini? Kapan gambar ini dibuat atau dimodifikasi?	memverifikasi metadata yang terkait dengan gambar dan melacak menggunakan fitur GPS berdasarkan Ketinggian GPS, GPS Latitude, GPS Bujur dan posisi GPS. menggunakan fitur Geo tagging

Tabel 1.1 *Literatur review* terhadap penelitian sebelumnya (Lanjutan)

No	Peper Utama	Metadata Object	Problem Penelitian	Pemecahan Masalah
9	(Subli, Sugiantoro & Prayudi, 2017)	- docx, pdf, mp3, jpg, mp4, DD dan E01	Bagaimana merancang sistem dan menguji kinerja sistem metadata yang dibangun..	Membangun sistem untuk dapat membaca metadata file secara umum dan korelasi file dengan salah satu parameter dari metadata file tersebut. Dengan teknik model pencarian metadata setiap file pada komputer.
10	Usulan Penelitian	- docx, pdf, mp3, jpg, mp4, parameter metadata file berdasarkan pemilik (*Author),* ekstensi *(type file),* tanggal *(file date),* ukuran *(file size).*		
		penelitian ini akan membuat sistem atau prototipe, dari pendekatan berbasis metadata untuk memahami dan membaca karakteristik metadata secara umum dan untuk menentukan korelasi antara artefak, dan mengidentifikasi korelasi metadata untuk mengelompokkan artefak yang terkait atau hubungan yang di anggap relevan dalam konteks penyelidikan, secara otomatis berdasarkan parameter metadata yaitu *Author, Size, File Type* dan *Date.* Dengan menggunakan beberapa dan seluruh parameter yang sudah ditentukan, sehingga dapat memperbanyak temuan barang bukti dan mempermudah analisis barang bukti digital tersebut. Dengan adanya penelitian ini diharapkan dapat memberikan konstribusi pada analis forensik dalam menganalisis korelasi bukti digital dengan pendekatan berbasis metadata. Perbedaan dari penelitian sebelumnya dalam segi teknik implementasi, pada penelitian ini bisa berkorelasi dengan beberapa dan seluruh parameter untuk menentukan korelasi antara artefek digital berbasis metadata.		

7.1 Metodologi Penelitian

Dalam penelitian perlu disusun langkah-langkah penyelesaian penelitian secara sistematika yang disebut dengan metodologi, adapun metodologi yang digunakan pada penelitian ini sebagai berikut :

1. Identifikasi Masalah

Tahap identifikasi masalah adalah tahap awal dalam penelitian ini yaitu merumuskan masalah yang akan di jadikan sebagai objek penelitian

2. Tinjauan Pustaka

Pada tahapan Tinjauan pustaka dilakukan guna mencari literatur pendukung penelitian ini.

3. Metode Pengumpulan Data

Metode yang digunakan untuk mengumpulkan data pada penelitian ini yaitu dengan melakukan studi literatur. Studi literatur dilakukan untuk mencari semua informasi yang berkaitan tentang konsep metadata forensik dalam membaca atau memahami karakteristik metadata file dan memudahkan pencarian dalam korelasi metadata file, seperti membaca buku-buku, paper atau jurnal-jurnal dan mengunjungi situs-situs yang ada di internet yang berhubungan dengan metadata forensik

4. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem digunakan untuk mempermudah menganalisis sebuah sistem dibutuhkan dua jenis kebutuhan. *fungsional* dan kebutuhan *nonfungsional*. Kebutuhan *fungsional* adalah kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. Sedangkan kebutuhan *nonfungsional* adalah kebutuhan yang menitikberatkan pada properti prilaku yang dimiliki oleh sistem

5. Perancangan Sistem

Metode yang digunakan untuk membangun sebuah sistem atau metode algoritma metadata forensik ini yaitu dengan menggunakan metode perancangan terstruktur serta menggunakan Workflow (Bagan Kerja) dan Flowchart (Bagan Alir). Perancangan ini dimulai dari perancangan secara umum yang disebut dengan desain konseptual *(conceptuai design)* atau desain logikal *(logical design).*

6. Implementasi Sistem

Implementasi sistem adalah proses untuk memastikan bahwa sistem atau metode algoritma yang dibangun bebas dari kesalahan dan mudah digunakan oleh pengguna dalam hal ini seorang investigator.

7. Pengujian Tools

Pada tahapan ini dilakukan pengujian sistem metadata forensik yang bertujuan untuk mendeteksi kegagalan perangkat lunak sehingga kesalahan sistem dapat ditemukan dan diperbaiki

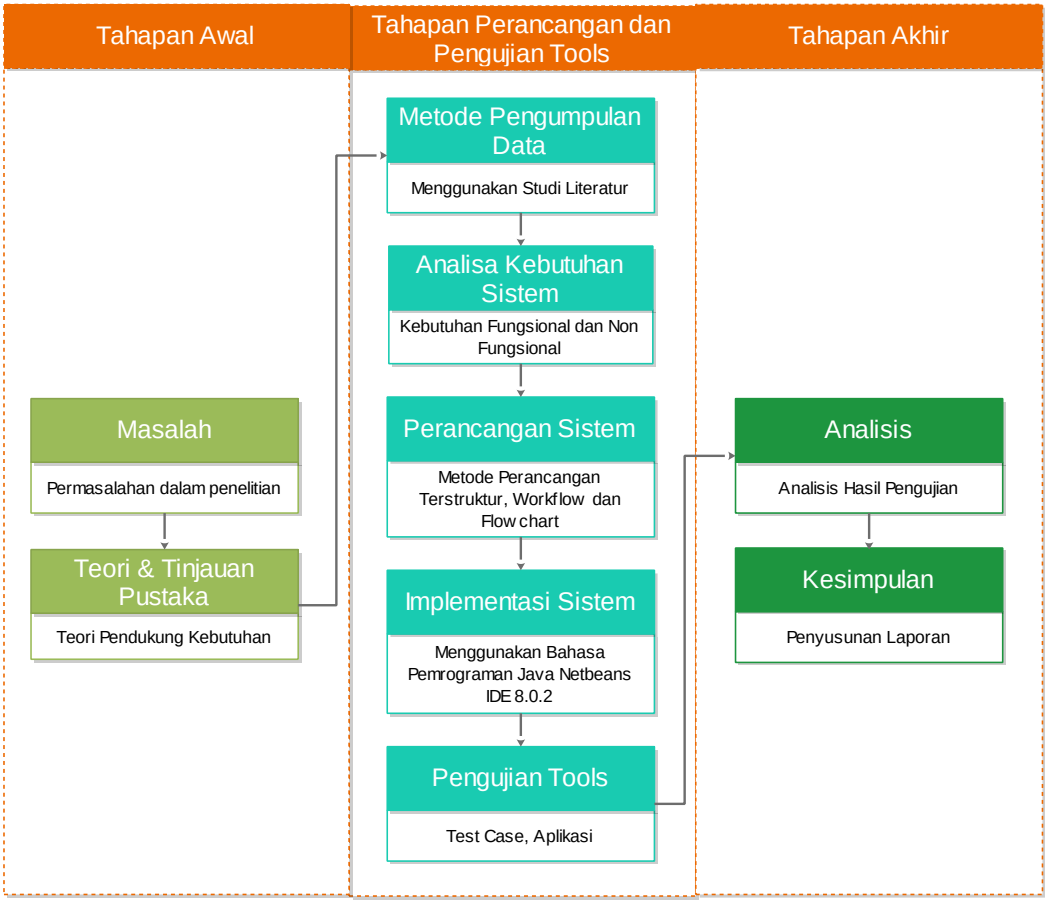
8. Analisis

pada tahap ini dilakukan analisis dari hasil yang sudah di ujikan menggunakan tools yang dibangun dengan parameter, seperti *Author, File Type, File Size, File date.*

9. Kesimpulan

Penyusunan laporan penelitian metadata forensik.

Berikut gambar 1.1 menampilkan tahapan metode penelitian rancangan sistem metadata forensik yang akan dibangun:



Gambar 1.1 Metodologi Penelitian

8.1 Sistematika Penelitian

Untuk memberikan gambaran dan mempermudah dalam penyusunan penelitian ini, maka dibuat sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Pendahuluan merupakan pengantar terhadap permasalahan yang akan dibahas. Didalamnya menguraikan tentang gambaran suatu penelitian yang terdiri dari latar belakang, rumusan masalah, batasan masalah, tuuan penelitan, manfaat penelitian, metodologi penelitian serta sistematika penulisan.

BAB II KAJIAN TEORI

Pada Bab ini menjelaskan tentang teori-teori dasar yang digunakan untuk memecahkan masalah dalam penelitian ini. Teori yang berkaitan dengan penelitian yang sedang diteliti.

BAB III METODOLOGI PENELITIAN

Bab ini membahas tentang langkah-langkah penelitian, kebutuhan perangkat lunak, perangkat keras dan bahan penelitian yang digunakan serta perancangan antar muka aplikasi yang akan dibuat.

BAB IV PEMBAHASAN

Pada Bab ini membahas tentang hasil dan pembahasan, terkait dengan pembahasan penyelesaian masalah yang diangkat, penentuan hasil analisis dan evaluasi dari penelitian yang diangkat.

BAB V PENUTUP

Pada bab ini memuat kesimpulan akhir dari semua proses penelitian sampai kepada hasil implementasi metode dan saran yang perlu diperhatikan karena keterbatasan dalam mendapatkan materi yang dibuat selama melakukan penelitian dan rekomendasi yang dibuat untuk pengembangan penelitian selanjutnya.

BAB 2
Landasan Teori

2.1 Metadata

Definisi metadata secara sederhana dapat diartikan sebagai data tentang data (data *about* data). Namun definisi tersebut masih belum lengkap karena metadata tidak sesederhana itu. Salah satu ciri dari metadata adalah data tersebut harus terstruktur. Jadi definisi yang tepat untuk menggambarkan metadata adalah data terstruktur tentang data (*structured* data *about* data). Definisi tersebut masih sederhana dan belum sepenuhnya menjelaskan lebih detail tentang metadata. *Task Force* on Metadata CC:DA (*committee on cataloguing: description and access*) dari ALA (*American library association*) menjelaskan secara lebih detail tentang metadata yaitu data yang terstruktur, ditandai dengan kode agar dapat diproses oleh komputer, mendeskripsikan ciri-ciri satuan-satuan pembawa informasi, dan membantu identifikasi, penemuan, penilaian dan pengelolaan satuan pembawa informasi tersebut.

Metadata adalah informasi yang ditanam pada sebuah file yang isinya berupa penjelasan tetang file tersebut. Metadata ini mengandung informasi mengenai isi dari suatu data yang dipakai untuk keperluan manajemen file atau data itu nantinya dalam suatu basis data (Putu Laxman Pendit 2007). Jika data tersebut dalam bentuk document docx metadatanya berupa keterangan mengenai *name file, content created, date last saved, content type, pages, word count, character count, line count, paragraph count, size, date created, date modified, date accessed, computer dan masih banya lagi. Jika dalam bentuk pdf metadatanya berupa name, type, foder path, size, date created, date modified, attributes, owner dan computer*. Untuk jenis data gambar jpg, metadata mengandung informasi mengenai siapa pemotretnya, kapan pemotretannya, dan setting kamera pada saat dilakukan pemotretan. Untuk audio jenis mp3 bisa tambahan metadatanya berupa *album, year, genre, length, bit rate* dan rekaman yang dipakai lainnya. Untuk jenis video mp4 metadatanya bisa berupa seperti mp3 dengan tambahan *frame width, frame height, date rate, total bitrate, frame rate, channels* dan jenis perekam video lainnya.

Metadata direkam komputer secara otomatis saat sebuah file dibuat, sehingga bisa diketahui kapan file dibuat, siapa user pembuatnya, berapa ukuran filenya, demikian juga *ekstensinya*. Namun demikian, metadata juga dapat disusun secara manual. Untuk mengedit dan membaca metadata sebuah file, digunakan software pengolah metadata.

2.2 Konsep Metadata

Metadata dapat diartikan sebagai "data tentang data (*spasial*)", berisikan informasi mengenai karakteristik data dan memegang peran penting di dalam mekanisme pertukaran data. Melalui informasi metadata diharapkan pengguna data dapat mengintepretasikan data secara sama, bilamana pengguna melihat langsung data spasialnya. Dokumen metadata berisikan informasi yang menjelaskan karakteristik data terutama isi, kwalitas, kondisi dan cara perolehannya. Metadata dipergunakan untuk melakukan dokumentasi data spasial yang berhubungan tentang siapa, apa, kapan, dimana, dan bagaimana data spasial dipersiapkan.

2.3 Jenis Metadata

Adapun jenis-jenis metadata file antara lain:

1. Metadata Deskriptif

Data yang dapat mengidentifikasi sumber informasi sehingga dapat digunakan untuk memperlancar proses penemuan dan seleksi. Cakupan yang ada pada data ini adalah pengarang, judul, tahun terbit, tajuk subjek atau kata kunci dan informasi lain yang proses pengisian datanya sama dengan katalog tradisional.

2. Metadata Administratif

Data yang tidak hanya dapat mengidentifikasi sumber informasi tapi juga cara pengelolaanya. Cakupan dari data ini adalah sama dengan data deskriptif hanya saja ditambah dengan pembuat data, waktu pembuatan, tipe file, data teknis lain. Selain itu data ini juga mengandung informasi tentang hak akses, hak kekayaan intelektual, penyimpanan dan pelestarian sumber informasi.

3. Metadata Struktural

Data yang dapat membuat antara data yang berkaitan dapat saling berhubungan satu sama lain. Secara lebih jelas, Metadata ini digunakan untuk mengetahui hubungan antara berkas fisik dan halaman, halaman dan bab dan bab dengan buku sebagai produk akhir.

2.4 Skema Metadata

Ada beberapa skema metadata file diantaranya sebagai berikut ini.

1. *Semantic*

Dalam kaitannya dengan metadata, semantik dapat diartikan sebagai makna kata. Lebih jelasnya adalah kesepakatan untuk membuat istilah yang digunakan untuk mewakili suatu makna. Selain itu, terkadang juga diberi keterangaan tentang status pada istilah tersebut.

2. *Content*

Dalam hal ini, konten bisa diartikan sebagai cara mengisi semantic. content tersebut bisa berupa peraturan untuk kriteria pengisian unsur skema atau peraturan untuk nilai-nilai unsur.

3. *Sintaksis*

Sintaksis dalam skema metadata dapat berarti sebagai machine readible (dapat dibaca mesin) atau dengan kata lain bahasa pemrogaman. Sehingga *semantic* dan content yang telah dibuat dapat dibaca oleh mesin.

2.5 Contoh Metadata

Berikut beberapa contoh metadata berdasarkan skema metadata:

1. CDWA *(Categories for Descriptions of Works of Art)*, skema untuk deskripsi karya seni
2. DCMES *(*Dublin Core Metadata Element Set), skema umum untuk deskripsi berbagai macam sumber digital.
3. EAD (Encoded Archival Description), skema untuk menciptakan sarana temu kembali pada *bahan kearsipan (archival finding aids) dalam bentuk elektronik.*
4. GEM *(*Gateway *to Educational Materials), skema untuk bahan pendidikan dan* pengajaran
5. MARC (Machine Readable Cataloguing), skema yang digunakan di perpustakaan sejak tahun 1960-an untuk membuat standar cantuman bibliografi elektronik.
6. METS (Metadata Encoding and Transmission Standard), skema metadata untuk obyek digital yang kompleks dalam koleksi perpustakaan
7. MODS (Metadata Object Description Standard), skema untuk deskripsi rinci sumber-sumber elektronik
8. MPEG (Moving Pictures Experts Group) MPEG-7 dan MPEG-21, skema untuk rekaman audio dan video dalam bentuk digital
9. ONIX (Online Information Exchange), skema untuk data bibliografi pada penerbit dan pedagang buku
10. TEI (Text Encoding Initiative), skema untuk encoding teks dalam bentuk elektronik menggunakan SGML dan XML, khususnya untuk peneliti teks di bidang humaniora.
11. VRA (Visual Resources Association), skema untuk deskripsi karya visual dan representasinya.

2.6 Pengertian Korelasi

Secara sederhana, korelasi dapat diartikan sebagai hubungan. Namun ketika dikembangkan lebih jauh, korelasi tidak hanya dapat dipahami sebatas pengertian tersebut. Korelasi merupakan salah satu teknik analisis dalam statistik yang digunakan untuk mencari hubungan antara dua variabel yang bersifat kuantitatif. Hubungan dua variabel tersebut dapat terjadi karena adanya hubungan sebab akibat atau dapat pula terjadi karena kebetulan saja. Dua variabel dikatakan berkolerasi apabila perubahan pada variabel yang satu akan diikuti perubahan pada variabel yang lain secara teratur dengan arah yang sama (korelasi *positif*) atau berlawanan (korelasi *negatif*).

Komputer berisi data dan program, Program merupakan file komputer yang digunakan untuk melakukan tugas tertentu, sedangkan data merupakan file hasil kerja program komputer yang dapat diedit, dibuka, dihapus, dan sebagainya. Sementara itu, folder adalah suatu tempat untuk mengumpulkan file

Dalam pengujian metode ini ada empat jenis korelasi metadata yang dijadikan sebagai contoh, yaitu metadata *file date*, *size*, *type file* dan *author*. Sesorang investigasi bisa mencari semua jenis file (tidak hanya lima jenis file yang telah dibahas diatas; Docx, Pdf, Jpg, Mp3 dan Mp4) yang ada didalam komputer berdasarkan dari empat pilihan korelasi tersebut.

2.7 Keunggulan dan Manfaat Metadata

Adapun kegunaan dan manfaat metadata yaitu:

1. Sebagai alat/*tool* pengelolaan investasi (data) seperti melakukan monitoring kemajuan pelaksanaan pekerjaan pembangunan data spasial, mendokumentasikan data data yang ada (selesai dikerjakan), menginformasikan data data yang dimiliki untuk dapat dimanfaatkan oleh pihak lain dan melakukan estimasi rencana kerja pengumpulan data dikemudian hari.
2. Sarana untuk menyebarluaskan kepemilikan data melalui mekanisme *clearinghouse*. Metadata merupakan faktor penting dalam konsep pemanfaatan data spasial bersama (data *sharing*).
3. Memberikan penjelasan (informasi) kepada pengguna data tentang tata cara pemrosesan dan mengintepretasikannya.

4. Metadata juga mengandung (berisikan) istilah istilah baku yang dipakai dalam kasanah *data* spasial. Dengan pembakuan istilah, kesalahan arti dalam penuturan data spasial dapat dihindari.

Untuk mencapai tujuan tersebut di atas, maka penyusunan metadata harus dipersiapkan dengan mempertimbangkan berbagai hal sedemikian hingga produk informasi yang dihasilkan dapat dimanfaatkan oleh berbagai pihak. Informasi metadata ditetapkan berdasarkan 4 (empat) karakteristik yang menentukan peranan dari metadata, yaitu :

1. Ketersediaan - informasi yang diperlukan untuk mengetahui ketersediaan data
2. Penggunaan - informasi yang diperlukan untuk mengetahui kegunaan data
3. Akses - informasi yang diperlukan tentang tatacara mendapatkan data
4. Transfer - informasi yang diperlukan untuk mengolah dan menggunakan data.

Pada tingkat global, terdapat beberapa tingkatan metadata yang biasa digunakan, yaitu :

1. *Discovery* metadata adalah informasi minimum yang diberikan untuk menjelaskan isi dari sumber data. Jenis metadata ini tentu saja tidak memenuhi kategori metadata yang bisa diaplikasikan pada tingkat internasional.
2. *Exploration* metadata adalah informasi yang lebih detil yang diberikan dalam menjelaskan isi dari sumber data. Jenis metadata ini diharapkan dapat membantu pengguna data untuk keperluan analisis
3. *Exploitation* metadata adalah metadata yang memuat informasi akses data, transfer data, *load* data, menginterpretasikan data dan penggunaan data untuk suatu aplikasi.

2.8 Konsep Analisis Metadata Forensik

2.8.1 pengertian File

File merupakan data yang ada pada komputer. Setiap data yang ada pada komputer dapat dikategorikan sebagai file. File tidak hanya terbatas pada data-data tertentu saja. Setiap data baik itu data gambar, data angka, data kata, data video, data suara, data aplikasi, dan data-data lainnya merupakan sebuah file.

File adalah kumpulan berbagai informasi yang berhubungan dan juga tersimpan di dalam secondary storage, secara konsep file memiliki beberapa tipe ada yang bertipe data terdiri dari *numeric, character dan binary*, lalu ada juga file yang bertipe program atau definisi file adalah arsip ataupun data yang tersimpan di dalam komputer.

File di komputer pada umumnya disimpan di dalam suatu folder tertentu tergantung dari pemilik komputer tersebut yang ingin dimana tempat menyimpannya, setiap file

memiliki ekstensi masing-masing tergantung jenis file itu sendiri. Ekstensi file adalah sebagai tanda yang membedakan jenis-jenis dari file.

Pengertian file menurut beberapa ahli, yaitu sebagai berikut:

1. Menurut Hendrayudi "File adalah data-data yang tersimpan dalam media yang mempunyai informasi besar file, tanggal & jam penyimpanan file, nama file, ciri file (ciri aplikasi yang membuat), dan *attribut file*."
2. Lalu menurut Rachmad Hakim S. "File merupakan dokumen yang mengandung informasi tertentu dan dapat dibuka dengan program."
3. Sindhunata "File adalah kumpulan catatan atau arsip."
4. Terus menurut Mcleod (PEARSON) "File adalah koleksi record yang saling berhubungan, seperti satu file dari seluruh record yang berisi field kode-kode mata kuliah dan namanya."
5. Sedangkan menurut Edi S. Mulyanta "File merupakan urutan data yang digunakan untuk melakukan *encode* informasi digital untuk urusan penyimpanan dan pertukaran data."

2.8.2 Jenis-jenis File di Komputer

Saat Anda mengklik kanan pada file dan memilih Properties, pada file komputer pasti ada tulisan tiga huruf sesudah titik. Itulah yang dinamakan *ekstensi file*. Fungsinya adalah untuk mengetahui atau membedakan jenis file. Untuk mengetahui *ekstensi file* lainnya Anda bisa membuka *Windows Explorer*, lalu pilih menu *View – Folder Options*.Pindah ke tab *Files Types*. Di sana terdapat puluhan dan mungkin ratusan ekstensi file. Semakin banyak Anda menginstall aplikasi maka daftar ekstensi file yang ada akan semakin panjang. Di antara beberapa ekstensi file di tunjukkan pada tabel 2.1 dan 2.2 berikut:

Tabel 2.1 Beberapa Ekstensi File

Ekstensi File	Jenis	Aplikasi
Doc	File dokumen	MS Word
JPG/JPEG *(Joint Photographic Experts Group)*	File gambar	PhotoShop, PhotoPaint, Paint, ACDSee dll
MP3	File audio	Winamp, Windows Media Player dll
PDF *(Portable (Document Format*	File dokumen adobe	Adobe raeder dll
MP4	File video	Pemutar video
PSD	File image, PhotoShop	PhotoShop

Tabel 2.2 Beberapa Ekstensi File (Lanjutan)

Ekstensi File	Jenis	Aplikasi
Gif *(Graphics Interchange Format)*	File gambar/animasi	PhotoShop, PhotoPaint, Paint, ACDSee, Ulead Gif Animator dll
Exe	File aplikasi (executable)	Sistem operasi windows
Asm	Source code pemrograman Assembly	Sembarang teks editor, seperti MS Word,NotePad,Wordpad

2.9 Klasifikasi Barang Bukti

Dalam investigasi adanya barang bukti sangatlah penting untuk keberlanjutan kasus yang sedang di investigasi, karena dengan adanya barang bukti itulah maka akan dilakukan analisa untuk mengungkap motif dan pelaku kejahatan tersebut.

Para investigator diharapkan dapat memahami jenis-jenis barang bukti sehingga pada saat melakukan proses investigasi mereka mengenali barang bukti yang menjadi prioritas untuk diutamakan.

Berikut akan dijelaskan klasifikasi barang bukti

1. Barang Bukti Elektronik

   Barang bukti elektronik merupakan barang bukti yang bersifat fisik dan dapat dikenali secara visual. Adapun jenis-jenis barang bukti elektronik adalah sebagai berikut ini:

   a. Komputer, PC/Leptop/Netbook, Notebook, Tablet
   b. Hanphone, Smartphone
   c. Flashdisk/Thumbdrive
   d. Harddisk
   e. CD/DVD
   f. Router, Switch ,Hub
   g. Floppydisk
   h. Camera Video, CCTV
   i. Camera Digital
   j. Digital Recorder
   k. Musik/Video Player, dan lain-lain.

2. Barang Bukti Digital

   Barang bukti digital merupakan setiap informasi pembuktian yang disimpan atau disalurkan dalam bentuk digital yang mana pihak dalam kasus hukum dapat gunakan untuk pemeriksaan pengadilan.

   Berikut contoh barang bukti digital :

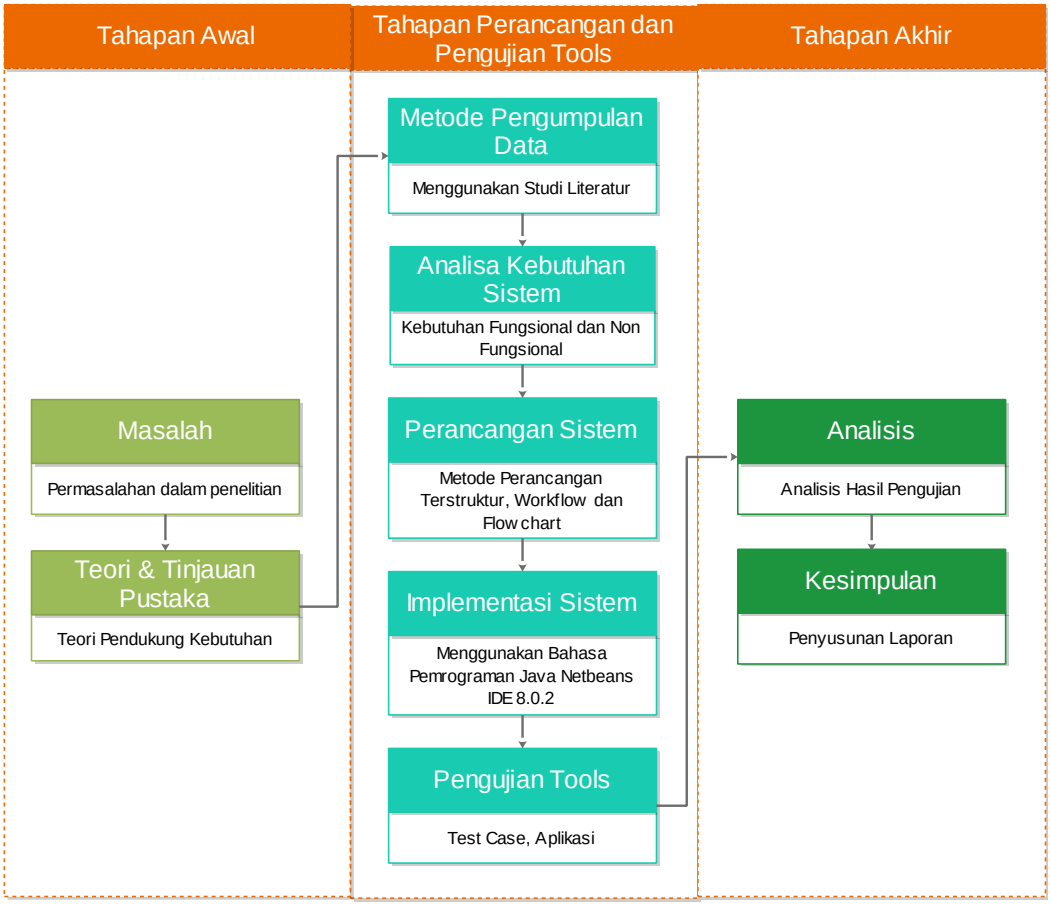
   a. *Logical file* yaitu file-file yang masih ada dan tercatat dalam file system yang sedang berjalan di suatu partisi.
   b. *Deleted file,* dikenal juga dengan istilah *unallocated cluster* yang merujuk pada *cluster* dan sektor tempat penyimpanan file yang sudah terhapus dan tidak teralokasikan lagi untuk file tersebut dengan ditandai dalam file *system* sebagai area yang dapat digunakan lagi untuk penympanan file-file baru.
   c. *Lost file* yaitu file yang sudah tidak tercatat lagi di file system yang sedang berjalan dari suatu partisi, namun file tersebut masih ada di sektor penyimpanannya.
   d. *File slack* yaitu sektor penyimpanan yang berada diantara *end of file* dengan *end of cluster.*
   e. *Log file* yaitu file-file yang merekam aktifitas dari suatu keadaan tertentu.
   f. *Encrypted file* yaitu file yang isinya sudah dilakukan enkripsi dengan menggunakan algoritma *kriptografi* yang kompleks, sehingga tidak bisa dibaca atau dilihat secara normal.
   g. *Steganography file* yaitu file yang berisikan informasi rahasia yang disisipkan ke file lain.
   h. *Office file* yaitu file-file yang merupakan produk dari aplikasi *office*.

3. Temuan Barang Bukti Digital

Temuan barang bukti merupakan Bukti digital lebih bermakna sebagai *output analysis* yang didapat oleh investigator yang langsung mengarah untuk kepentingan *rekontruksi* kasus yang sedang dihadapi. Dalam hal ini bukti digital adalah informasi yang langsung terkait dengan data-data yang diperlukan oleh investigator dalam proses penyidikan. Pada tahap akhir ini istilah yang lebih tepat adalah Temuan Bukti Digital menurut (Prayudi, 2014).

BAB 3
Metodologi Penelitian

Pada bab ini membahas tentang tatacara penelitian dimana terdapat rincian tentang urutan langkah-langkah yang dibuat secara sistematis, logis sehingga bisa dijadikan pedoman untuk menyelesaikan permasalahan berikut adalah gambaran tahapan metode penelitian rancangan sistem metadata forensik yang akan dibangun dapat dilihat pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

Berdasarkan gambar 3.1 diatas metodologi penelitian yang akan dibangun secara garis besar terbagi menjadi tiga tahapan, yaitu tahapan pertama terdiri dari identifikasi masalah dan tinjauan pustaka, tahapan kedua atau tahapan perancangan dan pengujian tools terdiri dari metode pengumpulan data, analisis kebutuhan sistem, perancangan sistem, implementasi sistem dan pengujian tools, analisis hasil dan tahapan yang trakhir yaitu tahapan penyelesaian berupa kesimpulan berisi penyusunan laporan penelitian. Berikut akan dijelaskan masing-masing tahapan

3.1 Masalah

Pada tahapan awal penelitian ini adalah bagaimana merumuskan masalah untuk dijadikan sebagai objek penelitian. Perumusan masalah dilakukan terlebih dahulu melihat situasi di lapangan. Setelah masalah dirumuskan selanjutnya menentukan tujuan penelitian. Tujuan penelitian merupakan sasaran yang ingin di wujudkan dari penyelesaian permasalah yang diteliti.

3.2 Teori dan Tinjauan Pustaka

Teori dan tinjauan pustaka dilakukan untuk mencari literatur tambahan penelitian. Pada tahapan ini mencari sumber-sumber literatur yang berada di internet yang berhubungan dengan metadata forensik, buku-buku, teori untuk mengumpulkan data dan *tools* yang digunakan dalam penelitian ini. Serta dijelaskan mengenai *tools* yang digunakan.

3.3 Metode Pengumpulan Data

Metode pengumpulan data yaitu metode yang digunakan untuk mengumpulkan data penelitian ini dengan melakukan studi literatur. Studi literatur digunakan untuk mengumpulkan data dari penelitian terdahulu dan dokumen lain seperti buku, jurnal dan teori-teori pendukung lainnya. Literatur yang berkaitan dengan metadata forensik yang diperoleh dari bahan acuan untuk dijadikan landasan kegiatan.

3.4 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem yaitu untuk menganalisis sebuah sistem dibutuhkan dua jenis diantaranya. Kebutuhan funsional dan non fungsional. Kebutuhan fungsional adalah kebutuhan yang berisi proses-proses yang dilakukan oleh sistem. Sedangkan kebutuhan fungsional adalah kebutuhan yang menitikberatkan pada properti prilaku yang dimiliki oleh sistem itu sendiri. Lebih jelasnya akan diterangkan dibawah ini.

3.4.1 Kebutuhan Fungsional

kebutuhan fungsional dalam analisis metadata forensik ini adalah sebagai berikut:

1. tools ini mampu memahami katarteristik metadata file
2. tools ini juga mampu merancang korelasi metadata antara file-file.

3.4.2 Kebutuhan Non Fungsional

kebutuhan non fungsional terdiri perangkat keras (*hardware*) dan perangkat lunak (*Software*) yang dunakan dalam membangun sistem metadata forensik ini seperti yang terlihat pada tabel 3.1 antara lain.

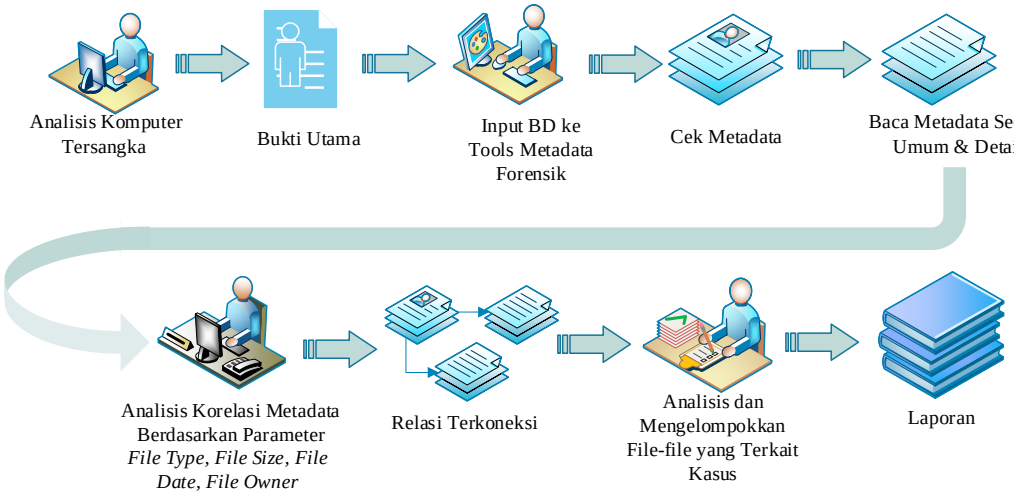
Tabel 3.1 Kebutuhan non fungsional

Perangkat Keras (*Hardware*)	**Perangkat Lunak (*Software*)**
Processor Intel(R) Core(TM) i3-3227U CPU @1.90GHz 1.90 GHz;	Sistem Operasi Windows 10 Enterprise 64 bit
RAM 8 gb	Netbeans 8.0.2
Harddisk 500 gb	Jenis file, Docx, PDF, Jpg, Mp3, Mp4

3.5 Perancangan Sistem

Pada perancangan sistem ini metode yang digunakan untuk membangun sebuah sistem atau *tools* metadata forensik yaitu menggunakan metode perancangan struktur serta menggunakan bagan kerja (*workflow*) dan bagan alir (*flowchart*). Perancangan ini dimulai dari perancangan secara umum disebut dengan desain konseptual (*conceptual design*) atau desain logical (*logical design*). Hasil dari perancangan sistem ini adalah bentuk esensial model yaitu apa yang dilakukan oleh sistem akan di implementasikan.

Adapun alur perancangan korelasi metadata file dengan parameter yang sudah ditentukan dari sistem/tools metadata forensik yang telah dibangun dapat dilihat pada gambar 3.2 berikut.

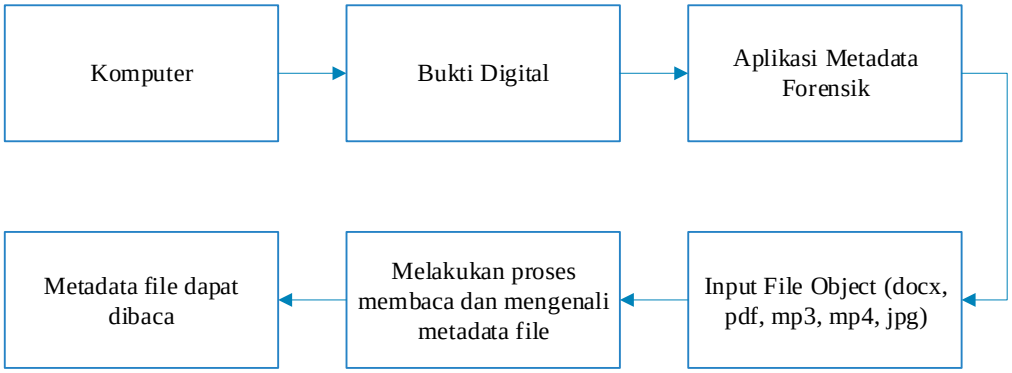


Gambar 3.2 alur rancangan korelasi metadata

Berikut penjelasan gambar 3.2 alur rancangan korelasi metadata forensik:

1. Penyidik atau ahli digital forensik melakukan pemeriksaan terhadap komputer/leptop tersangka.
2. Ditemukan satu barang bukti utama contoh berupa *image*
3. Kemudian setelah barang bukti image ditemukan, selanjutnya penyidik akan melakukan pengecekan terhadap metadata file tersebut menggunakan sistem metadata forensik/*tools* yang sudah dibangun.
4. Setelah metadata file tersebut dicek selanjutnya metadata file tersebut di baca secara umum.
5. Setelah metadata file tersebut di baca, selanjutnya dilakukan analisis korelasi metadata berdasarkan parameter yang sudah ditentukan.
6. Setelah dianalisis korelasi metadata Kemudian ditemukan file-file yang berkorelasi dengan file *image* bukti digital yang ditemukan pertama.
7. Setelah itu dilakukan analisis terhadap file-file yang berkorelasi untuk mengelompokkan file-file yang terkait dengan kasus kejahatan komputer.
8. Terakhir membuat laporan dari hasil analisis metadata forensik.

Untuk memahami karakteristik metadata setiap file dari sebuah sistem/tools metadata forensik yang telah dibangun berikut akan dijelaskan pada gambar 3.3 dibawah ini.



Gambar 3.3 Alur rancangan memahami karakteristik metadata file

Penjelasan gambar 3.3 alur rancangan memahami karakteristik metadata file dengan menggunakan sistem/tools metadata forensik sebagai berikut:

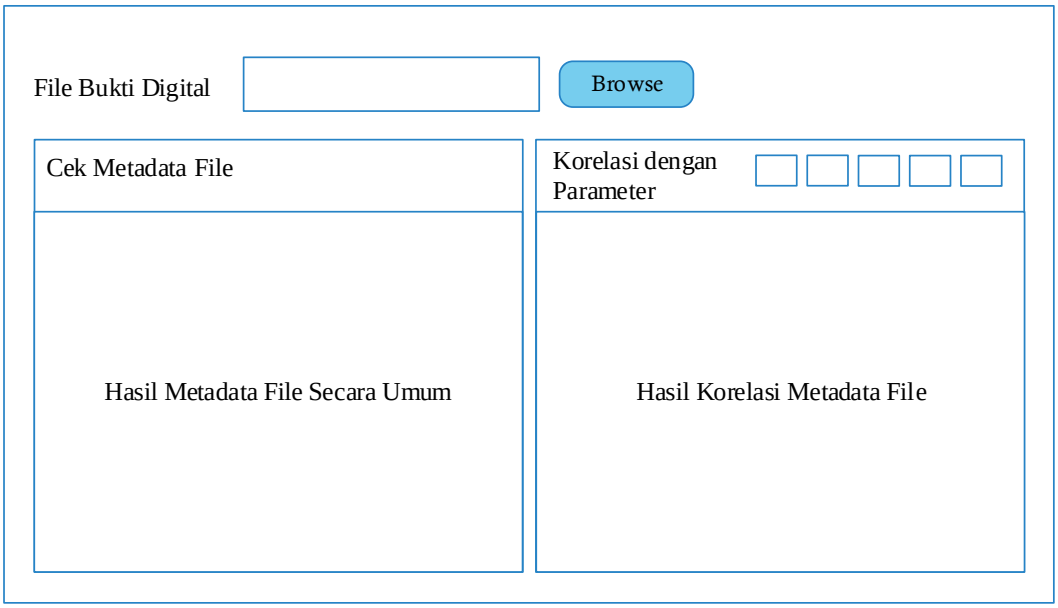
1. Pertama jalankan sistem operasi windows
2. Barang bukti digital ditemukan
3. Kemuadian jalankan tools metadata forensik yang sudah dibangun

4. Selanjutnya input sebuah file yang ingin di lihat metadatanya ke dalam tools yang sudah dibangun. Metadata yang yang akan dibaca yaitu file berikut docx, pdf, mp3, jpg, mp4
5. Kemudian melakukan proses membaca dan mengenali metadata file.
6. Setelah proses mengenali selesai, maka akan di tampilkan metadata file yang telah di inputkan tersebut.

3.6 Implementasi Sistem

Implementasi sistem merupakan proses dimana memastikan sebuah sistem atau tools yang dibangun terhindar dari kesalahan dan tentunya mudah untuk digunakan oleh seorang investigator. Dalam pembuatan sistem metadata forensik ini aplikasi yang akan digunakan adalah Netbeans IDE 8.0.2

Berikut adalah desain antarmuka sistem metadata forensik yang akan dibangun ditunjukkan pada gambar 3.4 dibawah ini.



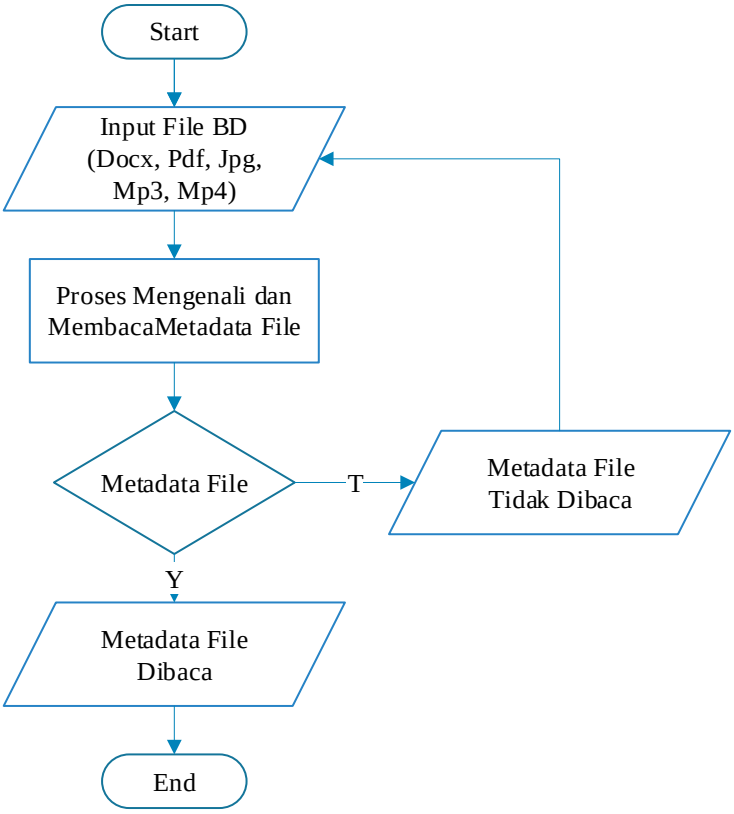
Gambar 3.4 Desain antarmuka tools metadata forensik

3.7 Pengujian Tools

pada tahapan ini dilakukan pengujian sistem atau *tools* metadata forensik untuk mendeteksi sejauh mana kegagalan atau kesalahan sistem dapat ditemukan dan diperbaiki. Pengujian perangkat lunak (*software testing*) merupakan suatu investigasi yang dilakukan untuk mendapatkan informasi mengenai kualitas produk yang sedang diuji (*under test*).

Dalam penelitian ini akan dilakukan dua proses pengujian sistem, dimana masing-masing tahapannya sebagai berikut:

1. Tahapan pertama yaitu proses pengujian yang dilukakan untuk membaca dan memahami karakteristik metadata file dalam sistem metadata forensik yang akan di gambarkan melalui *flawchart* Gambar 3.5 berikut ini:

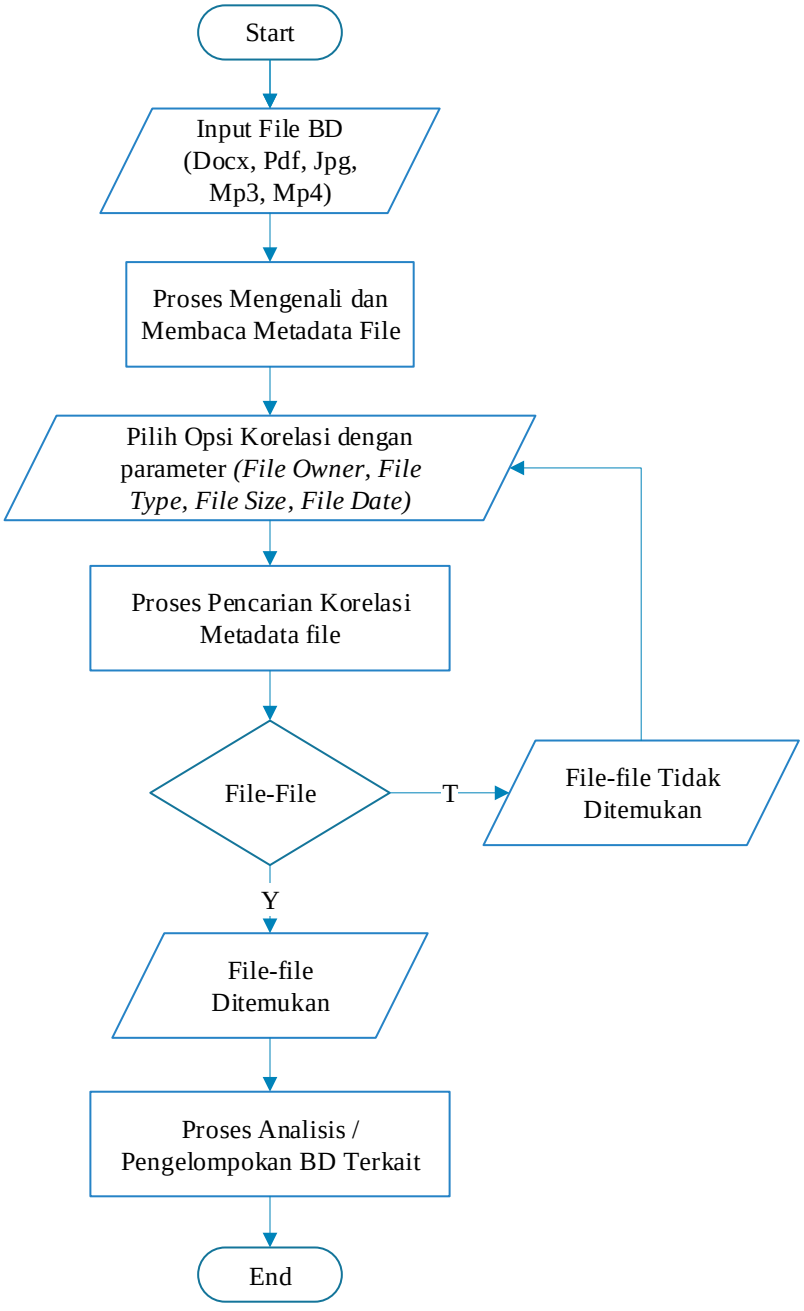


Gambar 3.5 Alur proses pengujian membaca atau memahami karakteristik metadata file

Penjelasan gambar 3.5 alur proses pengujian membaca atau memahami karakteristkk metadata file menggunakan sistem metadata forensik sebagai berikut.

1. Melakukan start atau sistem/*tools* metadata forensik dijalankan
2. Selanjutnya dilakukan pengimputan file barang bukti digital yang akan di baca atau dipahami metadatanya, dimana file yang akan dibaca yaitu file yang berekstensi docx, pdf, jpg, mp3, mp4.
3. Proses mengenali dan membaca metadata file
4. Kemudian program akan melakukan pemrosesan file yang telah di inputkan, terdapat kondisi dimana metadata file yang tidak bisa di baca akan kembali ke inputan file objek barang bukti.
5. Selanjutnya metadata file yang dapat dibaca akan langsung ditampilkan metadatanya.
6. Dan terakhir program di tutup atau selesai dijalankan.

2. Tahapan kedua yaitu proses pengujian yang dilakukan untuk mengkorelasikan metadata file berdasarkan parameter-parameter yang sudah ditentukan dalam sebuah sistem metadata forensik yang akan di gambarkan dalam bagan alur *flowchart* gambar 3.6 berikut ini



*Gambar 3.6 Alur Proses Pengujian Sistem/*Tools *Korelasi Metadata file*

Beriku penjelasan dari gambar 3.6 alur pengujian sistem/*Tools* korelasi metadata file sebagai berikut:

1. Melakukan start atau sistem/*tools* metadata forensik dijalankan

2. Selanjutnya dilakukan pengimputan file barang bukti digital yang akan di baca atau dipahami metadatanya, dimana file yang akan dibaca yaitu file yang berekstensi docx, pdf, jpg, mp3, mp4.
3. Kemudian proses mengenali dan membaca metadata file
4. Setelah proses metadata file di baca secara umum kemudan memilih opsi korelasi dengan parameter dari *file type, file owner, file size, file date*.
5. Kemudian sistem atau *tools* akan melakukan proses menemukan korelasi metadata file berdasarkan pemilihan parameter.
6. Kemudian terdapat kondisi atau pernyataan terdapat banyak file-file,
7. apabila korelasi file-file tidak ditemukan maka sistem akan kembali ke opsi koreasi metadata file,
8. tetapi apabila korelasi file-file ditemukan berdasarkan parameternya yang sudah ditentukan
9. akan dilanjutkan ke proses analisis/investigasi file-file yang berkorelasi yang sudah ditemukan tersebut guna untuk pengelompokan barang bukti terkait kasus.
10. Terakhir sistem selesai dan ditutup.

Setelah semua proses selesai dari proses awal sampai akhir, selanjutnya dilakukan penyusunan laporan-laporan dari penelitian ini.

3.8 Kesimpulan / Penulisan Laporan

Pada tahapan akhir ini terdapat kesimpulan yang memuat hasil dari penelitian ini. Dan pembuatan laporan, Laporan ini berisi tentang hal-hal yang di kerjakan selama penelitian dan hasil yang didapatkan pada saat melakukan penelitian. Dalam penulisannya, format yang digunakan berdasarkan format yang sudah diterapkan oleh Program Magister Teknik Informati Universitas Islam Indonesia Yogyakarta.

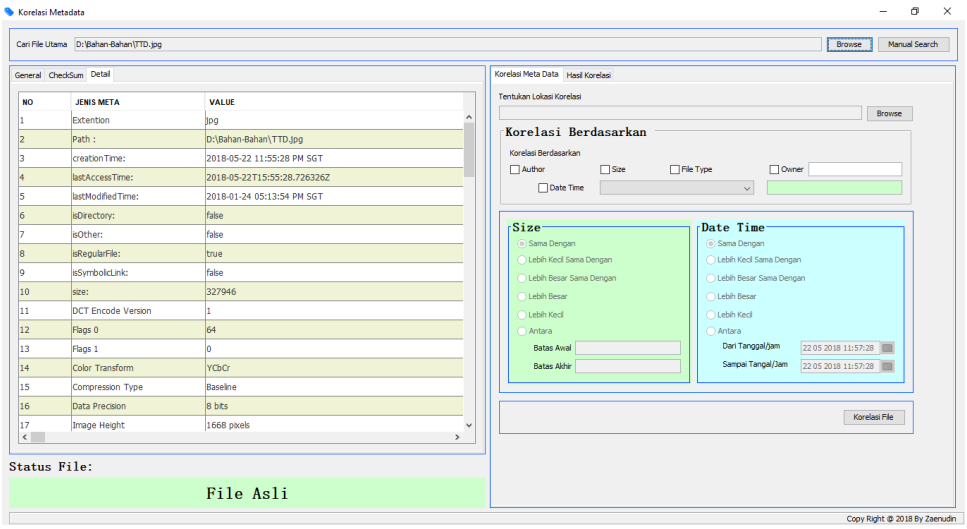
BAB 4
Implementasi dan Analisis

Pada penelitian tentang metadata forensik untuk analisis korelasi bukti digital, tahapan penelitian dimulai dari pembuatan aplikasi korelasi metadata kemudian dilakukan percobaan untuk membaca metadata secara umum dan mampu melakukan korelasi metadata dengan beberapa parameter yang ditentukan. Selanjutnya melakukan analisa terhapat hasil korelasi. Implementasi dan analisis pada penelitian ini menggunakan komputer dengan spesifikasi *hardware* dan *software* sebagai berikut.

a. Prosesor intel (R) Core (TM) i3-6100U CPU @2.30 Hz 2.30 GHz;
b. RAM 6,00 GB;
c. Hardisk 1 TB;
d. Netbeans IDE 8.0.2

4.1 Tampilan GUI Aplikasi Korelasi Metadata Forensik

Graphical User Interface (GUI) merupakan tampilan yang berfungsi sebagai window atau jendela yang menghubungkan pengguna dengan aplikasi korelasi metadata forensik. Pada penerapannya digunakan pembrograman netbeans 8.2 berbasis java dalam proses Run Program seperti gambar 4.1.



Gambar 4.1 tampilan aplikasi korelasi metadata forensik

Tombol browse pada cari file bukti utama digunakan untuk mencari barang bukti awal, kemudian menu *general*, *detail*, *check sum* berfungsi untuk menampilkan metadata secara umum dan *status file* berfungsi untuk manampilkan apakah file sudah dimofikasi atua masih asli. Kemudian pada menu korelasi metadata tombol browse digunakan untuk menentukan

lokasi korelasi, selanjutnya menetukan parameter yang digunakan untuk mencari korelasi metadata file tersebut. Tombol korelasi file untuk melihat hasil file-file yang terkorelasi dengan file awal barang bukti.

4.2 Jenis Karakteristik Metadata

Metadata terdiri atas berbagai jenis standar dalam menampilkan data. Secara sederhana yang dimaksud dengan standar metadata adalah satu set terminologi serta definisi umum yang digunakan dalam metadata serta dipresentasikan dalam format tersetruktur. Standar metadata spasial dibuat dan dikembangkan untuk mendefinisikan informasi yang diperlukan oleh seorang pengguna prospektif untuk mengetahui ketersediaan suatu set data spasial, mengetahui kesesuaian set data spasial untuk penggunaan yang diinginkan, mengetahui cara-cara pengaksesan data spasial serta untuk mentransfer set data spasial dengan sukses. Walaupun demikian standar tidak menetapkan tatacara bagaimana informasi diorganisasikan dalam suatu sistem komputer atau dalam suatu transfer data, tidak juga menetapkan tatacara bagaimana inforamsi tersebut ditransmisikan, dikomunikasikan atau disampaikan kepada pengguna. Jika standar metadata geospatial terkesan sangat komplek itu karena standar tersebut didesain untuk mendeskripsikan seluruh data geospasial yang bisa dideskripsikan.

Beberapa standar yang digunakan dalam pembuatan metadata spasial yaitu FGDC, ISO 19115, Dublin Core dan SNI metadata. Standar metadata ISO 19115/19139 merupakan standar untuk pembuatan metadata data geospasial. Format ISO 19115 merupakan standar internasional untuk metadata informasi geografi dan format ISO 19139 merupakan skema implementasi untuk ISO 19115. ISO 19115 merupakan 409 elemen dan terdapat 22 elemen inti (*core element*) yang dibutuhkan untuk mendeskripsikan data dan memiliki elemen compound (*role)* dibawahnya. Role tersebut terbagi menjadi 11 komponen utama, yang identifikasi, batasan, kualitas data, representasi spasial, sistem referensi, informasi data, referensi portal katalog, distribusi, informasi tambahan dan informasi skema (ISO 2003). Skema ISO 19139 digunakan untuk mendeskripsikan, melakukan validasi dan melakukan pertukaran metadata geospasial yang disiapkan dalam format XML (*Extensible Markup Language*).

Beberapa karakteristik metadata yang ditampilkan dalam sistem ini yaitu dibagi menjadi tiga katagori:

1. Metadata General yaitu Lokasi file, Nama file, Type file/*Extensi file*, *Outhors Owner* dan Computer.
2. Metadata Checksum yaitu Nilai MD5 dan SHA-256.

3. Metadata detail yaitu *cration time, last access time, last modified time, directory, other, regular file symbolic link, size, Make, Model, Orientation, X Resolution, Y Resolution, Resolution Unit, Software, Date/Time, Positioning, Exposure Time, F-Number, Exposure Program, ISO Speed Ratings, Exif Version, Date/Time Original, Date/Time Digitized, Components Configuration, Shutter Speed Value, Aperture Value, Brightness Value, Metering Mode, Flash, Focal Length, Sub-Sec Time, Sub-Sec Time Original, Sub-Sec Time Digitized, FlashPix Version, Color Space, Exif Image Width, Exif Image Height, Sensing Method, Scene Type, Exposure Mode, White Balance Mode, Focal Length, Scene Capture Type, Compression Type, Data Precision, Image Height, Image Width, Number of Components, Component 1, Component 2, Component 3, Interoperability Index, Interoperability Version, GPS Altitude Ref, GPS Time-Stamp dan GPS Date Stamp,* dll.

4.3 Jenis Korelasi Metadata File

Secara sederhana, korelasi dapat diartikan sebagai hubungan. Namun ketika dikembangkan lebih jauh, korelasi tidak hanya dipahami sebatas pengertian tersebut. Korelasi merupakan salah satu teknik analisis dalam statistik yang digunakan untuk mencari hubungan antara dua variabel yang bersifat kuantitatif. Hubungan dua variabel tersebut dapat terjadi karena adanya hubungan sebab akibat atau dapat pula terjadi karena kebetulan saja. Dua variabel dikatakan berkorelasi apabila perubahan pada variabel yang satu akan diikuti perubahan pada variabel yang lain secara teratur dengan arah yang sama (*korelasi positif*) atau berlawanan (*korelasi negatif*).

Komputer berisi data dan program, program merupakan file komputer yang digunakan untuk melakukan tugas tertentu, sedangkan data merupakan file hasil kerja program komputer yang dapat diedit, dibuka, dihapus dan sebagainya. Sementara itu, folder adalah suatu tempat untuk mengumpulkan file.

Dalam pengujian metode ini ada empat jenis korelasi metadata yang dijadikan sebagai contoh yaitu metadata file date, file size, file type dan Author. Seorang investigator dapat mencari semua jenis file yang ada di dalam komputer berdasarkan empat parameter tersebut.

4.4 Live Data dalam Proses Investigasi Metadata

Live data adalah format data didalam sebuah sistem yang dapat diakses langsung oleh pengguna. Dalam pengumpulan barang bukti live data cenderung lebih mudah karena data yang didapatkan secara langsung dapat dilihat dan dianalisa lebih lanjut lagi. Secara umum

live data mempunyai nilai yang kuat untuk dijadikan barang bukti karena data yang diperoleh terbukti secara langsung dan dilihat secara langsung serta berhubungan dengan sesuatu maupun tindakan apa yang telah dilakukan terhadap file tersebut. Selanjutnya karena live data dibuat dan dikelola oleh sistem opaerasi dan aplikasi perangkat lunak. Live data memiliki catatan waktu yang akurat, tergantung dengan kesesuaian jam yang ada dalam perangkat tersesebut.

Proses yang terjadi dalam sebuah sistem mempunyai catatan waktu. Catatan waktu yang terjadi biasanya dikenal dengan istilah MAC *(Modified, Accessed, Created).*

1. *Modified* adalah catatan yang menampilkan waktu kapan terakhir kali dimodifikasi, yaitu ketika terakhir kali disimpan. Modified menampilkan waktu terakhir perubahan file.
2. *Accessed* adalah catatan untuk menampilkan kapan waktu terakhir kali file tersebut diakses.
3. *Created* adalah catatan yang menampilkan kapan data tersebut dibuat pertama kalinya.

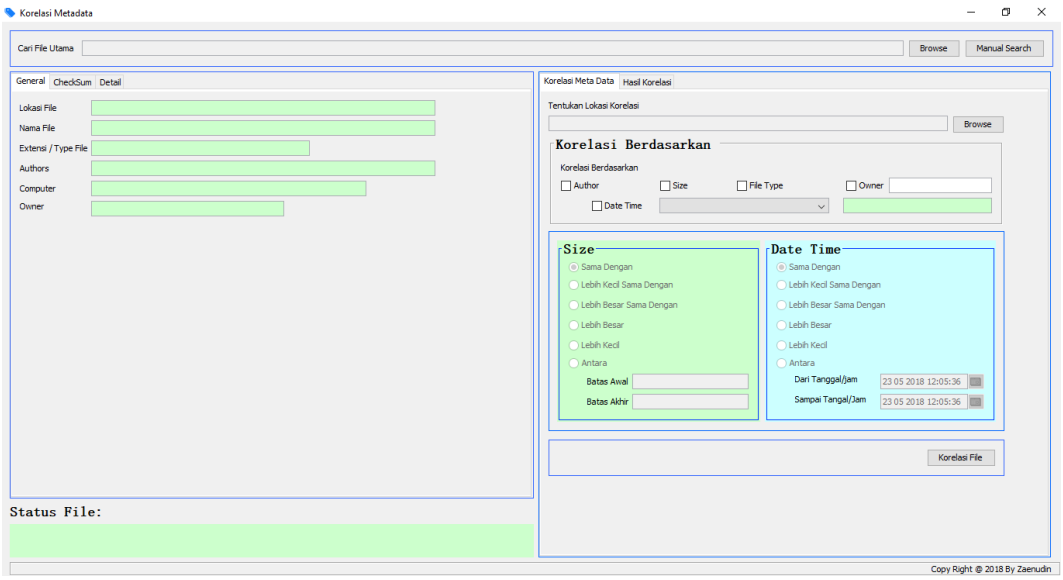
4.5 Pengujian sistem Metadata Forensik

Pada penelitan metadata forensik untuk analisis korelasi barang bukti meliputi beberapa tahapan pengujian yaitu tahap awal program, pengujian untuk membaca karakteristik metadata secara umum dan pengujian untuk melakukan korelasi metadata.

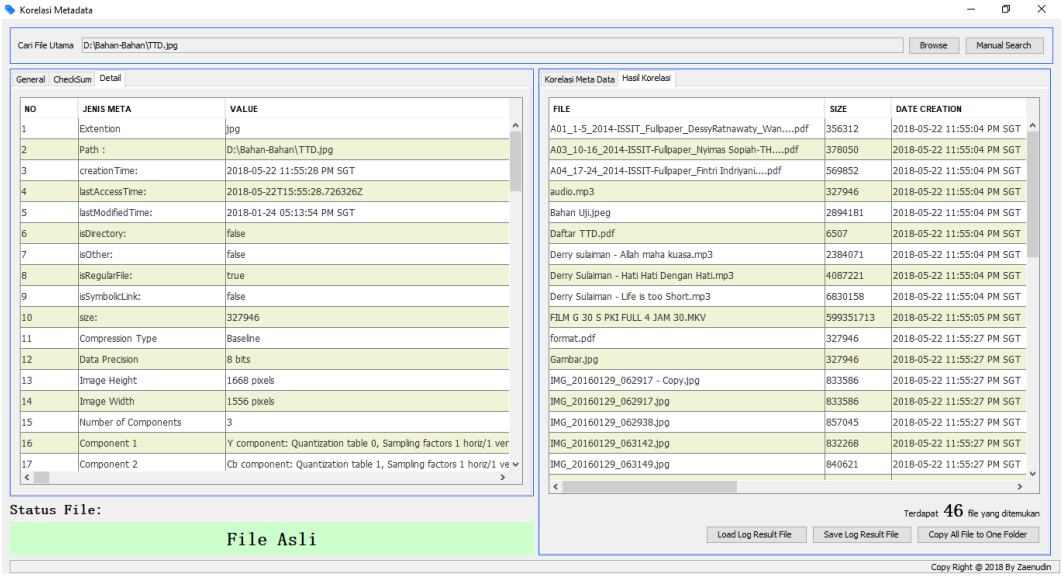
4.5.1 Tahap Awal Program

Tahap awal program metadata forensik ketika mulai di jalankan yaitu akan menampilkan Cari file bukti digital untuk mencari file bukti utama yang akan di browse dan dan menu general untuk melihat metadata file secara umum, Menu *Detail* untuk melihat metadata file secara lebih jelas lagi dan Menu *Checksum* untuk melihat nilai *hashing* suatu file dan Status file untuk melihat file masih asli atau sudah dimodifikasi. Selanjutnya Menu Korelasi Metadata didalamnya pertama harus menentukan lokasi pencarian file dan pilihan korelasi file dari empat parameter metadata file yaitu *Author*, *Size*, *Date*, *File type* dan *Owner* dan tombol korelasi file untuk menuju menu hasil korelasi file (tempat yang sudah disediakan untuk file-file yang sudah ditemukan, berdasarkan pilihan dari menu korelasi file yang sudah ditentukan). Dan terdapat beberapa buttom seperti *Copy all file to one folder* dan di file hasil korelasi bisa klik kanan *open file* dan *open location file, Buttom Save Log Result File dan Load Log Result File.*

Berikut adalah tampilan dari masing-masing program ketika baru dijalankan, seperti gambar 4.2 dibawah ini:



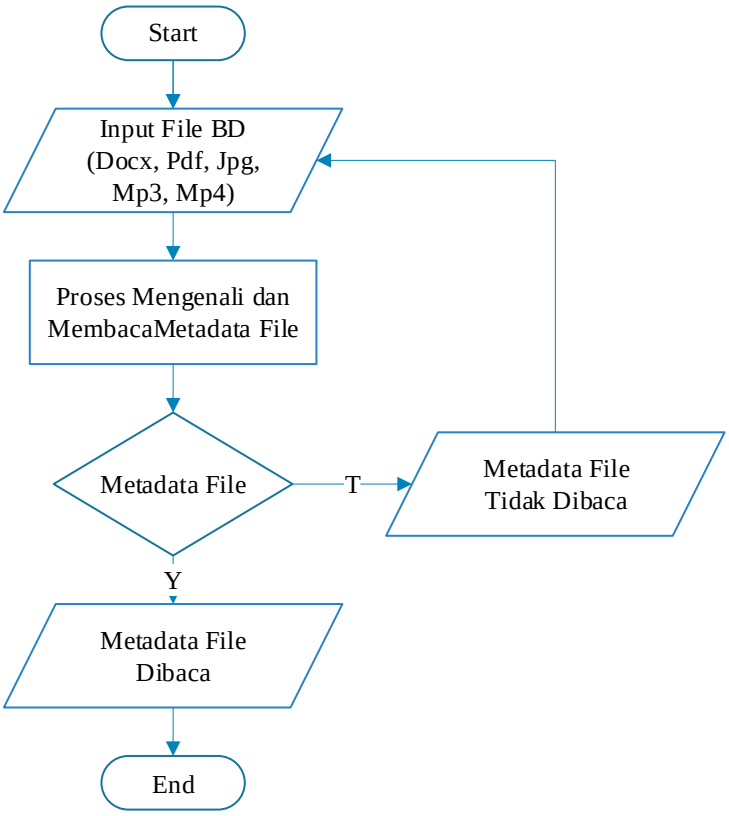
Gambar 4.2 Tampilan Awal Program Metadata Forensik



Gambar 4.3 Tampilan Hasil Program Metadata Forensik

4.5.2 Membaca Karakteristik Metadata File

Berikut dijelaskan secara detail langkah-langkah penggunaan aplikasi ini dalam melihat karakteristik metadata file pada gambar 4.4 *flowchart* dibawah:



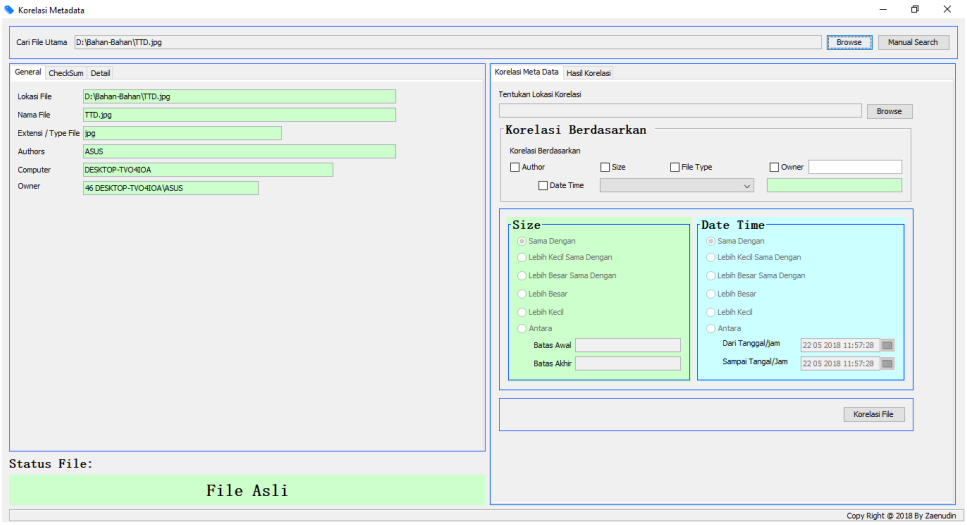
Gambar 4.4 Flowchart Membaca Karakteristik Metadata File

Penjelasan gambar 4.4 alur proses pengujian membaca atau memahami karakteristkk metadata file menggunakan sistem metadata forensik sebagai berikut.

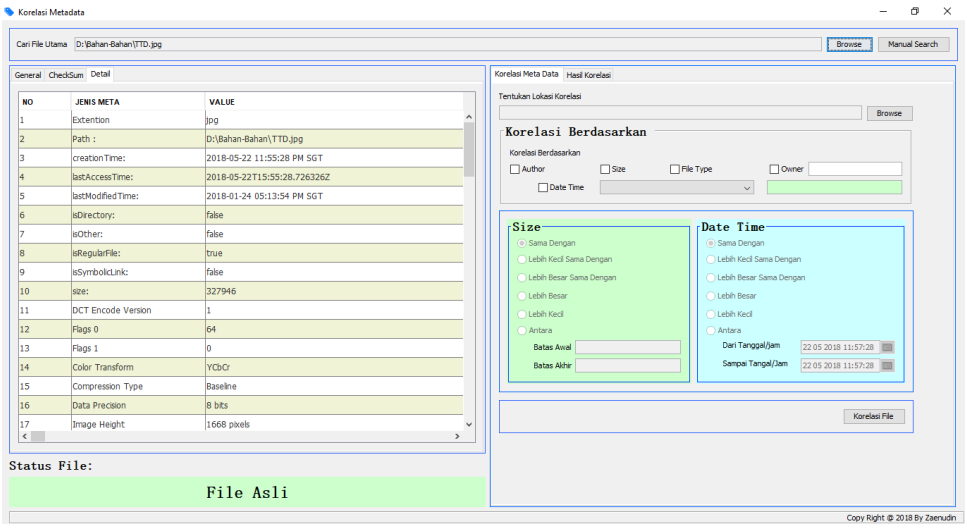
1. Melakukan start atau sistem/*tools* metadata forensik dijalankan
2. Selanjutnya dilakukan pengimputan file barang bukti digital yang akan di baca atau dipahami metadatanya, dimana file yang akan dibaca yaitu file yang berekstensi docx, pdf, jpg, mp3, mp4.
3. Proses mengenali dan membaca metadata file
4. Kemudian program akan melakukan pemrosesan file yang telah di inputkan, terdapat kondisi dimana metadata file yang tidak bisa di baca akan kembali ke inputan file objek barang bukti.
5. Selanjutnya metadata file yang dapat dibaca akan langsung ditampilkan metadatanya.
6. Dan terakhir program di tutup atau selesai dijalankan.

File Bukti Utama yang akan dibaca metadatanya, terlebih dahulu akan di browse atau dicari untuk diperiksa metadatanya, setelah itu baru kemudian program akan memproses file tersebut sampai teridentifikasi metadatanya satu persatu, kemudian akan tampil keterangan metadatanya di tabel *detail* metadata, seperti contoh mencoba mem *browse* sebuah file bukti

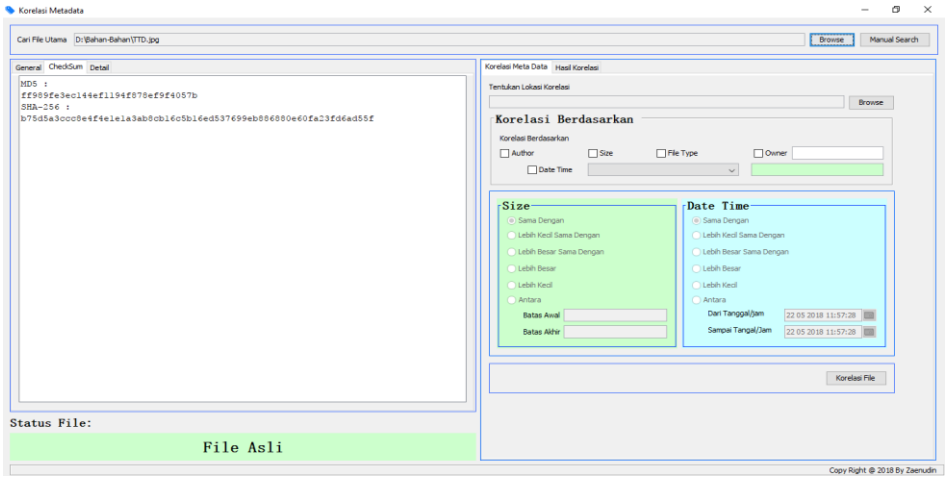
utama yang ada di dalam folder Bahan-bahan di drive D, hasilnya dapat dilihat seperti gambar 4.5, 4.6 dan 4.7 dibawah ini.



Gambar 4.5 menampilkan metadata general secara umum



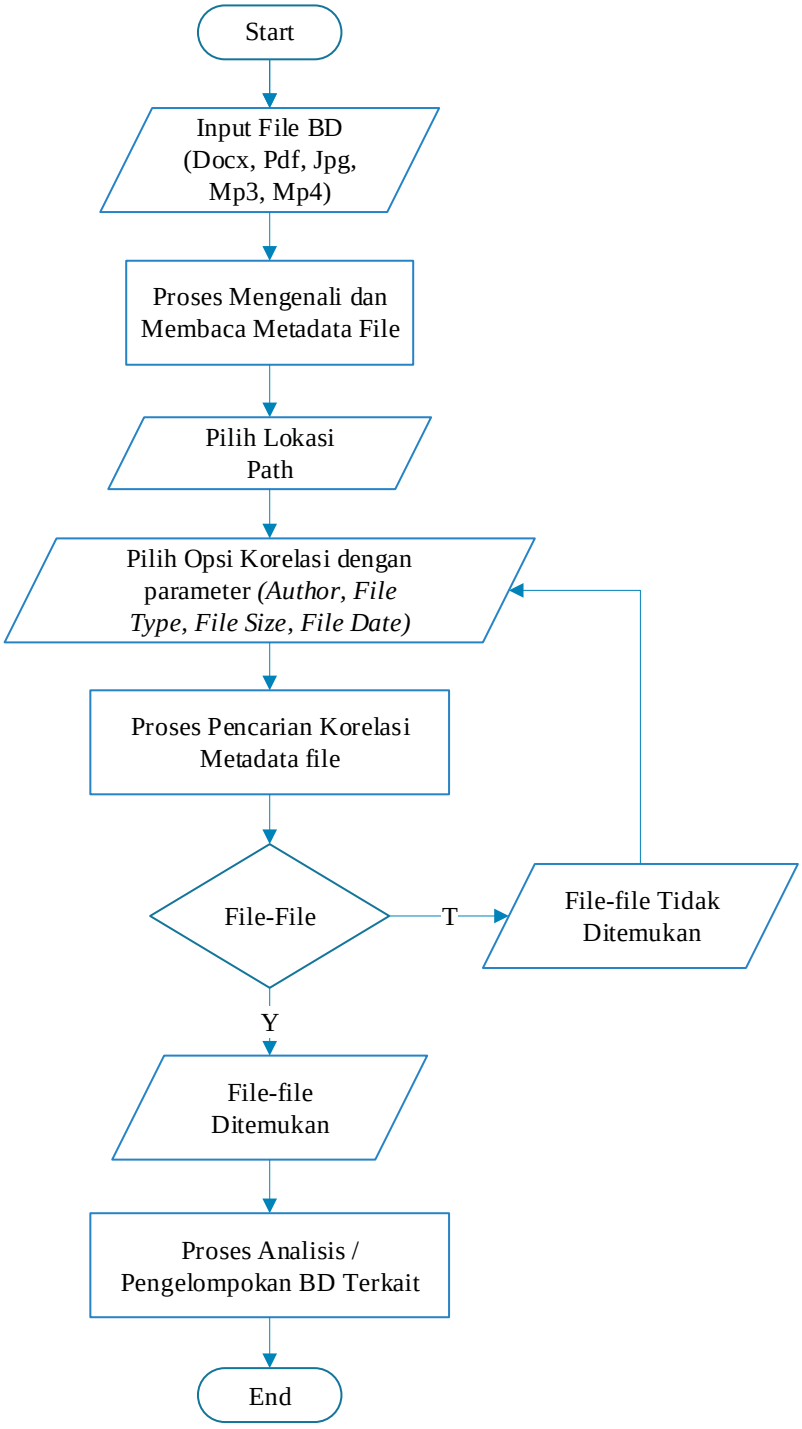
Gambar 4.6 Menampilkan metadata *detail*



Gambar 4.7 Metadata *checksum*

4.5.3 Melakukan Pengujian Korelasi File

Berikut dijelaskan secara rinci langkah-langkah penggunaan program aplikasi ini untuk melakukan korelasi file dalam gambar 4.8 flowchart dibawah:



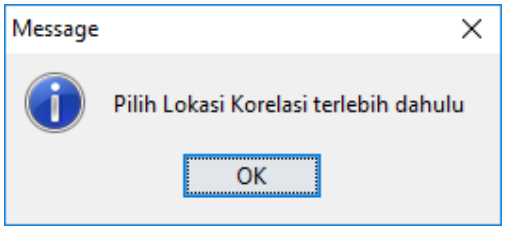
*Gambar 4.8 Alur Proses Pengujian Sistem/*Tools *Korelasi Metadata file*

Berikut penjelasan dari gambar 4.8 alur pengujian sistem/*Tools* korelasi metadata file sebagai berikut:

1. Melakukan start atau sistem/*tools* metadata forensik dijalankan
2. Selanjutnya dilakukan pengimputan file barang bukti digital yang akan di baca atau dipahami metadatanya, dimana file yang akan dibaca yaitu file yang berekstensi docx, pdf, jpg, mp3, mp4.
3. Kemudian proses mengenali dan membaca metadata file
4. Pilih lokasi path
5. Setelah proses metadata file di baca secara umum kemudan memilih opsi korelasi dengan parameter dari *file type, Author, file size, file date.*
6. Kemudian sistem atau *tools* akan melakukan proses menemukan korelasi metadata file berdasarkan pemilihan parameter.
7. Kemudian terdapat kondisi atau pernyataan terdapat banyak file-file,
8. apabila korelasi file-file tidak ditemukan maka sistem akan kembali ke opsi koreasi metadata file,
9. tetapi apabila korelasi file-file ditemukan berdasarkan parameternya yang sudah ditentukan
10. akan dilanjutkan ke proses analisis/investigasi file-file yang berkorelasi yang sudah ditemukan tersebut guna untuk pengelompokan barang bukti terkait kasus.
11. Terakhir sistem selesai dan ditutup.

Untuk hasil korelasi metadata file itu sendiri, ada empat jenis yang ditampilkan dari file yaitu Nama sebuah file (File Name),Ukuran dari file (Size),Tanggal suatu file (Date) dan Tempat lokasi atau folder dari suatu file (Path).

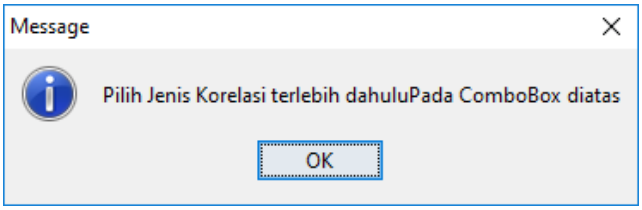
Selanjutnya untuk melihat korelasi metadata, silahkan browse terlebih dahulu lokasinya, apakah mau di data mana (drive C / drive D) atau di folder mana di setiap data (drive C / drive D). Jika belum di browse dan langsung klik button Korelasi File, maka muncul pesan Pilih Lokasi Korelasi terlebih dahulu, hasilnya seperti gambar 4.9 berikut:



Gambar 4.9 pilih lokasi korelasi

Jika sudah di browse lokasi korelasi metadata yang mau di lihat dan langsung klik button Korelasi File, maka muncul pesan Pilih Jenis Korelasi dahulu pada tabel disamping

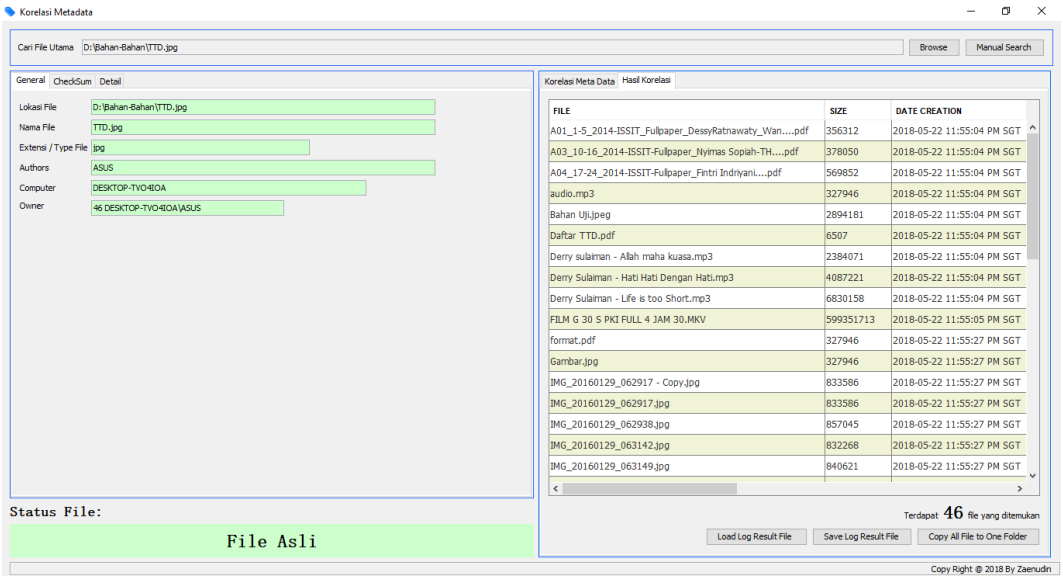
(parameter korelasi yang mau digunakan yaitu bisa Authors, tanggal/date, ukuran/size, ektensi/type atau pemilik/owner), hasilnya seperti gambar 4.10 berikut:



Gambar 4.10 Pilih jenis korelasi/parameter

1. Korelasi Berdasarkan Parameter *Author*

Korelasi berdasarkan parameter Author, yaitu pilih lokasi korelasi terlebih dahulu, misalnya di folder Bahan-bahan di drive D langsung dipilih option *Author*, maka file TTD.jpg yang sudah di browse sebelumnya yang Author filenya bernama ASUS akan segera di cari oleh sistem metadata forensik ini, kemudian menunggu beberapa saat, maka ditemukan file-file dengan Parameter *Author* ASUS yang ada di dalam folder bahan-bahan tadi sebanyak 46 file dan hasilnya dapat dilihat pada gambar 4.11 berikut:

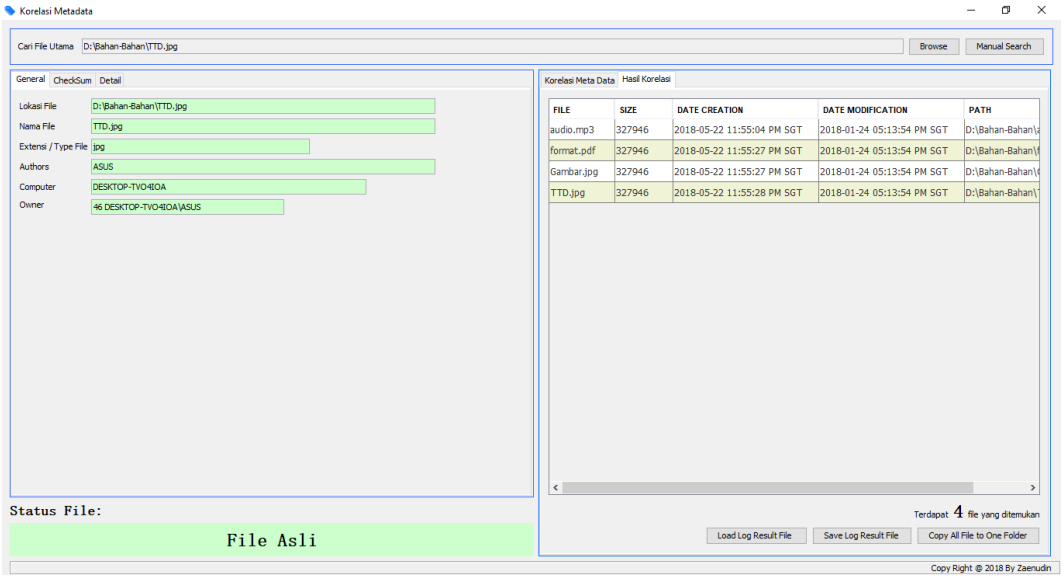


Gambar 4.11 Hasil Korelasi Berdasarkan Parameter *Author*

2. Korelasi Berdasarkan Parameter *Author* dan *Size*

Sama seperti diatas, jika yang di pilih adalah *Author* saja dan sekarang ditambah dengan parameter *size* atau ukuran filenya, dan dipilih option yang dimunculkan adalah sama dengan dari file TTD.jpg yang sudah di browse, baru kemudian di klik button Korelasi File, maka file-file yang ada di folder bahan-bahan di Drive D yang ukurannya sama dengan file TTD.jpg akan segera di cari oleh sistem metadata forensik, maka ditemukan file-file yang

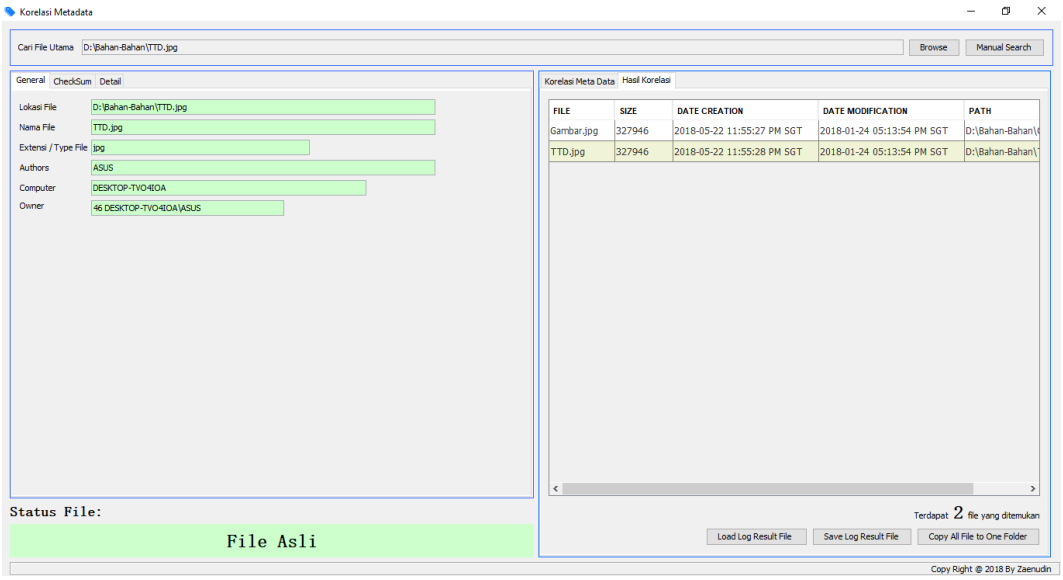
author dan *size* yang sama dengan File TTD.jpg sebanyak 4 file dengan *extensi* file yang berbeda dan hasilnya seperti gambar 4.12 berikut:



Gambar 4.12 Hasil Korelasi Berdasarkan Parameter Author dan Size

3. Korelasi Berdasarkan Parameter *Author, Size* dan *File Type*

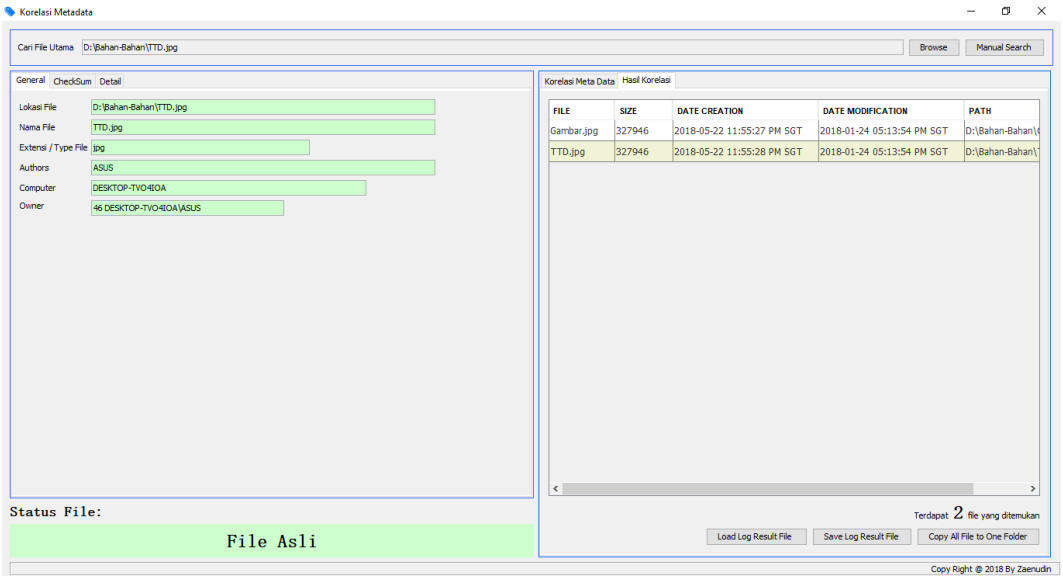
Korelasi berdasarkan parameter *Author, Size* dan *file type* atau *extensi* file dengan mengguanakan file TTD.jpg sebagai bukti awal yang sudah di browse yang berextensi .jpg akan segera di cari oleh sistem metadata forensik, maka ditemukan dua buah file yang sama dengan file TTD.jpg yaitu berektensi jpg dengan *size* yang sama dan *author* yang sama. dalam folder bahan-bahan di Drive D dan hasilnya seperti gambar 4.13 berikut:



Gambar 4.13 Hasil Korelasi Berdasarkan Parameter *Author, Size* dan *File Type*

4. Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

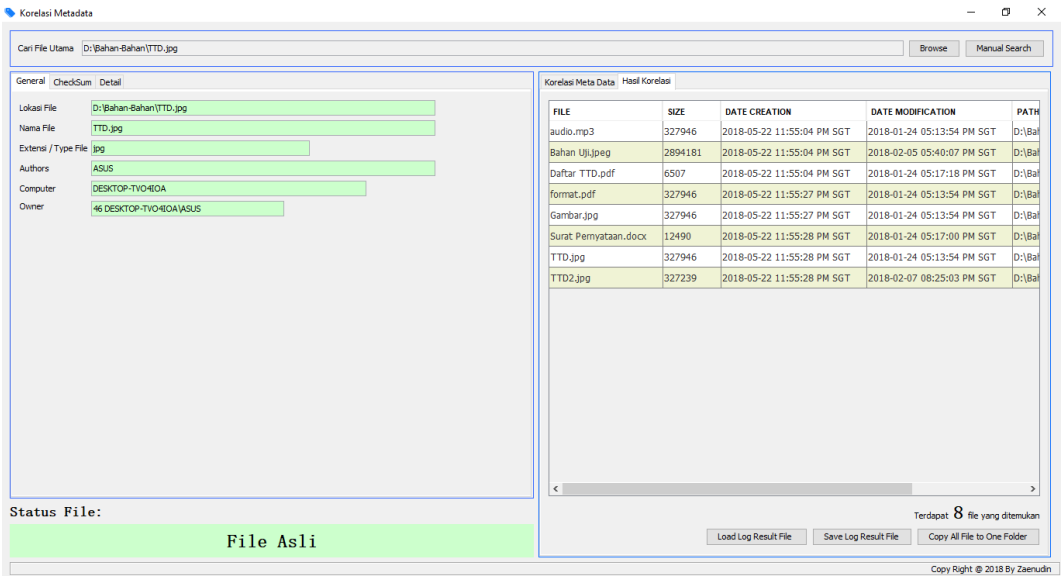
Selanjutnya untuk melihat hasil dari parameter keseluruhan *Author, Size, File Type* dan *Date* dengan memilih Creation Time lebih besar sama dengan tanggal dari file TTD.jpg yang sudah di browse, maka file-file yang ada di folder bahan-bahan yang tanggalnya lebih besar sama dengan file TTD.jpg akan segera di cari oleh sistem metadata forensik, maka ditemukan dua buah file yang *Author, Size, File Type* yang sama dan *Creation Time* dan hasilnya seperti gambar 4.14 berikut:



Gambar 4.14 Hasil Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

5. Korelasi Tanpa Parameter Size dan File Type

Korelasi Tanpa Parameter Size dan File Type yang dimaksud adalah untuk mencari berbagai jenis file dan ukuran sehingga didapatkan hasil pencarian yang bervariasi dengan barang bukti awal file TTD.jpg yang sudah di browse, maka ditemukan delapan file yang berbeda ukuran file dan type file yang ada di dalam folder bahan-bahan di Drive D dan hasilnya seperti gambar 4.15 berikut:



Gambar 4.15 Hasil Korelasi Tanpa Parameter *Size* dan *File Type*

4.6 Hasil Analisis Sistem Metadata Forensik

Berikut adalah hasil pengujian dan analisa hasil sistem yang sudah di ujikan.

4.6.1 Hasil Analisis Membaca Karakteristik Metadata File

1. File gambar ber-Extensi jpg

Nama file yang di uji coba dengan sistem metadata forensik ini TTD.jpg yang berlokasi didalam Folder Bahan-bahan yang ada di Drive D, kemudian dari pengujian tersebut di dapat hasil analisa metadatanya, bisa di lihat di tabel 4.1 berikut:

Tabel 4.1 Hasil membaca metadata file image TTD.jpg

General (a)

No	Jenis Metadata	Value
1	Lokasi file	D:\Bahan-Bahan\TTD.jpg
2	Name File	TTD.jpg
3	Type File	Jpg
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	46 ZEN-ALKARAMI\ASUS

Tabel 4.1 hasil membaca metadata file *image* TTD.jpg

Check Sum (b)

1	MD5	ff989fe3ec144ef1194f878ef9f4057b
2	SHA-256	b75d5a3ccc8e4f4e1e1a3ab8cb16c5b16ed537699eb886880e60fa23fd6ad55f

Tabel 4.1 hasil membaca metadata file *image* TTD.jpg

Detail (c)

No	Jenis Metadata	Value
1	Extention	Jpg
2	Path :	D:\Bahan-Bahan\TTD.jpg
3	creationTime:	2018-01-24 04:13:52 PM ICT
4	lastAccessTime:	2018-02-07T12:41:31.978717Z
5	lastModifiedTime:	2018-01-24 04:13:54 PM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	327946
11	Orientation	Top, left side (Horizontal / normal)
12	X Resolution	2999994/10000 dots per inch
13	Y Resolution	2999994/10000 dots per inch
14	Resolution Unit	Inch
15	Software	Adobe Photoshop CS6 (Windows)
16	Date/Time	2018:01:24 16:13:52
17	Color Space	sRGB
18	Exif Image Width	1556 pixels
19	Exif Image Height	1668 pixels
20	DCT Encode Version	1
21	Flags 0	64
22	Flags 1	0
23	Color Transform	YcbCr
24	XMP Value Count	25
25	Caption Digest	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
26	Print Info 2	[267 bytes]
27	Print Style	[557 bytes]
28	Resolution Info	299.9994x299.9994 DPI
29	Print Scale	Centered, Scale 1.0
30	Global Angle	30
31	Global Altitude	30
32	Print Flags	0 0 0 0 0 0 0 0 1
33	Print Flags Information	0 1 0 0 0 0 0 0 0 1
34	Color Halftoning Information	[72 bytes]
35	Color Transfer Functions	[112 bytes]
36	Layer State Information	0 1
37	Layers Group Information	0 0 0 0
38	Layer Groups Enabled ID	1 1
39	Layer Selection IDs	0 2 0 0 0 4 0 0 0 3
40	Grid and Guides Information	0 0 0 1 0 0 2 64 0 0 2 64 0 0 0 0
41	URL List	0
42	Slices	TTD (0,0,1668,1556) 1 Slices
43	Pixel Aspect Ratio	1.0
44	Seed Number	4
45	Thumbnail Data	JpegRGB, 149x160, Decomp 71680 bytes, 1572865 bpp, 3331 bytes
46	Version Info	1 (Adobe Photoshop, Adobe Photoshop CS6) 1

Tabel 4.1 hasil membaca metadata file *image* TTD.jpg

Detail (c) (Lanjutan)

No	Jenis Metadata	Value
47	JPEG Quality	12 (Maximum), Standard format, 3 scans
48	Compression Type	Baseline
49	Data Precision	8 bits
50	Image Height	1668 pixels
51	Image Width	1556 pixels
52	Number of Components	3
53	Component 1	Y component: Quantization table 0, Sampling factors 1 horiz/1 vert
54	Component 2	Cb component: Quantization table 1, Sampling factors 1 horiz/1 vert
55	Component 3	Cr component: Quantization table 1, Sampling factors 1 horiz/1 vert
56	Thumbnail Compression	JPEG (old-style)
57	X Resolution	72 dots per inch
58	Y Resolution	72 dots per inch
59	Resolution Unit	Inch
60	Thumbnail Offset	302 bytes
61	Thumbnail Length	3331 bytes
62	Profile Size	3144
63	CMM Type	Lino
64	Version	2.1.0
65	Class	Display Device
66	Color space	RGB
67	Profile Connection Space	XYZ
68	Profile Date/Time	Mon Mar 09 13:49:00 ICT 1998
69	Signature	Acsp
70	Primary Platform	Microsoft Corporation
71	Device manufacturer	IEC
72	Device model	sRGB
73	XYZ values	0.9642029 1.0 0.8249054
74	Tag Count	17
75	Copyright	Copyright (c) 1998 Hewlett-Packard Company
76	Profile Description	sRGB IEC61966-2.1
77	Media White Point	(0.9504547, 1.0, 1.0890503)
78	Media Black Point	(0.0, 0.0, 0.0)
79	Red Colorant	(0.43606567, 0.2224884, 0.013916016)
80	Green Colorant	(0.3851471, 0.71687317, 0.097076416)
81	Blue Colorant	(0.1430664, 0.06060791, 0.71409607)
82	Device Mfg Description	IEC **http://www.iec.ch**
83	Device Model Description	IEC 61966-2.1 Default RGB colour space - sRGB
84	Viewing Conditions Description	Reference Viewing Condition in IEC61966-2.1
85	Viewing Conditions	view(0x76696577): 36 bytes
86	Luminance	(76.03647, 80.0, 87.12462)
87	Measurement	1931 2° Observer, Backing (0.0, 0.0, 0.0), Geometry Unknown, Flare 1%, Illuminant D65
88	Technology	CRT

2. File Dokumen ber-Extensi Docx

Nama file yang di uji coba dengan sistem metadata forensik ini adalah file Surat Penyataan.docx yang berlokasi didalam Folder Bahan-bahan yang ada di Drive D, kemudian dari pengujian tersebut di dapat hasil analisa metadatanya, seperti pada tabel 4.2 berikut:

Tabel 4.2 Hasil membaca metadata file Dokumen Surat Pernyataan.docx

General (a)

No	Jenis Metadata	Value
1	Lokasi file	D:\Bahan-Bahan\Surat Pernyataan.docx
2	Name File	Surat Pernyataan.docx
3	Type File	Docx
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	46 ZEN-ALKARAMI\ASUS

Tabel 4.2 hasil membaca metadata file Dokumen Surat Pernyataan.docx

Check Sum (b)

1	MD5	a6785775a2f8d5cc64256c5e2578f1e3
2	SHA-256	241c73488beb611a734e5e9a24a2cf3eaf519 e5c8d75225cbaed024f49c32f34

Tabel 4.2 hasil membaca metadata file Dokumen Surat Pernyataan.docx

Detail (c)

No	Jenis Metadata	Value
1	Extention	Docx
2	Path :	D:\Bahan-Bahan\Surat Pernyataan.docx
3	creationTime:	2018-01-24 04:16:59 PM ICT
4	lastAccessTime:	2018-01-24T09:16:59.903806Z
5	lastModifiedTime:	2018-01-24 04:17:00 PM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	12490

3. File Dokumen ber-Extensi Pdf

Nama file yang di uji coba dengan sistem metadata forensik ini adalah file daftar TTD.pdf yang berlokasi didalam Folder Bahan-bahan yang ada di Drive D, kemudian dari pengujian tersebut di dapat hasil analisa metadatanya, seperti pada tabel 4.3 berikut:

Tabel 4.3 Hasil membaca metadata file Daftar TTD.pdf

General (a)

No	Jenis Metadata	Value
1	Lokasi file	D:\Bahan-Bahan\Daftar TTD.pdf
2	Name File	Daftar TTD.pdf
3	Type File	Pdf
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	46 ZEN-ALKARAMI\ASUS

Tabel 4.3 hasil membaca metadata file Daftar TTD.pdf

Check Sum (b)

1	MD5	481bfb63231f4bd8e33aefcac2e543f3
1	SHA-256	9941f36f17487fe8d3d50fbd62762a7ddfbbe4686e4cfcc5 2a225030643055e0

Tabel 4.3 hasil membaca metadata file Daftar TTD.pdf

Detail (b)

No	Jenis Metadata	Value
1	Extention	Pdf
2	Path :	D:\Bahan-Bahan\Daftar TTD.pdf
3	creationTime:	2018-01-24 04:17:17 PM ICT
4	lastAccessTime:	2018-01-24T09:17:18.541161Z
5	lastModifiedTime:	2018-01-24 04:17:18 PM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	6507

4. File Audio ber-Extensi Mp3

Nama file yang di uji coba dengan sistem metadata forensik ini adalah file Musik.mp3 yang berlokasi didalam Folder Bahan-bahan yang ada di Drive D, kemudian dari pengujian tersebut di dapat hasil analisa metadatanya, seperti pada tabel 4.4 berikut:

Tabel 4.4 Hasil membaca metadata file Musik.mp3

General (a)

No	Jenis Metadata	Value
1	Lokasi file	D:\Bahan-Bahan\Musik.mp3
2	Name File	Musik.mp3
3	Type File	mp3
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	43 ZEN-ALKARAMI\Zen

Tabel 4.4 hasil membaca metadata file Musik.mp3
Chcek Sum (b)

1	Checksum MD5	27f19b73f59a11cfb2771ae5838c2324
2	Checksum SHA-256	548d3db335bbb518fe6de6f218a83bf9c07ab65192210a7 f5e722aab024be1db

Tabel 4.4 hasil membaca metadata file Musik.mp3
Detail (c)

No	Jenis Metadata	Value
1	Extention	mp3
2	Path :	D:\Bahan-Bahan\Musik.mp3
3	creationTime:	2018-01-24 04:19:03 PM ICT
4	lastAccessTime:	2018-01-24T09:19:03.116426Z
5	lastModifiedTime:	2014-09-13 10:54:14 AM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	4029543

5. File Video Ber-Extensi Mp4

Nama file yang di uji coba dengan sistem metadata forensik ini adalah file Video Tutorial.mp4 yang berlokasi didalam Folder Bahan-bahan yang ada di Drive D, kemudian dari pengujian tersebut di dapat hasil analisa metadatanya, seperti pada tabel 4.5 berikut:

Tabel 4.5 Hasil membaca metadata file Video Tutorial.Mp4
General (a)

No	Jenis Metadata	Value
1	Lokasi file	D:\Bahan-Bahan\Video Tutorial.MP4
2	Name File	Video Tutorial.MP4
3	Type File	MP4
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	43 ZEN-ALKARAMI\Zen

Tabel 4.5 hasil membaca metadata file Video Tutorial.Mp4
Check Sum (b)

1	MD5	12a26275091a90ff4878eec10d883340
2	SHA-256	bd2c2373cc881d08da77ed6e7da5b16a38e343eb982 87be8fb2d9f9a4201a25f

Tabel 4.5 hasil membaca metadata file Video Tutorial.Mp4
Detail (c)

No	Jenis Metadata	Value
1	Extention	MP4
2	Path :	D:\Bahan-Bahan\Video Tutorial.MP4
3	creationTime:	2018-01-24 04:18:25 PM ICT
4	lastAccessTime:	2018-01-24T09:18:25.765484Z
5	lastModifiedTime:	2016-12-15 02:32:04 PM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	56651871

4.6.2 Hasil Analisis Korelasi Metadata File

1. Hasil Korelasi dengan Parameter *author*

Hasil Analisis metadata file yang dikorelasi yaitu file TTD.jpg dengna Parameter Author berupa "ASUS", yang dilakukan pencarian file-file yang berlokasi di folder Bahan-bahan pada drive D, maka ditemukan 46 file yang *Author* nya sama dengan file TTD.jpg yang ada di lokasi folder tersebut, tetapi dari sekian bayak file yang sudah ditemukan, diambil sepuluh sampel yang dijadikan sebagai analisa pencarian file berdasarkan korelasi Author. Berikut bisa di lihat hasil analisanya dari tabel 4.6 di bawah ini:

Tabel 4.6 Hasil Korelasi Berdasarkan Parameter *Author*

Nama File	Size	Date Creation	Date Modification	Path
Musik.mp3	4029543	2014-09-13 10:54:14	2018-01-24 04:19:03	D:\Bahan-Bahan\Musik.mp3
studio tv.docx	15593	2015-05-06 11:59:31	2018-01-25 05:45:28	D:\Bahan-Bahan\studio tv.docx
Surat Pernyataan.docx	12490	2015-05-06 11:59:31	2018-01-25 05:45:28	D:\Bahan-Bahan\Surat Pernyataan.docx
TTD.jpg	327946	2018-01-24 04:13:54	2018-01-24 04:13:52	D:\Bahan-Bahan\TTD.jpg
Undangan Nuzulul Quran.doc	26624	2017-06-16 01:43:35	2018-01-25 05:45:28	D:\Bahan-Bahan\Undangan Nuzulul Quran.doc
Video Tutorial.MP4	56651871	2016-12-15 02:32:04	2018-01-24 04:18:25	D:\Bahan-Bahan\Video Tutorial.MP4
مطلب الطلاب.docx	84421	2014-06-22 08:31:56	2018-01-25 05:45:28	D:\Bahan-Bahan\ مطلب الطلاب.docx

Tabel 4.6 Hasil Korelasi Berdasarkan Parameter *Author* (Lanjutan)

Nama File	Size	Date Creation	Date Modification	Path
Daftar TTD.pdf	6507	2018-01-24 04:17:18	2018-01-24 04:17:17	D:\Bahan-Bahan\Daftar TTD.pdf
FILM G 30 S PKI FULL 4 JAM 30.MKV	599351713	2017-10-01 07:40:47	2017-10-01 07:37:16	D:\Bahan-Bahan\FILM G 30 S PKI FULL 4 JAM 30.MKV
IMG_20160129_062917.jpg	833586	2016-03-16 03:26:38	2018-01-24 05:00:59	D:\Bahan-Bahan\IMG_20160129_062917.jpg

2. Hasil Korelasi dengan Parameter *Author* dan *Size*

Hasil Analisis metadata file yang dikorelasi yaitu file TTD.jpg metadata yang *Author* nya "ASUS" dan akan menambah korelasi dengan ukuran file TTD.jpg yang besar filenya "327946 byte", dilakukan pencarian file-file yang berlokasi di Folder Bahan-bahan di drive D dengan memilih salah satu pilihan yakni "Sama Dengan", maka ditemukan 4 file yang *author* nya "ASUS" dan ukuran filenya sama dengan "327946 byte" byte dari metadata *Size* file TTD.jpg yang ada di folder tersebut. Berikut dapat dilihat hasil analisanya dari tabel 4.7 di bawah ini:

Tabel 4.7 Hasil Korelasi Berdasarkan Parameter *Author* dan *Size*

Nama File	Size	Date Creation	Date Modification	Path
audio.mp3	327946	2018-01-24 04:13:54	2018-01-25 07:03:23	D:\Bahan-Bahan\audio.mp3
format.pdf	327946	2018-01-24 04:13:54	2018-01-25 07:03:23	D:\Bahan-Bahan\format.pdf
gambar.jpg	327946	2018-01-24 04:13:54	2018-01-25 10:51:09	D:\Bahan-Bahan\gambar.jpg
TTD.jpg	327946	2018-01-24 04:13:52	2018-01-24 04:13:54	D:\Bahan-Bahan\TTD.jpg

3. Hasil Korelasi dengan Parameter *Author, Size* dan *File Type*

Untuk analisis hasil metadata file yang dikorelasi yaitu file TTD.jpg yang metadata *Author* nya bernama "ASUS" dan ukuran filenya sebesar "327946 byte" dan akan menambah korelasi dengan ektensi filenya berupa "jpg", yang dilakukan pencarian file-file yang berlokasi di folder bahan-banah di drive D. Maka ditemukan 2 buah file yang Authornya "ASUS", Ukuran filenya "327946 byte" dan ektensi filenya jpg dari metadata file TTD.jpg yang ada di lokasi tersebut. Berikut bisa di lihat hasil analisanya dari tabel 4.8 di bawah ini:

Tabel 4.8 Hasil Korelasi Berdasarkan Parameter *Author, Size* dan *File Type*

Nama File	Size	Date Creation	Date Modificaton	Path
gambar.jpg	327946	2018-01-24 04:13:54	2018-01-25 10:51:09	D:\Bahan-Bahan\gambar.jpg
TTD.jpg	327946	2018-01-24 04:13:52	2018-01-24 04:13:54	D:\Bahan-Bahan\TTD.jpg

4. Hasil Korelasi dengan Parameter *Author, Size, File Type* dan *Date*

Hasil Analisis metadata file yang dikorelasi yaitu file TTD.jpg yang metadata Authornya "ASUS", Ukuran Filenya "327946 byte", type filenya Jpg dan akan menambahkan korelasi dengan tanggal pada file TTD.jpg yaitu "24 Januari 2018", dilakukan pencarian file-file yang berlokasi di folder bahan-bahan dengan pilihan "lebih besar sama dengan", maka ditemukan 2 buah file yang *Authornya* "ASUS", Ukuran filenya "327946 byte", Extensi filenya ".Jpg" dan tanggalnya sama dengan "24 Januari 2018" dari metadata tanggal file TTD.jpg yang ada dilokasi tersebut. Berikut bisa di lihat hasil analisanya dari tabel 4.9 di bawah ini:

Tabel 4.9 Hasil Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

Nama File	Size	Date Creation	Date Modificaton	Path
gambar.jpg	327946	2018-01-24 04:13:54	2018-01-25 10:51:09	D:\Bahan-Bahan\gambar.jpg
TTD.jpg	327946	2018-01-24 04:13:52	2018-01-24 04:13:54	D:\Bahan-Bahan\TTD.jpg

5. Hasil Korelasi Tanpa Paramevter Size dan File Type

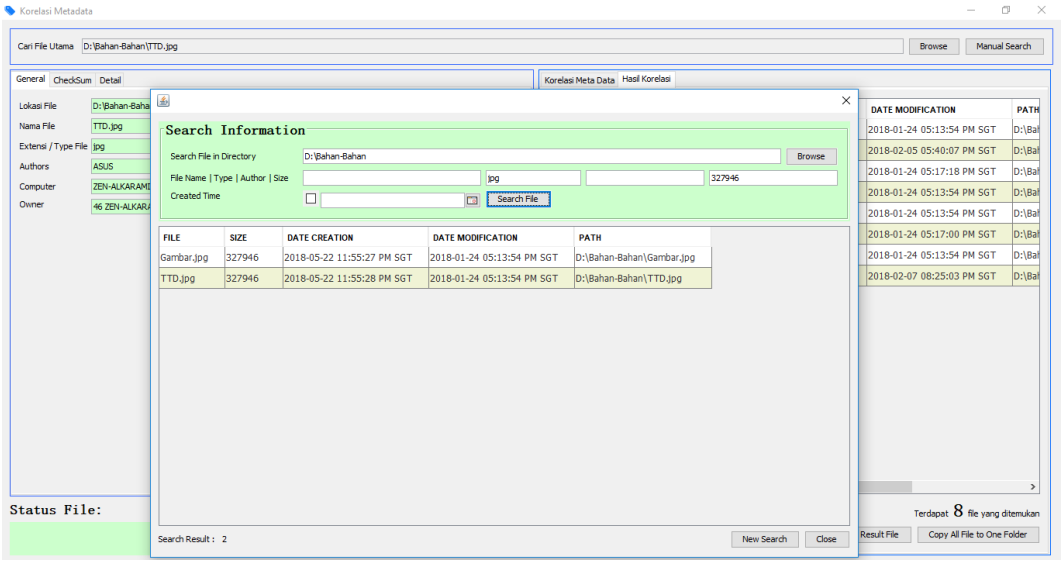
Hasil Analisis Korelasi Tanpa Parameter Size dan File Type yang dimaksud adalah untuk mencari berbagai jenis file dan ukuran sehingga didapatkan hasil pencarian yang bervariasi atau lebih banyak dengan barang bukti awal file TTD.jpg. maka didapatkan delapan file hasil analisis yang metadata Authornya "ASUS", tanggalnya "24-Januari-2018" dengan file type berupa "Mp3, Pdf, Jpg dan Docx" dan ukuran file yang berbeda-beda dan hasilnya seperti tabel 4.10 berikut:

Tabel 4.10 Hasil Korelasi Tanpa Parameter *Size* dan File *Type*

Nama File	Size	Date Creation	Date Modificaton	Path
audio.mp3	327946	2018-01-24 04:13:54	2018-01-25 07:03:23	D:\Bahan-Bahan\audio.mp3
Bahan Uji.jpeg	2894181	2018-05-22 11:55:04	2018-02-05 05:40:07	D:\Bahan-Bahan\Bahan Uji.jpeg
Daftar TTD.pdf	6507	2018-01-24 04:17:18	2018-01-24 04:17:17	D:\Bahan-Bahan\Daftar TTD.pdf
format.pdf	327946	2018-01-24 04:13:54	2018-01-25 07:03:23	D:\Bahan-Bahan\format.pdf
Gambar.jpg	327946	2018-01-24 04:13:54	2018-01-25 10:51:09	D:\Bahan-Bahan\Gambar.jpg
Surat Pernyataan.docx	12490	2018-01-24 04:17:00	2018-01-24 04:16:59	D:\Bahan-Bahan\Surat Pernyataan.docx
TTD.jpg	327946	2018-01-24 04:13:52	2018-01-24 04:13:54	D:\Bahan-Bahan\TTD.jpg
TTD2.jpg	327239	2018-05-22 11:55:28	2018-02-07 08:25:03	D:\Bahan-Bahan\TTD2.jpg

6. Fitur Tambahan

Setelah file-file hasil korelasi dengan parameter keseluruah di temukan untuk memudahkan investigator dalam menganalisa file-file tersebut maka dibuatkanlah fitur tambahan seperti Manual search, Status file (apakah file sudah di modifikasi atau file masih asli berdasarkan *timestamp* yaitu *Creation Time* dan *Modification time*), *copy all file to one folder* agar bisa memindahkan file hasil korelasi yang berbeda folder tersebut ke satu folder yang di inginkan. Dan tombol klik kanan pada hasil korelasi tersebut untuk *open file* dan *open location file* seperti diterangkan pada gambar 4.16 dibawah ini



Gambar 4.16 Keterangna Fitur Tambahan

7. Perbandingan Membaca Metadata file

pada bagian ini peneliti melakukan perbandingan dengan penelitian sebelumnya yaitu (Subli, Sugiantoro & Prayudi, 2017) dalam penelitiannya Metadata forensik untuk mendukung proses investigasi dengan membangun sebuah sistem untuk membaca metadata dan metadata extraktor dengan penelitian yang diusulkan. Dapat ditarik kesimpulan perbedaan pembacaan hasil metadatanya. Berikut dapat dilihat pada tabel 4.11 perbandingan hasil pemeriksaan metadata file gambar Bahan Uji.jpg dari kedua tools tersebut untuk lengkapnya ada pada lampiran.

Tabel 4.11 Perbandingan Membaca Metadata *file Type* Jpg

Jenis Metadata	**Metadata Extractor**	**Metadata Forensik**	**Metadata Forensik yang di Usulkan**
Lokasi file	D:\ProsesImaging\ Paket 001.jpg	D:\ProsesImaging\Paket 001.jpg	D:\Bahan-Bahan\TTD.jpg
Name File	Paket 001.jpg	Paket 001.jpg	TTD.jpg
Type File	jpg	jpg	Jpg
Author			ASUS
Computer	-	DESKTOP-H1C8GI7	ZEN-ALKARAMI
Owner	-	Amikom	46 ZEN-ALKARAMI\ASUS
Extention	-	-	Jpg
Path	-	-	D:\Bahan-Bahan\TTD.jpg
creationTime	2017030, 011144086	2017-03-29T16:47:39.155738Z	2018-01-24 04:13:52 PM ICT
lastAccessTime	-	2017-03-29T16:47:39.155738Z	2018-02-07T12:41:31.978717Z
lastModifiedTime	20140104, 122644000	2014-01-04T04:26:44Z	2018-01-24 04:13:54 PM ICT
isRegularFile	-	True	True
isSymbolicLink	-	False	False
size	2152995	2152995	327946
Orientation	-	-	Top, left side (Horizontal / normal)
Resolution Unit	-	-	Inch
Software	-	-	Adobe Photoshop CS6 (Windows)

4.7 Studi Kasus

Pada tanggal 10 januari 2018 pukul 10.00 WIB Komisi Pemberantasan Korupsi menetapkan seorang pejabat negara berinisial BI sebagai tersangka terkait kasus dugaan korupsi.

Penetapan tersangka dilakukan setelah penyidik melakukan penggeledahan dan penyegelan di rumahnya. Dalam penggeledahan itu penyidik berhasil menemukan dua alat bukti, 1 (satu) alat bukti berupa berkas dokumen dan 1 (satu) alat bukti elektronik berupa komputer terkait dengan dugaan korupsi tersebut.

1. H+1 Proses Penanganan Barang Bukti

Dari barang bukti yang disita, oleh badan investigator melakukan penanganan terhadap barang bukti dan mendapatkan beberapa petunjuk:

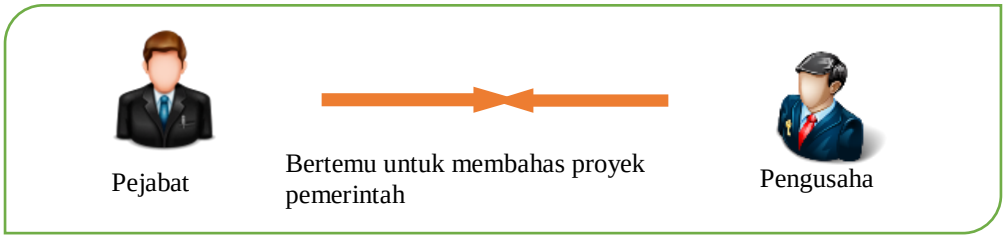
- Komputer yang digunakan tersangka untuk menyimpan data-data dan untuk di analisis lebih lanjut.
- Berkars-berkas dokumen yang terkait bukti dugaan korupsi

2. Ringkasan Skenario Kasus

Topik	: Korupsi
Petugas Penyidik	: Divisi Cybercrime (Investigator)
Objek penyelidikan	: inisial BI
Pihak yang terlibat	: 1. inisial BI
	: 2. inisial Pengusaha
Alat Bukti Elektronik	: 1 Unit Komputer,
Alat Bukti Digital	: file Bukti Utama photo.jpg, kemudian mencari file-file terkait dengan kasus

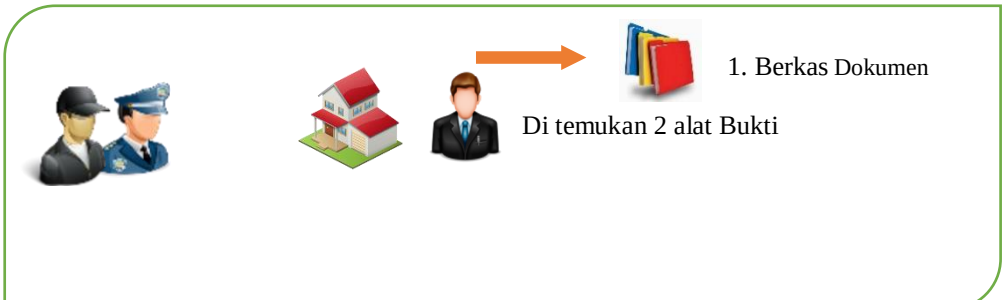
3. Ilustrasi Alur Kasus

Pra Insiden



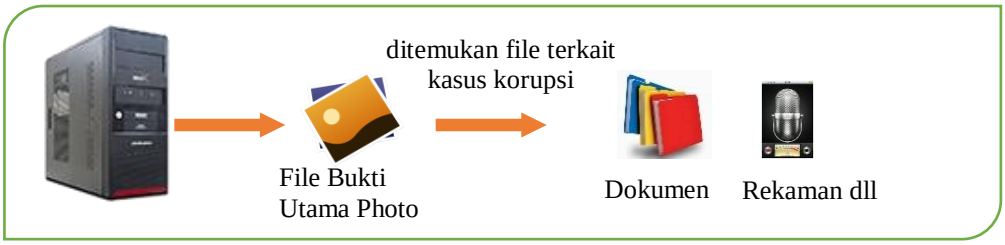
Gambar 4.17 Pra Insiden

Insiden



Gambar 4.18 Insiden

Past Insiden



Gambar 4.19 Past Insiden

4. Maksud dan Tujuan Pemeriksaan Barang Bukti

Adapun maksud dan tujuan dilakukannya pemeriksaan terhadap barang bukti dan dilakukannya investigasi yaitu :

a. Mencari Apa isi berkas-berkas dokumen cetak
b. Menganalisis Informasi photo untuk mencari file-file yang terkait kasus

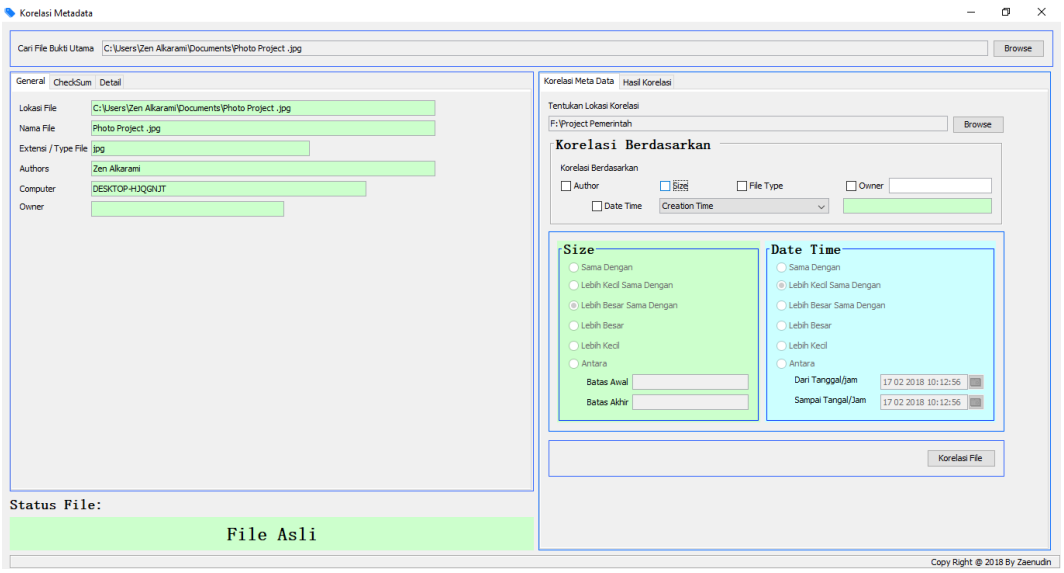
5. Akuisisi dan analisis barang bukti

Disini penyidik akan melakukan akuisisi terhadap media penyimpanan dari komputer tersebut, media penyimpanan disini maksudnya adalah hardisk. Untuk melakukan akuisisi penyidik menggunakan program *encase,* adapun langkah-langkah akuisisi hardisk tersebut ada pada lampiran:

Selanjutnya penyidik akan melakukan analisis terhadap barang bukti yang sudah di akuisisi tadi dengan melakukan pendekatan metadata menggunakan metadata forensik yang sudah dibangun guna untuk membaca metadata dan mencari korelasi antara file yang terkait dengan file tersebut antara sebagai berikut:

A. Membaca Metadata File Bukti Utama

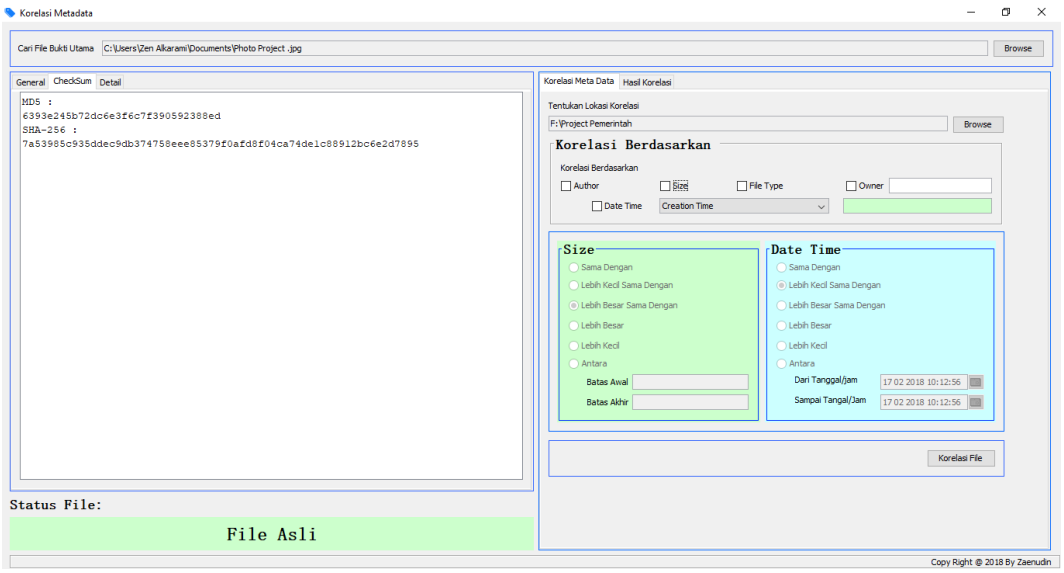
File Bukti Utama dengan nama “Photo Project.Jpg” yang ditemukan di My Document akan dibaca metadatanya menggunakan metadata forensik dengan cara browse dimana lokasi file tersebut, file teridentifikasi metadatanya, kemudian akan tampil keterangan metadatanya di tabel *General, Checksum dan Detail*, seperti gambar 4.20, Gambar 4.21 dan Gambar 4.22 berikut:



Gambar 4.20 Menampilkan metadata *general* file Photo Project.jpg

Tabel 4.12 Hasil membaca metadata general file Photo Project.jpg

No	Jenis Metadata	Value
1	Lokasi file	C:\Users\ASUS\Documents\Photo Project .jpg
2	Name File	Photo Project .jpg
3	Type File	Jpg
4	Author	ASUS
5	Computer	ZEN-ALKARAMI
6	Owner	

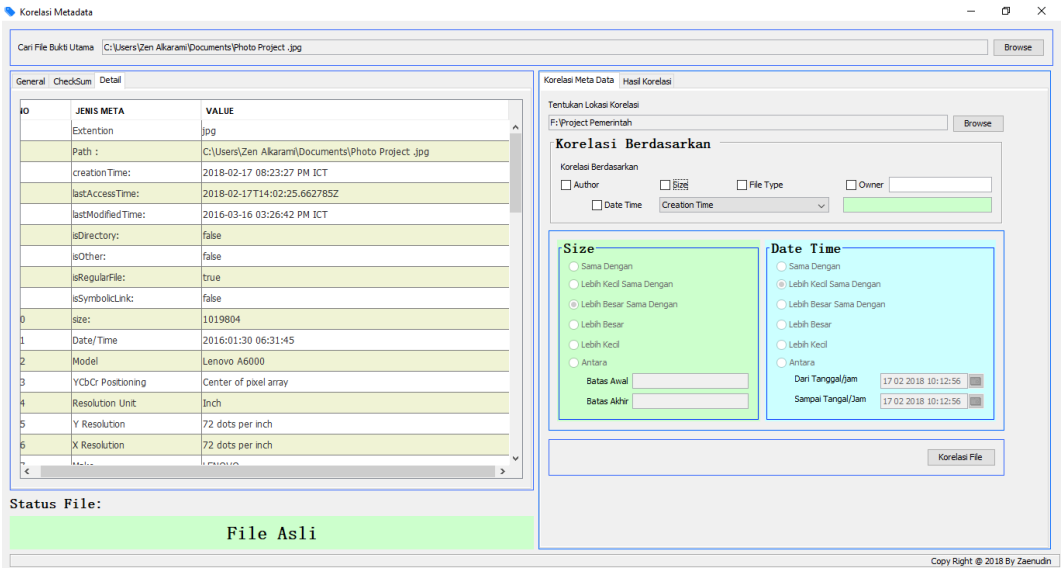


Gambar 4.21 Menampilkan metadata *Checksum* file Photo Project.jpg

Tabel 4.13 Hasil membaca metadata *checksum* file Photo Project.jpg

1	MD5	6393e245b72dc6e3f6c7f390592388ed
2	SHA-256	7a53985c935ddec9db374758eee85379f0afd8

		f04ca74de1c88912bc6e2d7895



Gambar 4.22 Metadata *Detail* file Photo Project.jpg

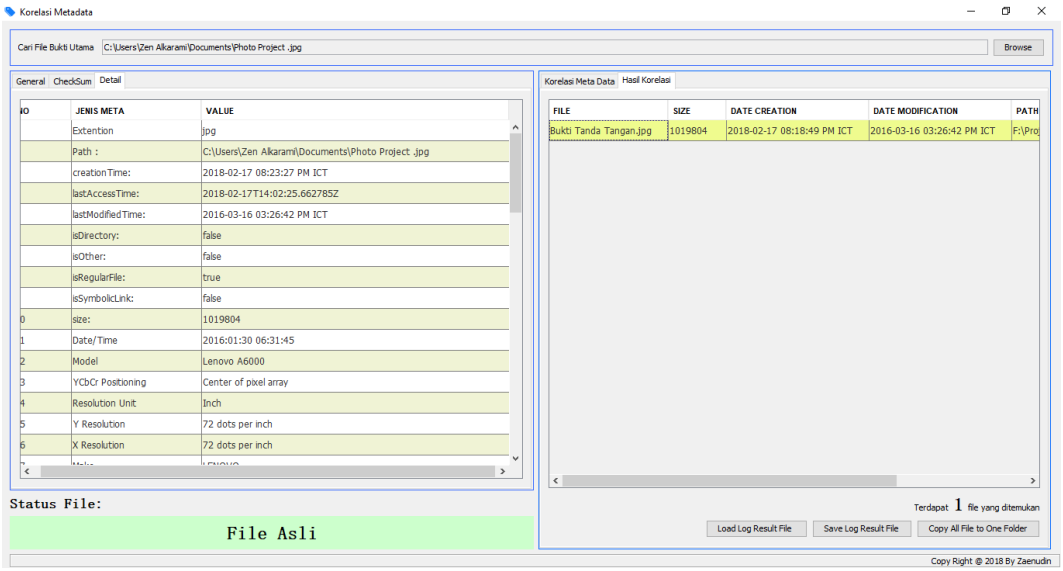
Tabel 4.14 Hasil membaca metadata *Detail* file Photo Project.jpg di tampilkan sebagian

No	Jenis Metadata	Value
1	Extention	Jpg
2	Path :	C:\Users\ASUS\Documents\Photo Project .jpg
3	creationTime:	2018-02-17 08:23:27 PM ICT
4	lastAccessTime:	2018-02-17T14:02:25.662785Z
5	lastModifiedTime:	2016-03-16 03:26:42 PM ICT
6	isDirectory:	False
7	isOther:	False
8	isRegularFile:	True
9	isSymbolicLink:	False
10	size:	1019804
11	Date/Time	2016:01:30 06:31:45
12	Model	Lenovo A6000
13	Y Resolution	72 dots per inch
14	Resolution Unit	Inch
15	Make	LENOVO
16	Focal Length	2.93 mm
17	Image Width	1600 pixels
18	Image Height	1200 pixels
19	Exif Image Height	1200 pixels
20	GPS Date Stamp	2016:01:29
21	GPS Img Direction Ref	Magnetic direction
22	GPS Time-Stamp	22:31:45.00 UTC
23	Data Precision	8 bits

B. Anaslisis Korelasi Metadata file

1. Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

Setelah membaca metadata file dari "Photo Project.jpg" selanjutnya melakukan analisis korelasi metadata dengan parameter keseluruhan *Author, Size, File Type* dan *Date* dengan memilih Creation Time sama dengan tanggal dari file Photo Project.jpg yang sudah di browse yang berada di My Document, selanjutnya dilakukan korelasi di drive F. Maka ditemukan 1 (satu) file yang *Author, Size, File Type* yang sama dan *Creation Time* yang sama dengan dan hasilnya seperti gambar 4.23 berikut:



Gambar 4.23 Hasil Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

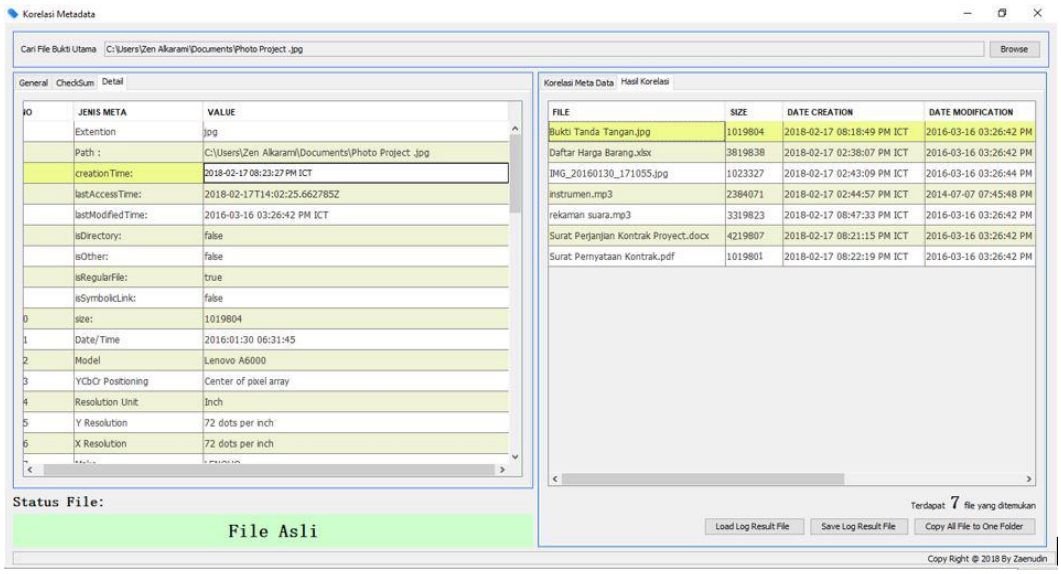
Hasil Analisis metadata file yang dikorelasi yaitu file "Photo Project.jpg" yang metadata Authornya "ASUS", Ukuran Filenya "1019804 *byte*", type filenya Jpg dan tanggal pada file Photo Project.jpg yaitu "2018-02-17", dilakukan pencarian file-file yang berlokasi di drive F", maka ditemukan 1 (satu) file yang *Authornya* "ASUS", Ukuran filenya "1019804 byte", Extensi filenya ".Jpg" dan tanggalnya sama dengan "2018-02-17" dari metadata tanggal file Photo Project.jpg yang ada dilokasi tersebut. Berikut bisa di lihat hasil analisanya dari tabel 4.15 di bawah ini:

Tabel 4.15 Hasil Korelasi Berdasarkan Parameter *Author, Size, File Type* dan *Date*

Nama File	Size	Date Creation	Date Modificaton	Path
Bukti Tanda Tangan.jpg	1019804	2018-02-17 08:18:49	2016-03-16 03:26:42	F:\Project Pemerintah\Bukti Tanda Tangan.jpg

2. Korelasi Tanpa Parameter Size dan File Type

Korelasi Tanpa Parameter Size dan File Type yang dimaksud adalah untuk mencari berbagai jenis file dan ukuran sehingga didapatkan hasil pencarian yang bervariasi dengan barang bukti utama file Photo Project.jpg yang sudah di browse, maka ditemukan 7 (tujuh) file yang berbeda ukuran file dan type file yang ada di dalam Drive F dan hasilnya seperti gambar 4.24 berikut:



Gambar 4.24 Hasil Korelasi Tanpa Parameter *Size* dan *File Type*

Hasil Analisis Korelasi Tanpa Parameter Size dan File Type yang dimaksud adalah untuk mencari berbagai jenis file dan ukuran sehingga didapatkan hasil pencarian yang bervariasi atau lebih banyak dengan barang bukti utama Photo Project.jpg maka didapatkan tujuh file hasil analisis yang metadata *Authornya* "ASUS", tanggalnya "2018-02-17" dengan file type berupa "Mp3, Pdf, Jpg, xlsx dan Docx" dan ukuran file yang berbeda-beda dan hasilnya seperti tabel 4.16 berikut:

Tabel 4.16 Hasil Korelasi Tanpa Parameter *Size* dan File *Type*

Nama File	Size	Date Creation	Date Modificaton	Path
Bukti Tanda Tangan.jpg	1019804	2018-02-17 08:18:49	2016-03-16 03:26:42	F:\Project Pemerintah\Bukti Tanda Tangan.jpg
Daftar Harga Barang.xlsx	3819838	2018-02-17 08:18:49	2016-03-16 03:26:42	F:\Project Pemerintah\Daftar Harga Barang.xlsx

Tabel 4.16 Hasil Korelasi Tanpa Parameter *Size* dan File *Type* (Lanjutan)

Nama File	Size	Date Creation	Date Modificaton	Path
IMG_20160130_171055.jpg	1023327	2018-02-17 08:18:49	2016-03-16 03:26:42	F:\Project Pemerintah\IMG_20160130_171055.jpg
instrumen.mp3	2384071	2018-01-24 04:13:54	2014-07-07 07:45:48	F:\Project Pemerintah\instrumen.mp3
rekaman suara.mp3	3319823	2018-01-24 04:17:00	2016-03-16 03:26:42	F:\Project Pemerintah\rekaman suara.mp3
Surat Perjanjian Kontrak Proyect.docx	4219807	2018-01-24 04:13:52	2016-03-16 03:26:42	F:\Project Pemerintah\Surat Perjanjian Kontrak Proyect.docx
Surat Pernyataan Kontrak.pdf	1019801	2018-01-24 04:13:52	2016-03-16 03:26:42	F:\Project Pemerintah\Surat Pernyataan Kontrak.pdf

6. Kesimpulan dari kasus

Kesimpulan dari penangan kasus korupsi ini, setelah didapatkan bukti utama yang berada pada my document maka dilakukan penelusuran lebih lanjut untuk menemukan barang bukti lainya dengan menggunakan pendekatan berbasis metadata. Setelah di analisis maka ditemukan korelasi antara file bukti utama dengan file-file yang berada pada drive F. adapun hasil yang didapatkan dengan menggunakan parameter keseluruhan (*author, size, file type dan date*) ditemukan satu file yang terkorelasi dengan barang bukti utama dan dengan menggunakan tanpa parameter *size* dan *file type* maka ditemukan tujuh file yang terkorelasi dengan file bukti utama tersebut.

Hasil percobaan ini juga membuktikan bahwa baik dengan menggunakan seluruh parameter maupun sebagian parameter menggunakan tools metadata forensik yang dibangun berhasil mendapatkan korelasi barang bukti digital dengan lebih mudah dan waktu yang efesien. Sedangkan mencari korelasi file bukti digital dengan manual tentu akan membutuhkan waktu lebih lama.

Dengan menggunakan tools metadata forensik yang dibangun tentu akan sangat membantu dan mempermudah seorang investigator dalam menganalisa korelasi barang bukti digital tersebut. Namun tools metadata forensik yang dibangun ini juga memiliki banyak keterbatasan belum bisa digunakan secara online dan belum bisa mencari dengan multi drive atau multi local.

4.8 Karakteristik Tools Metadata Forensik

Selama ini investigator analisis forensik dalam penanganan barang bukti utama dengan pendekatan berbasis metadata masih banyak secara manual dalam mencari korelasi file yang terkait. Ketika file yang dikorelasikan berada dilokasi (folder) yang terpisah dan banyaknya file tentu akan menjadi tantangan yang berat bagi para invesitigator forensik dalam menganalisis barang bukti digital tersebut (Raghavan & Raghavan, 2014a).

Dengan menggunakan tools metadata forensik yang dibangun, dapat mempermudah dalam analisis korelasi barang bukti digital tersebut. Tools metadata forensik yang dibangun memiliki false positif dan false negatif diantaranya.

a. False Positive

- Pengertian *False Positive*

False positive, diartikan bebas sebagai "positif palsu", biasanya disebabkan algoritma suatu program yang menyatakan adanya suatu gejala/sinyal/objek yang sebetulnya tidak ada. Disebut juga dengan *false alarm.*

b. False Negative

- Pengertian *False Negative*

False Negative adalah di mana hasil tes negatif salah. Dengan kata lain, mendapatkan hasil tes negatif, tetapi seharusnya mendapatkan hasil tes positif. Negatif palsu menciptakan dua masalah. Yang pertama adalah rasa aman yang salah. Misalnya, jika lini produksi Anda tidak menangkap item yang rusak, Anda mungkin berpikir proses berjalan lebih efektif daripada yang sebenarnya. Masalah kedua, yang berpotensi lebih serius, adalah bahwa situasi yang berpotensi berbahaya mungkin terlewatkan. Sebagai contoh, virus komputer yang melumpuhkan dapat mendatangkan malapetaka jika tidak terdeteksi

Setelah dilakukan pengecekan pada tools metadata forensik yang dibangun bahwa *false fositive* dan *false negatif* tidak terdeteksi atau tidak ditemukan.

BAB 5
Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil analisis dalam penelitian dengan judul "Metadata Forensik untuk analisis Korelasi Bukti Digital" dapat di tarik beberapa kesimpulan sebagai berikut:

1. Karakteristik metadata file dapat dipahami secara umum yang dibagi menjadi tiga bagian utama yaitu Metadata *General*, Metadata *Checksum* dan Metadata *detail*.
2. Korelasi metadata antara file bukti digital ditentukan berdasarkan parameter yaitu *Author, Size, File Type* dan *Date*
3. Pengujian kinerja sistem analisis korelasi metadata menggunakan tools metadata forensik yang dibangun dengan parameter *Author, Size, File Type* dan *Date* dengan *sample* file berekstensi *doc, pdf, jpg, mp3 dan mp4.*

5.2 Saran

Adapun saran-saran yang perlu di berikan untuk penelitian ini adalah sebagai berikut:

1. Pada penelitian selanjutnya perlu dilakukan analisis korelasi tidak hanya dengan parameter metadata tersebut.
2. Pengembangan dan penelitian lebih lanjut perlu ditambahkan pilihan multi Local/multi drive pada browse barang bukti utama agar bisa membrowse 2 local atau lebih sekaligus.
3. Dalam penelitian berikutnya perlu di kembangkan lagi detail metadata yang lebih spesifik selain type file jpg.

Daftar Pustaka

Alanazi, F., & Jones, A. (2016). The Value of Metadata in Digital Forensics. *Proceedings - 2015 European Intelligence and Security Informatics Conference, EISIC 2015*, *8*(2011), 182. https://doi.org/10.1109/EISIC.2015.26

Crossley, R., Asimakopoulou, E., Sotiriadis, S., & Bessis, N. (2013). A study on metadata tagging for tracking original file information within the cloud. *Proceedings - 2013 8th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC 2013*, 453–456. https://doi.org/10.1109/3PGCIC.2013.76

Drive, L., Hall, M., Hill, C., Woods, K., Chassanoff, A., & Lee, C. a. (2013). Managing and Transforming Digital Forensics Metadata for Digital Collections. *The 10th International Conference on Preservation of Digital Objects*, (November), 203–208. Retrieved from http://purl.pt/24107/1/iPres2013_PDF/Managing and Transforming Digital Forensics Metadata for Digital Collections.pdf

Kumar, P. R., Srikanth, C., & Sailaja, K. L. (2016). Location Identification of the Individual based on Image Metadata. *Procedia Computer Science*, *85*(Cms), 451–454. https://doi.org/10.1016/j.procs.2016.05.191

Prayudi, Y (2014). Problema Dan Solusi Digital Chain Of Custody Dalam Proses Investigasi, (April).

Peterson, A. (2016). Report Information from ProQuest. *Washington Post*, (May). https://doi.org/http://dx.doi.org/10.1108/17506200710779521

Raghavan, S., & Raghavan, S. V. (2014a). AssocGEN: Engine for analyzing metadata based associations in digital evidence. *Int. Workshop Syst. Approaches Digit. Forensics Eng., SADFE*. https://doi.org/10.1109/SADFE.2013.6911541

Raghavan, S., & Raghavan, S. V. (2014b). Eliciting file relationships using metadata based associations for digital forensics. *Int. Workshop Syst. Approaches Digit. Forensics Eng., SADFE*, *2*(March), 49–64. https://doi.org/10.1109/SADFE.2013.6911541

Raghavan, S., & Raghavan, S. V. (2013). A study of forensic & analysis tools. *2013 8th International Workshop on Systematic Approaches to Digital Forensics Engineering (SADFE)*, 1–5. https://doi.org/10.1109/SADFE.2013.6911540

Riley, J. (2017). *Understanding Metadata: What Is Metadata, and What is it for? NISO Primer*. https://doi.org/10.1017/S0003055403000534

Salama, U., Varadharajan, V., Hitchens, M., & DUMMY. (2012). Metadata Based Forensic Analysis of Digital Information in the Web. *Annual Symposium on Information Assurance & Secure Knowledge Management*, 9–15.

Spore, A. (2016). Report Information from ProQuest, (June).

Subli, Sugiantoro & Prayudi. (2017). "Metadata Forensik Untuk Mendukung Proses Investigasi". Jurnal Ilmiah DASI.

Lampiran

A. Perbandingan Membaca Metadata File

Tabel 4.11 Perbandingan Membaca Metadata *file Type* Jpg

Jenis Metadata	Metadata Extractor	Metadata Forensik	Metadata Forensik yang di Usulkan
Lokasi file	D:\ProsesImaging \Paket 001.jpg	D:\ProsesImaging \Paket 001.jpg	D:\Bahan-Bahan\TTD.jpg
Name File	Paket 001.jpg	Paket 001.jpg	TTD.jpg
Type File	jpg	Jpg	Jpg
Author			ASUS
Computer	-	DESKTOP-H1C8GI7	ZEN-ALKARAMI
Owner	-	Amikom	46 ZEN-ALKARAMI\ASUS
Extention	-	-	Jpg
Path	-	-	D:\Bahan-Bahan\TTD.jpg
creationTime	2017030, 011144086	2017-03-29T16:47:39.155738Z	2018-01-24 04:13:52 PM ICT
lastAccessTime	-	2017-03-29T16:47:39.155738Z	2018-02-07T12:41:31.978717Z
lastModifiedTime	20140104, 122644000	2014-01-04T04:26:44Z	2018-01-24 04:13:54 PM ICT
isDirectory	-	False	False
isOther	-	False	False
isRegularFile	-	True	True
isSymbolicLink	-	False	False
Size	2152995	2152995	327946
Orientation	-	-	Top, left side (Horizontal / normal)
X Resolution	-	-	2999994/10000 dots per inch
Y Resolution	-	-	2999994/10000 dots per inch
Resolution Unit	-	-	Inch
Software	-	-	Adobe Photoshop CS6 (Windows)
Date/Time	-	-	2018:01:24 16:13:52
Color Space	-	-	sRGB
Exif Image Width	-	-	1556 pixels
Exif Image Height	-	-	1668 pixels
DCT Encode Version	-	-	1
Flags 0	-	-	64
Flags 1	-	-	0
Color Transform	-	-	YcbCr
XMP Value Count	-	-	25
Caption Digest	-	-	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Print Info 2	-	-	[267 bytes]
Print Style	-	-	[557 bytes]

Tabel 4.11 Perbandingan Membaca Metadata file Type Jpg (Lanjutan)

Jenis Metadata	Metadata Extractor	Metadata Forensik	Metadata Forensik yang di Usulkan
Resolution Info	-	-	299.9994x299.9994 DPI
Print Scale	-	-	Centered, Scale 1.0
Global Angle	-	-	30
Global Altitude	-	-	30
Print Flags	-	-	0 0 0 0 0 0 0 0 1
Print Flags Information	-	-	0 1 0 0 0 0 0 0 0 1
Color Halftoning Information	-	-	[72 bytes]
Color Transfer Functions	-	-	[112 bytes]
Layer State Information	-	-	0 1
Layers Group Information	-	-	0 0 0 0
Layer Groups Enabled ID	-	-	1 1
Layer Selection IDs	-	-	0 2 0 0 0 4 0 0 0 3
Grid and Guides Information	-	-	0 0 0 1 0 0 2 64 0 0 2 64 0 0 0 0
URL List	-	-	0
Slices	-	-	TTD (0,0,1668,1556) 1 Slices
Pixel Aspect Ratio	-	-	1.0
Seed Number	-	-	4
Thumbnail Data	-	-	JpegRGB, 149x160, Decomp 71680 bytes, 1572865 bpp, 3331 bytes
Version Info	-	-	1 (Adobe Photoshop, Adobe Photoshop CS6) 1
JPEG Quality	-	-	12 (Maximum), Standard format, 3 scans
Compression Type	-	-	Baseline
Data Precision	-	-	8 bits
Image Height	-	-	1668 pixels
Image Width	-	-	1556 pixels
Number of Components	-	-	3
Component 1	-	-	Y component: Quantization table 0, Sampling factors 1 horiz/1 vert
Component 2	-	-	Cb component: Quantization table 1, Sampling factors 1 horiz/1 vert
Component 3	-	-	Cr component: Quantization table 1, Sampling factors 1 horiz/1 vert
Thumbnail Compression	-	-	JPEG (old-style)
X Resolution	-	-	72 dots per inch
Y Resolution	-	-	72 dots per inch
Resolution Unit	-	-	Inch
Thumbnail Offset	-	-	302 bytes
Thumbnail Length	-	-	3331 bytes

Tabel 4.11 Perbandingan Membaca Metadata file Type Jpg (Lanjutan)

Jenis Metadata	Metadata Extractor	Metadata Forensik	Metadata Forensik yang di Usulkan
Profile Size	-	-	3144
CMM Type	-	-	Lino
Version	-	-	2.1.0
Class	-	-	Display Device
Color space	-	-	RGB
Profile Connection Space	-	-	XYZ
Profile Date/Time	-	-	Mon Mar 09 13:49:00 ICT 1998
Signature	-	-	Acsp
Primary Platform	-	-	Microsoft Corporation
Device manufacturer	-	-	IEC
Device model	-	-	sRGB
XYZ values	-	-	0.9642029 1.0 0.8249054
Tag Count	-	-	17
Copyright	-	-	Copyright (c) 1998 Hewlett-Packard Company
Profile Description	-	-	sRGB IEC61966-2.1
Media White Point	-	-	(0.9504547, 1.0, 1.0890503)
Media Black Point	-	-	(0.0, 0.0, 0.0)
Red Colorant	-	-	(0.43606567, 0.2224884, 0.013916016)
Green Colorant	-	-	(0.3851471, 0.71687317, 0.097076416)
Blue Colorant	-	-	(0.1430664, 0.06060791, 0.71409607)
Device Mfg Description	-	-	IEC **http://www.iec.ch**
Device Model Description	-	-	IEC 61966-2.1 Default RGB colour space – sRGB
Viewing Conditions Description	-	-	Reference Viewing Condition in IEC61966-2.1
Viewing Conditions	-	-	view(0x76696577): 36 bytes
Luminance	-	-	(76.03647, 80.0, 87.12462)
Measurement	-	-	1931 2° Observer, Backing (0.0, 0.0, 0.0), Geometry Unknown, Flare 1%, Illuminant D65
Technology	-	-	CRT
Red TRC	-	-	0.0000763, 0.0001526, , dll
Green TRC	-	-	0.0000763, 0.0001526, , dll
Blue TRC	-	-	0.9955596, 0.9977722, dll
MD5	-	dfcdf3bb5dfb01292796b594382daeb5	ff989fe3ec144ef1194f878ef9f4057b

Tabel 4.11 Perbandingan Membaca Metadata file Type Jpg (Lanjutan)

Jenis Metadata	Metadata Extractor	Metadata Forensik	Metadata Forensik yang di Usulkan
SHA-256	-	7974c1e041502d6c7bfe89485b1 aa068ad9eff565c667ab05d4e374 0504b740f	b75d5a3ccc8e4f4e1e1a3ab8cb16c5b16ed537699eb886880e60fa23fd6ad55f
System Type	amd64	-	-
Jenis OS	Windows 10	-	-
JVM	Oracle Corporation 1.8.0_20	-	-

B. Source Code Aplikasi Metadata Forensik

```
import com.drew.imaging.ImageMetadataReader;
import com.drew.metadata.Directory;
import com.drew.metadata.Tag;
import java.awt.Color;
import java.awt.Desktop;
import java.awt.event.MouseEvent;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileFilter;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.attribute.BasicFileAttributes;
import java.nio.file.attribute.FileOwnerAttributeView;
import java.nio.file.attribute.UserPrincipal;
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Date;
import java.util.Enumeration;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.AbstractButton;
import javax.swing.ButtonGroup;
import javax.swing.JComboBox;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JRadioButton;
import javax.swing.table.DefaultTableModel;
import org.apache.commons.io.FileUtils;
import org.apache.commons.io.FilenameUtils;
*
 * @author ASUS
 */
public class Home extends javax.swing.JFrame {
    SimpleDateFormat sdf2 = new SimpleDateFormat("YYYY-MM-dd hh:mm:ss a z");
    /**
     * Creates new form Home
     */
    public Home() {
        initComponents();
        setIconImage(new javax.swing.ImageIcon(this.getClass().getResource("/tag.png")).getImage());
        tabelMeta.setModel(tModel);
```

```
    tabelHasil.setModel(tModelHasil);
    disablekorelasi(false);
    disablekorelasiDate(false);
    setLocationRelativeTo(null);
    //jPanel12.setVisible(false);
    //jPanel9.setVisible(false);
}
/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {
    jFileChooser1 = new javax.swing.JFileChooser();
    buttonGroup1 = new javax.swing.ButtonGroup();
    jLabel5 = new javax.swing.JLabel();
    jFileChooser2 = new javax.swing.JFileChooser();
    jPopupMenu1 = new javax.swing.JPopupMenu();
    jMenuItem1 = new javax.swing.JMenuItem();
    jMenuItem2 = new javax.swing.JMenuItem();
    popUp = new javax.swing.JPopupMenu();
    SemuaType = new javax.swing.JMenuItem();
    FileType = new javax.swing.JMenuItem();
    buttonGroup2 = new javax.swing.ButtonGroup();
    popUpDate = new javax.swing.JPopupMenu();
    creationTime = new javax.swing.JMenuItem();
    lastModifiedTime = new javax.swing.JMenuItem();
    lastAccessTime = new javax.swing.JMenuItem();
    buttonGroup3 = new javax.swing.ButtonGroup();
    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    txtNamaFile = new javax.swing.JTextField();
    jButton1 = new javax.swing.JButton();
    jTabbedPane2 = new javax.swing.JTabbedPane();
    jPanel3 = new javax.swing.JPanel();
    jLabel2 = new javax.swing.JLabel();
    txtPath = new javax.swing.JTextField();
    jButton3 = new javax.swing.JButton();
    jPanel8 = new javax.swing.JPanel();
    jLabel10 = new javax.swing.JLabel();
    txtKorelasiTglFile = new javax.swing.JTextField();
    cmbFilterDateTime = new javax.swing.JComboBox();
    chkAuthor = new javax.swing.JCheckBox();
    chkSize = new javax.swing.JCheckBox();
    chkFiletype = new javax.swing.JCheckBox();
    chkOwner = new javax.swing.JCheckBox();
    txtOwnerManual = new javax.swing.JTextField();
    chkDateTime = new javax.swing.JCheckBox();
    jPanel9 = new javax.swing.JPanel();
    jButton2 = new javax.swing.JButton();
    jPanel2 = new javax.swing.JPanel();
    jPanel13 = new javax.swing.JPanel();
    rbSama = new javax.swing.JRadioButton();
    rbSamaKecil = new javax.swing.JRadioButton();
    rbSamaBesar = new javax.swing.JRadioButton();
    rbBesar = new javax.swing.JRadioButton();
    rbKecil = new javax.swing.JRadioButton();
    rbantara = new javax.swing.JRadioButton();
    jLabel11 = new javax.swing.JLabel();
    txtAntaraAwal = new javax.swing.JFormattedTextField();
    jLabel12 = new javax.swing.JLabel();
    txtAntaraAkhir = new javax.swing.JFormattedTextField();
    jPanel14 = new javax.swing.JPanel();
    rbSama1 = new javax.swing.JRadioButton();
    rbSamaKecil1 = new javax.swing.JRadioButton();
    rbSamaBesar1 = new javax.swing.JRadioButton();
    rbBesar1 = new javax.swing.JRadioButton();
    rbKecil1 = new javax.swing.JRadioButton();
    rbantara1 = new javax.swing.JRadioButton();
    jLabel20 = new javax.swing.JLabel();
    tglAnataraAkhir1 = new com.toedter.calendar.JDateChooser();
    jLabel21 = new javax.swing.JLabel();
    tglAntaraAwal1 = new com.toedter.calendar.JDateChooser();
    jPanel4 = new javax.swing.JPanel();
    jScrollPane2 = new javax.swing.JScrollPane();
    tabelHasil = new javax.swing.JTable();
    jButton4 = new javax.swing.JButton();
    jButton5 = new javax.swing.JButton();
    jButton6 = new javax.swing.JButton();
    jLabel14 = new javax.swing.JLabel();
    labelJumlahFile = new javax.swing.JLabel();
    jLabel17 = new javax.swing.JLabel();
    jPanel10 = new javax.swing.JPanel();
    jLabel13 = new javax.swing.JLabel();
    jTabbedPane1 = new javax.swing.JTabbedPane();
```

```
jPanel5 = new javax.swing.JPanel();
jLabel3 = new javax.swing.JLabel();
jLabel6 = new javax.swing.JLabel();
jLabel4 = new javax.swing.JLabel();
jLabel7 = new javax.swing.JLabel();
jLabel8 = new javax.swing.JLabel();
txtLokasi = new javax.swing.JTextField();
txtNamaFile2 = new javax.swing.JTextField();
txtTypeFile = new javax.swing.JTextField();
txtAuthors = new javax.swing.JTextField();
txtComputer = new javax.swing.JTextField();
jLabel9 = new javax.swing.JLabel();
txtOwner = new javax.swing.JTextField();
jPanel7 = new javax.swing.JPanel();
jScrollPane3 = new javax.swing.JScrollPane();
txtSUM = new javax.swing.JTextArea();
jPanel6 = new javax.swing.JPanel();
jScrollPane1 = new javax.swing.JScrollPane();
tabelMeta = new javax.swing.JTable();
jLabel15 = new javax.swing.JLabel();
labelStatusFile = new javax.swing.JLabel();
jLabel5.setText("jLabel5");
jMenuItem1.setText("Open File");
jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem1ActionPerformed(evt);
    }
});
jPopupMenu1.add(jMenuItem1);

jMenuItem2.setText("Open File Location");
jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem2ActionPerformed(evt);
    }
});
jPopupMenu1.add(jMenuItem2);

SemuaType.setText("Semua Type");
popUp.add(SemuaType);

FileType.setText("File Type");
popUp.add(FileType);

creationTime.setText("Creation Time");
popUpDate.add(creationTime);

lastModifiedTime.setText("Last Modified Time");
popUpDate.add(lastModifiedTime);

lastAccessTime.setText("Last Access Time");
popUpDate.add(lastAccessTime);

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setTitle("Korelasi Metadata");

jPanel1.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(51, 102, 255)));

jLabel1.setText("Cari File Bukti Utama");

txtNamaFile.setEditable(false);

jButton1.setText("Browse");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jLabel1)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(txtNamaFile)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(jButton1)
        .addContainerGap())
);
jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
                .addComponent(jLabel1)
                .addComponent(txtNamaFile, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jButton1))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jTabbedPane2.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 102, 255)));

        jLabel2.setText("Tentukan Lokasi Korelasi");

        txtPath.setEditable(false);

        jButton3.setText("Browse");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jPanel8.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.createEtchedBorder(), "Korelasi Berdasarkan ",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT_POSITION, new java.awt.Font("SimSun", 1,
18))); // NOI18N

        jLabel10.setText("Korelasi Berdasarkan");

        txtKorelasiTglFile.setEditable(false);
        txtKorelasiTglFile.setBackground(new java.awt.Color(204, 255, 204));

        cmbFilterDateTime.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "  ", "Creation Time", "Last Modified Time", "Last Access
Time" }));
        cmbFilterDateTime.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cmbFilterDateTimeActionPerformed(evt);
            }
        });

        chkAuthor.setText("Author");

        chkSize.setText("Size");
        chkSize.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                chkSizeActionPerformed(evt);
            }
        });

        chkFiletype.setText("File Type");

        chkOwner.setText("Owner");
        chkOwner.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                chkOwnerMouseClicked(evt);
            }
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                chkOwnerMouseReleased(evt);
            }
        });

        chkDateTime.setText("Date Time");
        chkDateTime.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                chkDateTimeActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel8Layout = new javax.swing.GroupLayout(jPanel8);
        jPanel8.setLayout(jPanel8Layout);
        jPanel8Layout.setHorizontalGroup(
            jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addGap(10, 10, 10)
                        .addComponent(jLabel10)
                        .addGap(0, 0, Short.MAX_VALUE))
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addContainerGap()
                        .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                            .addGroup(jPanel8Layout.createSequentialGroup()
                                .addComponent(chkDateTime)
                                .addGap(18, 18, 18)
                                .addComponent(cmbFilterDateTime, javax.swing.GroupLayout.PREFERRED_SIZE, 219,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(javax.swing.GroupLayout.Alignment.LEADING, jPanel8Layout.createSequentialGroup()
                                .addComponent(chkAuthor, javax.swing.GroupLayout.PREFERRED_SIZE, 101, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(27, 27, 27)
                                .addComponent(chkSize, javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                        .addGap(18, 18, 18)
                        .addComponent(chkFiletype, javax.swing.GroupLayout.PREFERRED_SIZE, 123, javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(18, 18, 18)
                .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(txtKorelasiTglFile, javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(chkOwner)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(txtOwnerManual)))))
            .addContainerGap())
        );
        jPanel8Layout.setVerticalGroup(
            jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel10)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(chkAuthor)
                    .addComponent(chkSize)
                    .addComponent(chkFiletype)
                    .addComponent(chkOwner)
                    .addComponent(txtOwnerManual, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(chkDateTime)
                    .addComponent(cmbFilterDateTime, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtKorelasiTglFile, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jPanel9.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(51, 102, 255)));

        jButton2.setText("Korelasi File");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel9Layout = new javax.swing.GroupLayout(jPanel9);
        jPanel9.setLayout(jPanel9Layout);
        jPanel9Layout.setHorizontalGroup(
            jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel9Layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jButton2)
                .addContainerGap())
        );
        jPanel9Layout.setVerticalGroup(
            jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel9Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jButton2)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 102, 255)));

        jPanel13.setBackground(new java.awt.Color(204, 255, 204));
        jPanel13.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 51, 255)),
"Size", javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("SimSun", 1, 18))); // NOI18N

        buttonGroup1.add(rbSama);
        rbSama.setSelected(true);
        rbSama.setText("Sama Dengan");
        rbSama.setOpaque(false);

        buttonGroup1.add(rbSamaKecil);
        rbSamaKecil.setText("Lebih Kecil Sama Dengan");
        rbSamaKecil.setOpaque(false);

        buttonGroup1.add(rbSamaBesar);
        rbSamaBesar.setText("Lebih Besar Sama Dengan");
        rbSamaBesar.setOpaque(false);

        buttonGroup1.add(rbBesar);
        rbBesar.setText("Lebih Besar");
        rbBesar.setOpaque(false);

        buttonGroup1.add(rbKecil);
        rbKecil.setText("Lebih Kecil");
        rbKecil.setOpaque(false);
```

```
        buttonGroup1.add(rbantara);
        rbantara.setText("Antara");
        rbantara.setOpaque(false);
        rbantara.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                rbantaraActionPerformed(evt);
            }
        });

        jLabel11.setText("Batas Awal");

        txtAntaraAwal.setFormatterFactory(new javax.swing.text.DefaultFormatterFactory(new javax.swing.text.NumberFormatter(new
java.text.DecimalFormat("#0"))));

        jLabel12.setText("Batas Akhir");

        txtAntaraAkhir.setFormatterFactory(new javax.swing.text.DefaultFormatterFactory(new javax.swing.text.NumberFormatter(new
java.text.DecimalFormat("#0"))));

        javax.swing.GroupLayout jPanel13Layout = new javax.swing.GroupLayout(jPanel13);
        jPanel13.setLayout(jPanel13Layout);
        jPanel13Layout.setHorizontalGroup(
            jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel13Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(rbSama)
                    .addComponent(rbSamaKecil)
                    .addComponent(rbKecil)
                    .addComponent(rbBesar)
                    .addComponent(rbSamaBesar)
                    .addComponent(rbantara, javax.swing.GroupLayout.PREFERRED_SIZE, 222, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(jPanel13Layout.createSequentialGroup()
                        .addGap(28, 28, 28)
                        .addGroup(jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel11)
                            .addComponent(jLabel12))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addGroup(jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(txtAntaraAkhir, javax.swing.GroupLayout.PREFERRED_SIZE, 150, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(txtAntaraAwal, javax.swing.GroupLayout.PREFERRED_SIZE, 150, javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        jPanel13Layout.setVerticalGroup(
            jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel13Layout.createSequentialGroup()
                .addComponent(rbSama)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(rbSamaKecil)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbSamaBesar)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbBesar)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbKecil)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(rbantara)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel11)
                    .addComponent(txtAntaraAwal, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel13Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel12)
                    .addComponent(txtAntaraAkhir, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jPanel14.setBackground(new java.awt.Color(204, 255, 255));
        jPanel14.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 51, 255)),
"Date Time", javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("SimSun", 1, 18))); // NOI18N

        buttonGroup3.add(rbSama1);
        rbSama1.setSelected(true);
        rbSama1.setText("Sama Dengan");
        rbSama1.setOpaque(false);

        buttonGroup3.add(rbSamaKecil1);
        rbSamaKecil1.setText("Lebih Kecil Sama Dengan");
        rbSamaKecil1.setOpaque(false);

        buttonGroup3.add(rbSamaBesar1);
        rbSamaBesar1.setText("Lebih Besar Sama Dengan");
```

```
        rbSamaBesar1.setOpaque(false);

        buttonGroup3.add(rbBesar1);
        rbBesar1.setText("Lebih Besar");
        rbBesar1.setOpaque(false);

        buttonGroup3.add(rbKecil1);
        rbKecil1.setText("Lebih Kecil");
        rbKecil1.setOpaque(false);

        buttonGroup3.add(rbantara1);
        rbantara1.setText("Antara");
        rbantara1.setOpaque(false);
        rbantara1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                rbantara1ActionPerformed(evt);
            }
        });

        jLabel20.setText("Dari Tanggal/jam");

        tglAnataraAkhir1.setDate(new Date());
        tglAnataraAkhir1.setDateFormatString("dd MM yyyy hh:mm:ss");

        jLabel21.setText("Sampai Tangal/Jam");

        tglAntaraAwal1.setDate(new Date());
        tglAntaraAwal1.setDateFormatString("dd MM yyyy hh:mm:ss");

        javax.swing.GroupLayout jPanel14Layout = new javax.swing.GroupLayout(jPanel14);
        jPanel14.setLayout(jPanel14Layout);
        jPanel14Layout.setHorizontalGroup(
            jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel14Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(rbSama1)
                    .addComponent(rbSamaKecil1)
                    .addComponent(rbKecil1)
                    .addComponent(rbBesar1)
                    .addComponent(rbSamaBesar1)
                    .addComponent(rbantara1, javax.swing.GroupLayout.PREFERRED_SIZE, 222, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(jPanel14Layout.createSequentialGroup()
                        .addGap(28, 28, 28)
                        .addGroup(jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel21)
                            .addComponent(jLabel20))
                        .addGap(18, 18, 18)
                        .addGroup(jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(tglAntaraAwal1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(tglAnataraAkhir1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        jPanel14Layout.setVerticalGroup(
            jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel14Layout.createSequentialGroup()
                .addComponent(rbSama1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(rbSamaKecil1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbSamaBesar1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbBesar1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(rbKecil1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(rbantara1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel20)
                    .addComponent(tglAntaraAwal1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel14Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel21)
                    .addComponent(tglAnataraAkhir1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
            jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
```

```
            .addContainerGap()
            .addComponent(jPanel13, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jPanel14, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(
            jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                    .addComponent(jPanel14, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel13, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(
            jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                    .addComponent(jPanel9, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel2)
                        .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                            .addGroup(jPanel3Layout.createSequentialGroup()
                                .addComponent(txtPath)
                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(jButton3))
                            .addComponent(jPanel2, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jPanel8, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        jPanel3Layout.setVerticalGroup(
            jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel3Layout.createSequentialGroup()
                        .addComponent(txtPath, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel8, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jPanel9, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jButton3))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        jTabbedPane2.addTab("Korelasi Meta Data", jPanel3);

        tabelHasil.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3", "Title 4", "Title 5"
            }
        ));
        tabelHasil.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                tabelHasilMouseClicked(evt);
            }
        });
        jScrollPane2.setViewportView(tabelHasil);

        jButton4.setText("Copy All File to One Folder");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton4ActionPerformed(evt);
```

```
    }
});

jButton5.setText("Save Log Result File");
jButton5.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton5ActionPerformed(evt);
    }
});

jButton6.setText("Load Log Result File ");
jButton6.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton6ActionPerformed(evt);
    }
});

jLabel14.setText("file yang ditemukan");

labelJumlahFile.setFont(new java.awt.Font("SimSun", 1, 24)); // NOI18N
labelJumlahFile.setText("0");

jLabel17.setText("Terdapat");

javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
    jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jScrollPane2)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()
                .addGap(0, 66, Short.MAX_VALUE)
                .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()
                        .addComponent(jButton6)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jButton5)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jButton4))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()
                        .addComponent(jLabel17)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(labelJumlahFile)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel14)))))
        .addContainerGap())
);
jPanel4Layout.setVerticalGroup(
    jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT_SIZE, 576, Short.MAX_VALUE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel14)
            .addComponent(labelJumlahFile)
            .addComponent(jLabel17))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jButton4)
            .addComponent(jButton5)
            .addComponent(jButton6))
        .addContainerGap())
);

jTabbedPane2.addTab("Hasil Korelasi", jPanel4);

jPanel10.setBorder(javax.swing.BorderFactory.createEtchedBorder());

jLabel13.setText("Copy Right @ 2018 By Zaenudin");

javax.swing.GroupLayout jPanel10Layout = new javax.swing.GroupLayout(jPanel10);
jPanel10.setLayout(jPanel10Layout);
jPanel10Layout.setHorizontalGroup(
    jPanel10Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel10Layout.createSequentialGroup()
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addComponent(jLabel13)
        .addContainerGap())
);
jPanel10Layout.setVerticalGroup(
    jPanel10Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jLabel13)
);
```

```
        jTabbedPane1.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(51, 102, 255)));
        jTabbedPane1.setOpaque(true);

        jLabel3.setText("Lokasi File ");

        jLabel6.setText("Nama File");

        jLabel4.setText("Extensi / Type File");

        jLabel7.setText("Authors");

        jLabel8.setText("Computer");

        txtLokasi.setEditable(false);
        txtLokasi.setBackground(new java.awt.Color(204, 255, 204));

        txtNamaFile2.setEditable(false);
        txtNamaFile2.setBackground(new java.awt.Color(204, 255, 204));

        txtTypeFile.setEditable(false);
        txtTypeFile.setBackground(new java.awt.Color(204, 255, 204));

        txtAuthors.setEditable(false);
        txtAuthors.setBackground(new java.awt.Color(204, 255, 204));

        txtComputer.setEditable(false);
        txtComputer.setBackground(new java.awt.Color(204, 255, 204));

        jLabel9.setText("Owner");

        txtOwner.setEditable(false);
        txtOwner.setBackground(new java.awt.Color(204, 255, 204));

        javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel3)
                    .addComponent(jLabel6)
                    .addComponent(jLabel4)
                    .addComponent(jLabel7)
                    .addComponent(jLabel8)
                    .addComponent(jLabel9))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                        .addComponent(txtAuthors, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE, 444,
Short.MAX_VALUE)
                        .addComponent(txtNamaFile2, javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(txtLokasi, javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(txtComputer, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED_SIZE, 355,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(txtTypeFile, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED_SIZE, 282,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(txtOwner, javax.swing.GroupLayout.PREFERRED_SIZE, 249, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(123, Short.MAX_VALUE))
        );
        jPanel5Layout.setVerticalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel3)
                    .addComponent(txtLokasi, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel6)
                    .addComponent(txtNamaFile2, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel4)
                    .addComponent(txtTypeFile, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel7)
                    .addComponent(txtAuthors, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel8)
```

```
                .addComponent(txtComputer, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jLabel9)
                .addComponent(txtOwner, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(435, Short.MAX_VALUE))
        );

        jTabbedPane1.addTab("General", jPanel5);

        txtSUM.setEditable(false);
        txtSUM.setColumns(20);
        txtSUM.setRows(5);
        txtSUM.setAutoscrolls(false);
        jScrollPane3.setViewportView(txtSUM);

        javax.swing.GroupLayout jPanel7Layout = new javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(
            jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane3, javax.swing.GroupLayout.DEFAULT_SIZE, 649, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel7Layout.setVerticalGroup(
            jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addComponent(jScrollPane3, javax.swing.GroupLayout.DEFAULT_SIZE, 585, Short.MAX_VALUE)
                .addContainerGap())
        );

        jTabbedPane1.addTab("CheckSum", jPanel7);

        tabelMeta.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null},
                {null, null, null},
                {null, null, null},
                {null, null, null}
            },
            new String [] {
                "Title 1", "Title 2", "Title 3"
            }
        ));
        jScrollPane1.setViewportView(tabelMeta);

        javax.swing.GroupLayout jPanel6Layout = new javax.swing.GroupLayout(jPanel6);
        jPanel6.setLayout(jPanel6Layout);
        jPanel6Layout.setHorizontalGroup(
            jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel6Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 649, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel6Layout.setVerticalGroup(
            jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 570, Short.MAX_VALUE)
                .addContainerGap())
        );

        jTabbedPane1.addTab("Detail", jPanel6);

        jLabel15.setFont(new java.awt.Font("SimSun", 1, 18)); // NOI18N
        jLabel15.setText("Status File:");

        labelStatusFile.setBackground(new java.awt.Color(204, 255, 204));
        labelStatusFile.setFont(new java.awt.Font("SimSun", 1, 24)); // NOI18N
        labelStatusFile.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        labelStatusFile.setOpaque(true);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jPanel10, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                            .addComponent(jTabbedPane1)
                            .addComponent(jLabel15)
                            .addComponent(labelStatusFile, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jTabbedPane2)
                        .addGap(3, 3, 3)))
                .addContainerGap()))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jTabbedPane1)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel15)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(labelStatusFile, javax.swing.GroupLayout.PREFERRED_SIZE, 43, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jTabbedPane2))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel10, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        pack();
    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        kosong();
        if (jFileChooser1 == null) {
            jFileChooser1 = new JFileChooser();
            jFileChooser1.addChoosableFileFilter(new MyCostumFilter());
            jFileChooser1.setAcceptAllFileFilterUsed(false);
        }
        int a = jFileChooser1.showOpenDialog(this);
        if (a == JFileChooser.APPROVE_OPTION) {
            File file = jFileChooser1.getSelectedFile();
            txtNamaFile.setText(file.getAbsolutePath());
            getMetaData(file, file.getName());
        }

    }

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int row = tabelHasil.getSelectedRow();
        if (row >= 0) {
            String path = tabelHasil.getValueAt(row, 3).toString();
            File file = new File(path);
            if (!Desktop.isDesktopSupported()) {
                JOptionPane.showMessageDialog(null, "File tidak didukung untuk dibuka");
                return;
            }

            Desktop desktop = Desktop.getDesktop();
            if (file.exists()) {
                try {
                    desktop.open(file);
                } catch (IOException ex) {
                    JOptionPane.showMessageDialog(null, "File tidak dapat dibuka");
                }
            }

        } else {
            JOptionPane.showMessageDialog(null, "Pilih File Terlebih Dahulu");
        }
    }

    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int row = tabelHasil.getSelectedRow();
        if (row >= 0) {
            String path = tabelHasil.getValueAt(row, 3).toString();
            File file = new File(path);
            file = file.getParentFile();
            if (!Desktop.isDesktopSupported()) {
                JOptionPane.showMessageDialog(null, "File tidak didukung untuk dibuka");
                return;
```

```
        }

        Desktop desktop = Desktop.getDesktop();
        if (file.exists()) {
            try {
                desktop.open(file);
            } catch (IOException ex) {
                JOptionPane.showMessageDialog(null, "File tidak dapat dibuka");
            }
        }

    } else {
        JOptionPane.showMessageDialog(null, "Pilih File Terlebih Dahulu");
    }
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    jFileChooser2 = new JFileChooser();
    jFileChooser2.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
    jFileChooser2.setAcceptAllFileFilterUsed(false);
    int a = jFileChooser2.showOpenDialog(this);
    int suksess = 0, gagal = 0;
    if (a == JFileChooser.APPROVE_OPTION) {
        File file = jFileChooser2.getSelectedFile();
        String tujuan = file.getAbsolutePath();
        for (int i = 0; i < tabelHasil.getRowCount(); i++) {
            String asal = tabelHasil.getValueAt(i, 3).toString();
            String namafile = tabelHasil.getValueAt(i, 0).toString();
            File fileasal = new File(asal);
            File filetujuan = new File(tujuan + "/" + namafile);
            try {
                FileUtils.copyFile(fileasal, filetujuan);
                suksess++;
            } catch (IOException ex) {
                gagal++;
            }
        }
        JOptionPane.showMessageDialog(null, "File Suksess : " + suksess + " File Gagal : " + gagal);
    }
}

private void tabelHasilMouseClicked(java.awt.event.MouseEvent evt) {
    if (evt.getButton() == MouseEvent.BUTTON3) {
        jPopupMenu1.setInvoker(tabelHasil);
        jPopupMenu1.pack();
        jPopupMenu1.setVisible(true);
        jPopupMenu1.setLocation(evt.getPoint());
    }
}

private void chkOwnerMouseReleased(java.awt.event.MouseEvent evt) {
    if (chkOwner.isSelected()) {
        txtOwnerManual.setEnabled(true);
    } else {
        txtOwnerManual.setEnabled(false);
    }
}

private void chkOwnerMouseClicked(java.awt.event.MouseEvent evt) {
    if (chkOwner.isSelected()) {
        txtOwnerManual.setEnabled(true);
    } else {
        txtOwnerManual.setEnabled(false);
    }
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    jFileChooser2 = new JFileChooser();
    jFileChooser2.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
    jFileChooser2.setAcceptAllFileFilterUsed(false);

    int a = jFileChooser2.showOpenDialog(this);
    if (a == JFileChooser.APPROVE_OPTION) {
        File file = jFileChooser2.getSelectedFile();
        txtPath.setText(file.getAbsolutePath());
        // getMetaData(file);
    }
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    if (txtPath.getText().isEmpty()) {
        JOptionPane.showMessageDialog(rootPane, "Pilih Lokasi Korelasi terlebih dahulu");
    } else {
        if (rbSama.isSelected()) {
```

```
    perbandingan = 1;
} else if (rbBesar.isSelected()) {
    perbandingan = 3;
} else if (rbKecil.isSelected()) {
    perbandingan = 2;
} else if (rbSamaBesar.isSelected()) {
    perbandingan = 5;
} else if (rbantara.isSelected()) {
    perbandingan = 7;
    try {
        sizeawal = Integer.parseInt(txtAntaraAwal.getText());
        sizeakhir = Integer.parseInt(txtAntaraAkhir.getText());
        // System.out.println("Size awal:"+sizeawal+" Size Kahir: "+sizeakhir);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Input Size Awal atau Size akhir salah");
    }
} else {
    perbandingan = 4;
}
if (rbSama1.isSelected()) {
    perbandingantgl = 1;
} else if (rbBesar1.isSelected()) {
    perbandingantgl = 3;
} else if (rbKecil1.isSelected()) {
    perbandingantgl = 2;
} else if (rbSamaBesar1.isSelected()) {
    perbandingantgl = 5;
} else if (rbantara1.isSelected()) {
    perbandingantgl = 7;
    try {
        tglawal = tglAntaraAwal1.getDate().getTime();
        tglakhir = tglAnataraAkhir1.getDate().getTime();
        // System.out.println("Size awal:"+sizeawal+" Size Kahir: "+sizeakhir);
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Input Size Awal atau Size akhir salah");
    }
} else {
    perbandingantgl = 4;
}
/*
if (rbAuthor.isSelected()) {
    getHasil(txtPath.getText(), author, true);
    walk(txtPath.getText(), author);
} else if (rbFileType.isSelected()) {
    getHasil(txtPath.getText(), extensi, true);
    walk(txtPath.getText(), extensi);
} else
 */

if (chkAuthor.isSelected()) {
    jTabbedPane2.setSelectedIndex(1);
    if (chkSize.isSelected()) {
        if (chkFiletype.isSelected()) {
            if (chkOwner.isSelected()) {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterAuthorSizeFileTypeOwnerTgl, true);
                    walk(txtPath.getText(), filterAuthorSizeFileTypeOwnerTgl);
                } else {
                    getHasil(txtPath.getText(), filterAuthorSizeFileTypeOwner, true);
                    walk(txtPath.getText(), filterAuthorSizeFileTypeOwner);
                }

            } else {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterAuthorSizeFileTypeTgl, true);
                    walk(txtPath.getText(), filterAuthorSizeFileTypeTgl);
                } else {
                    getHasil(txtPath.getText(), filterAuthorSizeFileType, true);
                    walk(txtPath.getText(), filterAuthorSizeFileType);
                }
            }
        } else {
            if (chkOwner.isSelected()) {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterAuthorSizeOwnerTgl, true);
                    walk(txtPath.getText(), filterAuthorSizeOwnerTgl);
                } else {
                    getHasil(txtPath.getText(), filterAuthorSizeOwner, true);
                    walk(txtPath.getText(), filterAuthorSizeOwner);
                }

            } else {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterAuthorSizeTgl, true);
                    walk(txtPath.getText(), filterAuthorSizeTgl);
                } else {
                    getHasil(txtPath.getText(), filterAuthorSize, true);
```

```
                walk(txtPath.getText(), filterAuthorSize);
            }
        }
    }
} else {
    if (chkFiletype.isSelected()) {
        if (chkOwner.isSelected()) {
            if (chkDateTime.isSelected()) {
                getHasil(txtPath.getText(), filterAuthorFileTypeOwnerTgl, true);
                walk(txtPath.getText(), filterAuthorFileTypeOwnerTgl);
            } else {
                getHasil(txtPath.getText(), filterAuthorFileTypeOwner, true);
                walk(txtPath.getText(), filterAuthorFileTypeOwner);
            }

        } else {
            if (chkDateTime.isSelected()) {
                getHasil(txtPath.getText(), filterAuthorFileTypeTgl, true);
                walk(txtPath.getText(), filterAuthorFileTypeTgl);
            } else {
                getHasil(txtPath.getText(), filterAuthorFileType, true);
                walk(txtPath.getText(), filterAuthorFileType);
            }
        }
    } else {
        if (chkOwner.isSelected()) {
            if (chkDateTime.isSelected()) {
                getHasil(txtPath.getText(), filterAuthorOwnerTgl, true);
                walk(txtPath.getText(), filterAuthorOwnerTgl);
            } else {
                getHasil(txtPath.getText(), filterAuthorOwner, true);
                walk(txtPath.getText(), filterAuthorOwner);
            }

        } else {
            if (chkDateTime.isSelected()) {
                getHasil(txtPath.getText(), filterAuthorTgl, true);
                walk(txtPath.getText(), filterAuthorTgl);
            } else {
                getHasil(txtPath.getText(), filterAuthor, true);
                walk(txtPath.getText(), filterAuthor);
            }
        }
    }
}
if (tabelHasil.getRowCount() <= 0) {
    JOptionPane.showMessageDialog(null, "Tidak ada file yang ditemukan");
}
} else {
    if (chkSize.isSelected()) {
        jTabbedPane2.setSelectedIndex(1);
        if (chkFiletype.isSelected()) {
            if (chkOwner.isSelected()) {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterSizeFileTypeOwnerTgl, true);
                    walk(txtPath.getText(), filterSizeFileTypeOwnerTgl);
                } else {
                    getHasil(txtPath.getText(), filterSizeFileTypeOwner, true);
                    walk(txtPath.getText(), filterSizeFileTypeOwner);
                }
            } else {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterSizeFileTypeTgl, true);
                    walk(txtPath.getText(), filterSizeFileTypeTgl);
                } else {
                    getHasil(txtPath.getText(), filterSizeFileType, true);
                    walk(txtPath.getText(), filterSizeFileType);
                }
            }
        } else {
            if (chkOwner.isSelected()) {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterSizeOwnerTgl, true);
                    walk(txtPath.getText(), filterSizeOwnerTgl);
                } else {
                    getHasil(txtPath.getText(), filterSizeOwner, true);
                    walk(txtPath.getText(), filterSizeOwner);
                }
            } else {
                if (chkDateTime.isSelected()) {
                    getHasil(txtPath.getText(), filterSizeTgl, true);
                    walk(txtPath.getText(), filterSizeTgl);
                } else {
                    getHasil(txtPath.getText(), filterSize, true);
                    walk(txtPath.getText(), filterSize);
                }
            }
```

```
            }

        } else {
            if (chkFiletype.isSelected()) {
                if (chkOwner.isSelected()) {
                    if (chkDateTime.isSelected()) {
                        getHasil(txtPath.getText(), filterFileTypeOwnerTgl, true);
                        walk(txtPath.getText(), filterFileTypeOwnerTgl);
                    } else {
                        getHasil(txtPath.getText(), filterFileTypeOwner, true);
                        walk(txtPath.getText(), filterFileTypeOwner);
                    }
                } else {
                    if (chkDateTime.isSelected()) {
                        getHasil(txtPath.getText(), filterFileTypeTgl, true);
                        walk(txtPath.getText(), filterFileTypeTgl);
                    } else {
                        getHasil(txtPath.getText(), filterFileType, true);
                        walk(txtPath.getText(), filterFileType);
                    }
                }
            } else {
                if (chkOwner.isSelected()) {
                    if (chkDateTime.isSelected()) {
                        getHasil(txtPath.getText(), filterOwnerTgl, true);
                        walk(txtPath.getText(), filterOwnerTgl);
                    } else {
                        getHasil(txtPath.getText(), filterOwner, true);
                        walk(txtPath.getText(), filterOwner);
                    }
                } else {
                    if (chkDateTime.isSelected()) {
                        getHasil(txtPath.getText(), filterTgl, true);
                        walk(txtPath.getText(), filterTgl);
                    } else {
                        jTabbedPane2.setSelectedIndex(1);
                        getHasil(txtPath.getText(), filterOwner, true);
                        walk(txtPath.getText(), filterOwner);
                        if (tabelHasil.getRowCount() <= 0) {
                            JOptionPane.showMessageDialog(null, "Tidak ada file yang ditemukan");
                        }
                    }
                }

            }
        }

    }
}

private void cmbFilterDateTimeActionPerformed(java.awt.event.ActionEvent evt) {
    if (cmbFilterDateTime.getSelectedIndex() == 0) {
        disablekorelasiDate(false);
    } else {
        if (txtNamaFile.getText().isEmpty()) {
            JOptionPane.showMessageDialog(txtNamaFile, "Pilih file utama terlebih dahulu...!");
        } else if (txtPath.getText().isEmpty()) {
            JOptionPane.showMessageDialog(txtPath, "Pilih lokasi korelasi terlebih dahulu...!");
        } else {
            disablekorelasiDate(true);
            if (cmbFilterDateTime.getSelectedIndex() == 1) {
                txtKorelasiTglFile.setText(tabelMeta.getModel().getValueAt(2, 2).toString());
                //jPanel11.setVisible(false);
                //jPanel12.setVisible(true);
            } else if (cmbFilterDateTime.getSelectedIndex() == 2) {
                txtKorelasiTglFile.setText(tabelMeta.getModel().getValueAt(3, 2).toString());
                //jPanel11.setVisible(false);
                //jPanel12.setVisible(true);
            } else if (cmbFilterDateTime.getSelectedIndex() == 3) {
                txtKorelasiTglFile.setText(tabelMeta.getModel().getValueAt(4, 2).toString());
                //jPanel11.setVisible(false);
                // jPanel12.setVisible(true);
            }
        }
    }
}

private void chkSizeActionPerformed(java.awt.event.ActionEvent evt) {
    if (chkSize.isSelected()) {
        disablekorelasi(true);
    } else {
        disablekorelasi(false);
    }
}
```

```
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        jFileChooser2 = new JFileChooser();
        jFileChooser2.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        jFileChooser2.setAcceptAllFileFilterUsed(false);
        SimpleDateFormat sdfn = new SimpleDateFormat("YYYYMMdd hhmmss");
        int a = jFileChooser2.showOpenDialog(this);
        if (a == JFileChooser.APPROVE_OPTION) {
            File file = new File(jFileChooser2.getSelectedFile().getAbsolutePath() + "/LogFile-"
                    + sdfn.format(new Date()) + ".log");
            String content = "";
            content += txtNamaFile.getText() + "==>" + txtNamaFile2.getText() + "==>"
                    + chkAuthor.isSelected() + "==>" + chkSize.isSelected() + "==>"
                    + chkFiletype.isSelected() + "==>" + chkOwner.isSelected() + "==>"
                    + txtOwnerManual.getText() + "==>" + chkDateTime.isSelected() + "==>"
                    + cmbFilterDateTime.getSelectedIndex() + "==>" + txtKorelasiTglFile.getText() + "==>"
                    + pilihRadioButton(buttonGroup1) + "==>" + txtAntaraAwal.getText() + "==>" + txtAntaraAkhir.getText() + "==>"
                    + pilihRadioButton(buttonGroup2) + sdf2.format(tglAntaraAwal1.getDate()) + "==>"
                    + sdf2.format(tglAnataraAkhir1.getDate()) + "===>";
            for (int i = 0; i < tabelHasil.getRowCount(); i++) {
                content += tabelHasil.getValueAt(i, 0) + "==>" + tabelHasil.getValueAt(i, 1) + "==>"
                        + tabelHasil.getValueAt(i, 2) + "==>" + tabelHasil.getValueAt(i, 3) + "==>"
                        + tabelHasil.getValueAt(i, 4) + "===>";
            }
            simpanLog(file, content);
//            txtPath.setText(file.getAbsolutePath());
            // getMetaData(file);
        }
    }

    private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void rbantara1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void rbantaraActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void chkDateTimeActionPerformed(java.awt.event.ActionEvent evt) {
        if (chkDateTime.isSelected()) {
            disablekorelasiDate(true);
        } else {
            disablekorelasiDate(false);
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Windows".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Home().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JMenuItem FileType;
    private javax.swing.JMenuItem SemuaType;
```

```
private javax.swing.ButtonGroup buttonGroup1;
private javax.swing.ButtonGroup buttonGroup2;
private javax.swing.ButtonGroup buttonGroup3;
private javax.swing.JCheckBox chkAuthor;
private javax.swing.JCheckBox chkDateTime;
private javax.swing.JCheckBox chkFiletype;
private javax.swing.JCheckBox chkOwner;
private javax.swing.JCheckBox chkSize;
private javax.swing.JComboBox cmbFilterDateTime;
private javax.swing.JMenuItem creationTime;
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton6;
private javax.swing.JFileChooser jFileChooser1;
private javax.swing.JFileChooser jFileChooser2;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel14;
private javax.swing.JLabel jLabel15;
private javax.swing.JLabel jLabel17;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel20;
private javax.swing.JLabel jLabel21;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JMenuItem jMenuItem1;
private javax.swing.JMenuItem jMenuItem2;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel10;
private javax.swing.JPanel jPanel13;
private javax.swing.JPanel jPanel14;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel3;
private javax.swing.JPanel jPanel4;
private javax.swing.JPanel jPanel5;
private javax.swing.JPanel jPanel6;
private javax.swing.JPanel jPanel7;
private javax.swing.JPanel jPanel8;
private javax.swing.JPanel jPanel9;
private javax.swing.JPopupMenu jPopupMenu1;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JScrollPane jScrollPane2;
private javax.swing.JScrollPane jScrollPane3;
private javax.swing.JTabbedPane jTabbedPane1;
private javax.swing.JTabbedPane jTabbedPane2;
private javax.swing.JLabel labelJumlahFile;
private javax.swing.JLabel labelStatusFile;
private javax.swing.JMenuItem lastAccessTime;
private javax.swing.JMenuItem lastModifiedTime;
private javax.swing.JPopupMenu popUp;
private javax.swing.JPopupMenu popUpDate;
private javax.swing.JRadioButton rbBesar;
private javax.swing.JRadioButton rbBesar1;
private javax.swing.JRadioButton rbKecil;
private javax.swing.JRadioButton rbKecil1;
private javax.swing.JRadioButton rbSama;
private javax.swing.JRadioButton rbSama1;
private javax.swing.JRadioButton rbSamaBesar;
private javax.swing.JRadioButton rbSamaBesar1;
private javax.swing.JRadioButton rbSamaKecil;
private javax.swing.JRadioButton rbSamaKecil1;
private javax.swing.JRadioButton rbantara;
private javax.swing.JRadioButton rbantara1;
private javax.swing.JTable tabelHasil;
private javax.swing.JTable tabelMeta;
private com.toedter.calendar.JDateChooser tglAnataraAkhir1;
private com.toedter.calendar.JDateChooser tglAntaraAwal1;
private javax.swing.JFormattedTextField txtAntaraAkhir;
private javax.swing.JFormattedTextField txtAntaraAwal;
private javax.swing.JTextField txtAuthors;
private javax.swing.JTextField txtComputer;
private javax.swing.JTextField txtKorelasiTglFile;
private javax.swing.JTextField txtLokasi;
private javax.swing.JTextField txtNamaFile;
private javax.swing.JTextField txtNamaFile2;
private javax.swing.JTextField txtOwner;
```

```
    private javax.swing.JTextField txtOwnerManual;
    private javax.swing.JTextField txtPath;
    private javax.swing.JTextArea txtSUM;
    private javax.swing.JTextField txtTypeFile;
    // End of variables declaration

    String header[] = {"NO", "JENIS META", "VALUE"};
    String typeImage[] = {"jpeg", "png", "jpg", "mp4"};
    String typeDoc[] = {"doc", "docx", "pdf", "mp3"};
    DefaultTableModel tModel = new DefaultTableModel(header, 0);
    long size = 0, lastmodifide = 0, lastaccess = 0, creationtime = 0, sizeawal = 0, sizeakhir = 0,
            tglawal = 0, tglakhir = 0, tglTerpilih;

    public long getTanggalterpilih(JComboBox jc) {
        if (jc.getSelectedIndex() == 1) {
            tglTerpilih = creationtime;
        } else if (jc.getSelectedIndex() == 2) {
            tglTerpilih = lastaccess;
        } else if (jc.getSelectedIndex() == 3) {
            tglTerpilih = lastmodifide;
        }
        return tglTerpilih;
    }

    public void getMetaData(File f, String namafile) {
        tModel = new DefaultTableModel(header, 0);
        String ext = FilenameUtils.getExtension(namafile);
        String datam2[] = {"1", "Extention ", ext};
        tModel.addRow(datam2);
        String data0[] = {"2", "Path : ", f.getAbsolutePath()};
        tModel.addRow(data0);
//MEMBACA DETAI FILE
        FileInfo filein = new FileInfo(f);
        try {
            txtOwner.setText(filein.getOwner());
        } catch (Exception ex) {

        }

        txtLokasi.setText(f.getPath());
        txtNamaFile2.setText(namafile);
        txtTypeFile.setText(getFileExtension(f));
        txtComputer.setText(getComputerName());

        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(f.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner;
        try {
            owner = ownerAttributeView.getOwner();
            txtAuthors.setText(owner.getName().substring(owner.getName().lastIndexOf("\\") + 1));
        } catch (IOException ex) {
            Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);
        }
        //System.out.println("owner: " + owner.getName());

        String checksum = "MD5 :\n" + getChecksum(f.getPath(), "MD5")
                + "\nSHA-256 :\n" + getChecksum(f.getPath(), "SHA-256");
        txtSUM.setText(checksum);

        Path file = Paths.get(f.getAbsolutePath());
        BasicFileAttributes attr = null;
        try {
            attr = Files.readAttributes(file, BasicFileAttributes.class);
        } catch (IOException ex) {
            ex.printStackTrace();
        }
        long tgl1 = attr.creationTime().toMillis();
        long tgl2 = attr.lastModifiedTime().toMillis();
        Calendar cal = Calendar.getInstance();
        cal.setTimeInMillis(tgl1);
        Calendar cal2 = Calendar.getInstance();
        cal2.setTimeInMillis(tgl2);
        String data1[] = {"3", "creationTime: ", sdf2.format(cal.getTime())};
        tModel.addRow(data1);
        creationtime = attr.creationTime().toMillis();
        String data8[] = {"4", "lastAccessTime: ", attr.lastAccessTime().toString()};
        tModel.addRow(data8);
        lastaccess = attr.lastAccessTime().toMillis();
        String data2[] = {"5", "lastModifiedTime: ", sdf2.format(cal2.getTime())};
        tModel.addRow(data2);
        lastmodifide = attr.lastModifiedTime().toMillis();
        //System.out.println("LONG File: " + lastmodifide);
        String data3[] = {"6", "isDirectory: ", String.valueOf(attr.isDirectory())};
        tModel.addRow(data3);
        String data4[] = {"7", "isOther: ", String.valueOf(attr.isOther())};
        tModel.addRow(data4);
        String data5[] = {"8", "isRegularFile: ", String.valueOf(attr.isRegularFile())};
        tModel.addRow(data5);
```

```
    String data6[] = {"9", "isSymbolicLink: ", String.valueOf(attr.isSymbolicLink())};
    tModel.addRow(data6);
    String data7[] = {"10", "size: ", String.valueOf(attr.size())};
    size = attr.size();
    tModel.addRow(data7);
    //String data9[] = {"9", "isProgramName: ", String.valueOf(attr.)};
    //tModel.addRow(data6);
    if (Arrays.asList(typeImage).contains(ext)) {
        InputStream imageFile = null;
        com.drew.metadata.Metadata metadata = null;
        try {
            imageFile = new FileInputStream(f);
            metadata = ImageMetadataReader.readMetadata(imageFile);
        } catch (Exception ex) {
            Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);
        }
        int no = 10;
        for (Directory directory : metadata.getDirectories()) {
            for (Tag tag : directory.getTags()) {
                no++;
                String datam[] = {String.valueOf(no), tag.getTagName(), tag.getDescription()};
                tModel.addRow(datam);
            }
        }
    }

    if (cal.before(cal2)) {
        labelStatusFile.setText("File Sudah Di Modifikasi");
        labelStatusFile.setBackground(new Color(255, 0, 0));
    } else {
        labelStatusFile.setText("File Asli");
        labelStatusFile.setBackground(new Color(204, 255, 204));
    }

    tabelMeta.setModel(tModel);
    AturLebarKolom alk = new AturLebarKolom(tabelMeta);
    alk.adjustColumns();
}
String headerHasil[] = {"FILE", "SIZE", "DATE CREATION", "DATE MODIFICATION", "PATH"};
DefaultTableModel tModelHasil = new DefaultTableModel(headerHasil, 0);

SimpleDateFormat sdf = new SimpleDateFormat("dd MMMM YYYY");

public void walk(String path, FileFilter dd) {

    File root = new File(path);
    File[] list = root.listFiles();

    if (list == null) {
        return;
    }

    for (File f : list) {
        if (f.isDirectory()) {
            walk(f.getAbsolutePath(), dd);
            getHasil(f.getAbsolutePath(), dd, false);
        }
    }
}

public void getHasil(String dirPath, FileFilter ff, boolean cek) {
    if (cek) {
        tModelHasil = new DefaultTableModel(headerHasil, 0);
    }

    File dir = new File(dirPath);
    File[] files = dir.listFiles(ff);
    try {
        for (File aFile : files) {
            //System.out.println(aFile.getName() + " - " + aFile.length()+"-"+aFile.getAbsolutePath());
            String namafile = aFile.getName();
            String ext = FilenameUtils.getExtension(namafile);
            if (namafile.length() > (50 - ext.length())) {
                namafile = namafile.substring(0, (50 - ext.length())) + "...." + ext;
            }
            Path file = Paths.get(aFile.getAbsolutePath());
            BasicFileAttributes attr = null;
            try {
                attr = Files.readAttributes(file, BasicFileAttributes.class);
            } catch (IOException ex) {
                ex.printStackTrace();
            }
            Calendar cal = Calendar.getInstance();
            cal.setTimeInMillis(attr.creationTime().toMillis());
            String data[] = {namafile, String.valueOf(aFile.length()), sdf2.format(cal.getTime()),
                sdf2.format(new Date(aFile.lastModified())), aFile.getPath()};
            tModelHasil.addRow(data);
```

```
        }
    } catch (Exception ex) {
        System.out.println("Folder : " + dirPath + " Hasil = " + files);
    }
    labelJumlahFile.setText(String.valueOf(tModelHasil.getRowCount()));
    tabelHasil.setModel(tModelHasil);
    AturLebarKolom alk = new AturLebarKolom(tabelHasil);
    alk.adjustColumns();

}

int perbandingan = 1;
int perbandingantgl = 1;

FileFilter filterSize = new FileFilter() {
    public boolean accept(File file) {
        if (file.isFile()) {
            if (file.length() == size && perbandingan == 1) {
                return true;
            } else if (file.length() < size && perbandingan == 2) {
                return true;
            } else if (file.length() > size && perbandingan == 3) {
                return true;
            } else if (file.length() <= size && perbandingan == 4) {
                return true;
            } else if (file.length() >= size && perbandingan == 5) {
                return true;
            } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }
};

FileFilter filterTgl = new FileFilter() {
    public boolean accept(File file) {
        if (file.isFile()) {
            if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                return true;
            } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                return true;
            } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                return true;
            } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                return true;
            } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                return true;
            } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }
};

FileFilter filterFileType = new FileFilter() {
    @Override
    public boolean accept(File file) {
        if (file.isFile()) {
            return getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim());
        } else {
            return false;
        }
    }
};

FileFilter filterAuthor = new FileFilter() {

    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    return true;
                } else {
                    return false;
                }
```

```
            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }
    }
};

FileFilter filterOwner = new FileFilter() {

    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                    return true;
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }
    }
};

FileFilter filterAuthorSize = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (file.length() == size && perbandingan == 1) {
                        return true;
                    } else if (file.length() < size && perbandingan == 2) {
                        return true;
                    } else if (file.length() > size && perbandingan == 3) {
                        return true;
                    } else if (file.length() <= size && perbandingan == 4) {
                        return true;
                    } else if (file.length() >= size && perbandingan == 5) {
                        return true;
                    } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterAuthorSizeFileType = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                        if (file.length() == size && perbandingan == 1) {
                            return true;
                        } else if (file.length() < size && perbandingan == 2) {
                            return true;
                        } else if (file.length() > size && perbandingan == 3) {
                            return true;
```

```
                } else if (file.length() <= size && perbandingan == 4) {
                    return true;
                } else if (file.length() >= size && perbandingan == 5) {
                    return true;
                } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                    return true;
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } else {
            return false;
        }

    } else {
        return false;
    }
} catch (IOException ex) {
    return false;
}

  }
};
FileFilter filterAuthorSizeFileTypeOwner = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
            if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
              if (file.length() == size && perbandingan == 1) {
                return true;
              } else if (file.length() < size && perbandingan == 2) {
                return true;
              } else if (file.length() > size && perbandingan == 3) {
                return true;
              } else if (file.length() <= size && perbandingan == 4) {
                return true;
              } else if (file.length() >= size && perbandingan == 5) {
                return true;
              } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                return true;
              } else {
                return false;
              }
            } else {
              return false;
            }
          } else {
            return false;
          }
        } else {
          return false;
        }

      } else {
        return false;
      }
    } catch (IOException ex) {
      return false;
    }

  }
};
FileFilter filterAuthorSizeFileTypeOwnerTgl = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
            if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
              boolean lanjut = false;
              if (file.length() == size && perbandingan == 1) {
                lanjut = true;
              } else if (file.length() < size && perbandingan == 2) {
                lanjut = true;
              } else if (file.length() > size && perbandingan == 3) {
```

```
                        lanjut = true;
                    } else if (file.length() <= size && perbandingan == 4) {
                        lanjut = true;
                    } else if (file.length() >= size && perbandingan == 5) {
                        lanjut = true;
                    } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                        lanjut = true;
                    } else {
                        lanjut = false;
                    }
                    if (lanjut) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                            return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                            return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                            return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                            return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                            return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                            return true;
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }
            } else {
                return false;
            }

        } else {
            return false;
        }
    } catch (IOException ex) {
        return false;
    }

  }
};

FileFilter filterSizeOwnerTgl = new FileFilter() {
   public boolean accept(File file) {
      FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
      UserPrincipal owner = null;
      try {
         owner = ownerAttributeView.getOwner();
         String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
         if (file.isFile()) {

            if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
               boolean lanjut = false;
               if (file.length() == size && perbandingan == 1) {
                  lanjut = true;
               } else if (file.length() < size && perbandingan == 2) {
                  lanjut = true;
               } else if (file.length() > size && perbandingan == 3) {
                  lanjut = true;
               } else if (file.length() <= size && perbandingan == 4) {
                  lanjut = true;
               } else if (file.length() >= size && perbandingan == 5) {
                  lanjut = true;
               } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                  lanjut = true;
               } else {
                  lanjut = false;
               }
               if (lanjut) {
                  if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                     return true;
                  } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                     return true;
                  } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                     return true;
                  } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                     return true;
                  } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                     return true;
                  } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
```

```
                    return true;
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } else {
            return false;
        }

    } else {
        return false;
    }
} catch (IOException ex) {
    return false;
}

}
};
FileFilter filterSizeTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {

                boolean lanjut = false;
                if (file.length() == size && perbandingan == 1) {
                    lanjut = true;
                } else if (file.length() < size && perbandingan == 2) {
                    lanjut = true;
                } else if (file.length() > size && perbandingan == 3) {
                    lanjut = true;
                } else if (file.length() <= size && perbandingan == 4) {
                    lanjut = true;
                } else if (file.length() >= size && perbandingan == 5) {
                    lanjut = true;
                } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                    lanjut = true;
                } else {
                    lanjut = false;
                }
                if (lanjut) {
                    if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                        return true;
                    } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                        return true;
                    } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                        return true;
                    } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                        return true;
                    } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                        return true;
                    } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterSizeFileTypeTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {

                if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {

                    boolean lanjut = false;
                    if (file.length() == size && perbandingan == 1) {
```

```
                    lanjut = true;
                } else if (file.length() < size && perbandingan == 2) {
                    lanjut = true;
                } else if (file.length() > size && perbandingan == 3) {
                    lanjut = true;
                } else if (file.length() <= size && perbandingan == 4) {
                    lanjut = true;
                } else if (file.length() >= size && perbandingan == 5) {
                    lanjut = true;
                } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                    lanjut = true;
                } else {
                    lanjut = false;
                }
                if (lanjut) {
                    if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                        return true;
                    } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                        return true;
                    } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                        return true;
                    } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                        return true;
                    } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                        return true;
                    } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }

        } else {
            return false;
        }
    } catch (IOException ex) {
        return false;
    }

  }
};
FileFilter filterAuthorSizeOwnerTgl = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
            boolean lanjut = false;
            if (file.length() == size && perbandingan == 1) {
              lanjut = true;
            } else if (file.length() < size && perbandingan == 2) {
              lanjut = true;
            } else if (file.length() > size && perbandingan == 3) {
              lanjut = true;
            } else if (file.length() <= size && perbandingan == 4) {
              lanjut = true;
            } else if (file.length() >= size && perbandingan == 5) {
              lanjut = true;
            } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
              lanjut = true;
            } else {
              lanjut = false;
            }
            if (lanjut) {
              if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                return true;
              } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                return true;
              } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                return true;
              } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                return true;
              } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                return true;
              } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                return true;
              } else {
```

```
                        return false;
                    }
                } else {
                    return false;
                }
            } else {
                return false;
            }

        } else {
            return false;
        }

    } else {
        return false;
    }
} catch (IOException ex) {
    return false;
}

}
};

FileFilter filterAuthorSizeTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {

                    boolean lanjut = false;
                    if (file.length() == size && perbandingan == 1) {
                        lanjut = true;
                    } else if (file.length() < size && perbandingan == 2) {
                        lanjut = true;
                    } else if (file.length() > size && perbandingan == 3) {
                        lanjut = true;
                    } else if (file.length() <= size && perbandingan == 4) {
                        lanjut = true;
                    } else if (file.length() >= size && perbandingan == 5) {
                        lanjut = true;
                    } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                        lanjut = true;
                    } else {
                        lanjut = false;
                    }
                    if (lanjut) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                            return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                            return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                            return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                            return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                            return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                            return true;
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }

                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterAuthorSizeFileTypeTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
```

```
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
               if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                  if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                     boolean lanjut = false;
                     if (file.length() == size && perbandingan == 1) {
                        lanjut = true;
                     } else if (file.length() < size && perbandingan == 2) {
                        lanjut = true;
                     } else if (file.length() > size && perbandingan == 3) {
                        lanjut = true;
                     } else if (file.length() <= size && perbandingan == 4) {
                        lanjut = true;
                     } else if (file.length() >= size && perbandingan == 5) {
                        lanjut = true;
                     } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                        lanjut = true;
                     } else {
                        lanjut = false;
                     }
                     if (lanjut) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                           return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                           return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                           return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                           return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                           return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                           return true;
                        } else {
                           return false;
                        }
                     } else {
                        return false;
                     }

                  } else {
                     return false;
                  }
               } else {
                  return false;
               }

            } else {
               return false;
            }
         } catch (IOException ex) {
            return false;
         }

      }
   };
   FileFilter filterSizeFileTypeOwnerTgl = new FileFilter() {
      public boolean accept(File file) {
         FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
         UserPrincipal owner = null;
         try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {

               if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                  if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                     boolean lanjut = false;
                     if (file.length() == size && perbandingan == 1) {
                        lanjut = true;
                     } else if (file.length() < size && perbandingan == 2) {
                        lanjut = true;
                     } else if (file.length() > size && perbandingan == 3) {
                        lanjut = true;
                     } else if (file.length() <= size && perbandingan == 4) {
                        lanjut = true;
                     } else if (file.length() >= size && perbandingan == 5) {
                        lanjut = true;
                     } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                        lanjut = true;
                     } else {
                        lanjut = false;
                     }
                     if (lanjut) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                           return true;
```

```
                    } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                        return true;
                    } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                        return true;
                    } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                        return true;
                    } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                        return true;
                    } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } else {
            return false;
        }
    } catch (IOException ex) {
        return false;
    }

  }
};
FileFilter filterAuthorSizeOwner = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
            if (file.length() == size && perbandingan == 1) {
              return true;
            } else if (file.length() < size && perbandingan == 2) {
              return true;
            } else if (file.length() > size && perbandingan == 3) {
              return true;
            } else if (file.length() <= size && perbandingan == 4) {
              return true;
            } else if (file.length() >= size && perbandingan == 5) {
              return true;
            } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
              return true;
            } else {
              return false;
            }
          } else {
            return false;
          }
        } else {
          return false;
        }
      } else {
        return false;
      }
    } catch (IOException ex) {
      return false;
    }

  }
};
FileFilter filterSizeFileTypeOwner = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
          if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
            if (file.length() == size && perbandingan == 1) {
              return true;
            } else if (file.length() < size && perbandingan == 2) {
              return true;
            } else if (file.length() > size && perbandingan == 3) {
```

```
                    return true;
                } else if (file.length() <= size && perbandingan == 4) {
                    return true;
                } else if (file.length() >= size && perbandingan == 5) {
                    return true;
                } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
                    return true;
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } else {
            return false;
        }
    } catch (IOException ex) {
        return false;
    }

  }
};
FileFilter filterSizeFileType = new FileFilter() {
  @Override
  public boolean accept(File file) {
    if (file.isFile()) {
      if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
        if (file.length() == size && perbandingan == 1) {
          return true;
        } else if (file.length() < size && perbandingan == 2) {
          return true;
        } else if (file.length() > size && perbandingan == 3) {
          return true;
        } else if (file.length() <= size && perbandingan == 4) {
          return true;
        } else if (file.length() >= size && perbandingan == 5) {
          return true;
        } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
          return true;
        } else {
          return false;
        }
      } else {
        return false;
      }
    } else {
      return false;
    }
  }
};
FileFilter filterSizeOwner = new FileFilter() {
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {

        if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
          if (file.length() == size && perbandingan == 1) {
            return true;
          } else if (file.length() < size && perbandingan == 2) {
            return true;
          } else if (file.length() > size && perbandingan == 3) {
            return true;
          } else if (file.length() <= size && perbandingan == 4) {
            return true;
          } else if (file.length() >= size && perbandingan == 5) {
            return true;
          } else if ((file.length() >= sizeawal && file.length() <= sizeakhir) && perbandingan == 6) {
            return true;
          } else {
            return false;
          }
        } else {
          return false;
        }
      } else {
        return false;
      }

    } catch (IOException ex) {
      return false;
```

```
        }

    }
};

FileFilter filterAuthorTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        Path f = Paths.get(file.getAbsolutePath());
        try {
            BasicFileAttributes attr = Files.readAttributes(f, BasicFileAttributes.class);
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                        return true;
                    } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                        return true;
                    } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                        return true;
                    } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                        return true;
                    } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                        return true;
                    } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterAuthorFileTypeTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                            return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                            return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                            return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                            return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                            return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                            return true;
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterAuthorFileTypeOwnerTgl = new FileFilter() {
    public boolean accept(File file) {
```

```
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                        if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                            if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                                return true;
                            } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                                return true;
                            } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                                return true;
                            } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                                return true;
                            } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                                return true;
                            } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                                return true;
                            } else {
                                return false;
                            }
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterFileTypeOwnerTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {

                if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                    if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                            return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                            return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                            return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                            return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                            return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                            return true;
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }

                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterAuthorOwnerTgl = new FileFilter() {
```

```
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
                    if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                        if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                            return true;
                        } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                            return true;
                        } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                            return true;
                        } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                            return true;
                        } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                            return true;
                        } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                            return true;
                        } else {
                            return false;
                        }
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }

            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterOwnerTgl = new FileFilter() {
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtOwnerManual.getText().trim())) {
                    if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                        return true;
                    } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                        return true;
                    } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                        return true;
                    } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                        return true;
                    } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                        return true;
                    } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                        return true;
                    } else {
                        return false;
                    }
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }

    }
};
FileFilter filterFileTypeTgl = new FileFilter() {
    @Override
    public boolean accept(File file) {
        if (file.isFile()) {
            if (getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim())) {
                if (file.lastModified() == getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 1) {
                    return true;
                } else if (file.lastModified() < getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 2) {
                    return true;
                } else if (file.lastModified() > getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 3) {
                    return true;
```

```
            } else if (file.lastModified() <= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 4) {
                return true;
            } else if (file.lastModified() >= getTanggalterpilih(cmbFilterDateTime) && perbandingantgl == 5) {
                return true;
            } else if ((file.lastModified() >= tglawal && file.lastModified() <= tglakhir) && perbandingantgl == 7) {
                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    } else {
        return false;
    }
  }
};

FileFilter filterAuthorFileType = new FileFilter() {

  @Override
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          return getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim());
        } else {
          return false;
        }
      } else {
        return false;
      }
    } catch (IOException ex) {
      return false;
    }
  }
};
FileFilter filterAuthorFileTypeOwner = new FileFilter() {

  @Override
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (dd.equalsIgnoreCase(txtOwner.getText().trim())) {
            return getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim());
          } else {
            return false;
          }
        } else {
          return false;
        }
      } else {
        return false;
      }
    } catch (IOException ex) {
      return false;
    }
  }
};
FileFilter filterAuthorOwner = new FileFilter() {

  @Override
  public boolean accept(File file) {
    FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
    UserPrincipal owner = null;
    try {
      owner = ownerAttributeView.getOwner();
      String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
      if (file.isFile()) {
        if (dd.equalsIgnoreCase(txtAuthors.getText().trim())) {
          if (dd.equalsIgnoreCase(txtOwner.getText().trim())) {
            return true;
          } else {
            return false;
          }
        } else {
          return false;
        }
```

```
            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }
    }
};
FileFilter filterFileTypeOwner = new FileFilter() {

    @Override
    public boolean accept(File file) {
        FileOwnerAttributeView ownerAttributeView = Files.getFileAttributeView(file.toPath(), FileOwnerAttributeView.class);
        UserPrincipal owner = null;
        try {
            owner = ownerAttributeView.getOwner();
            String dd = owner.getName().substring(owner.getName().lastIndexOf("\\") + 1);
            if (file.isFile()) {
                if (dd.equalsIgnoreCase(txtOwner.getText().trim())) {
                    return getFileExtension(file).equalsIgnoreCase(txtTypeFile.getText().trim());
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } catch (IOException ex) {
            return false;
        }
    }
};

private String getFileExtension(File file) {
    String name = file.getName();
    try {
        return name.substring(name.lastIndexOf(".") + 1);
    } catch (Exception e) {
        return "";
    }
}

public byte[] createChecksum(String filename, String type) throws Exception {
    InputStream fis = new FileInputStream(filename);

    byte[] buffer = new byte[1024];
    MessageDigest complete = MessageDigest.getInstance(type);
    int numRead;

    do {
        numRead = fis.read(buffer);
        if (numRead > 0) {
            complete.update(buffer, 0, numRead);
        }
    } while (numRead != -1);

    fis.close();
    return complete.digest();
}

public String getChecksum(String filename, String type) {
    String result = "";
    byte[] b;
    try {
        b = createChecksum(filename, type);
        for (int i = 0; i < b.length; i++) {
            result += Integer.toString((b[i] & 0xff) + 0x100, 16).substring(1);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "GAGAL cek CheckSum");
    }

    return result;
}

private String getComputerName() {
    Map<String, String> env = System.getenv();
    if (env.containsKey("COMPUTERNAME")) {
        return env.get("COMPUTERNAME");
    } else if (env.containsKey("HOSTNAME")) {
        return env.get("HOSTNAME");
    } else {
        return "Unknown Computer";
    }
}

public void kosong() {
    txtAuthors.setText(null);
```

```
        txtComputer.setText(null);
        txtLokasi.setText(null);
        txtNamaFile2.setText(null);
        txtOwner.setText(null);
        txtPath.setText(null);
        txtSUM.setText(null);
        txtTypeFile.setText(null);
        txtAntaraAkhir.setText(null);
        txtAntaraAwal.setText(null);
        tModel = new DefaultTableModel(header, 0);
        tabelMeta.setModel(tModel);
        tModelHasil = new DefaultTableModel(headerHasil, 0);
        tabelHasil.setModel(tModelHasil);
    }

    private void disablekorelasi(boolean status) {
        txtKorelasiTglFile.setText("");
        txtKorelasiTglFile.setEnabled(status);
        rbSama.setEnabled(status);
        rbBesar.setEnabled(status);
        rbKecil.setEnabled(status);
        rbSamaBesar.setEnabled(status);
        rbSamaKecil.setEnabled(status);
        rbantara.setEnabled(status);
        //rbantaraDate.setEnabled(status);
        txtAntaraAwal.setEnabled(status);
        txtAntaraAkhir.setEnabled(status);
        //tglAnataraAkhir.setDate(new Date());
        //tglAntaraAwal.setDate(new Date());

        rbSama.setSelected(status);

    }

    private void disablekorelasiDate(boolean status) {
        tglAnataraAkhir1.setDate(new Date());
        tglAntaraAwal1.setDate(new Date());
        tglAnataraAkhir1.setEnabled(status);
        tglAntaraAwal1.setEnabled(status);
        txtKorelasiTglFile.setText("");
        txtKorelasiTglFile.setEnabled(status);
        rbSama1.setEnabled(status);
        rbBesar1.setEnabled(status);
        rbKecil1.setEnabled(status);
        rbSamaBesar1.setEnabled(status);
        rbSamaKecil1.setEnabled(status);
        rbantara1.setEnabled(status);
        rbSama1.setSelected(status);

    }

    public void simpanLog(File f, String content) {
        BufferedWriter bw = null;
        FileWriter fw = null;

        try {
            fw = new FileWriter(f.getAbsolutePath());
            bw = new BufferedWriter(fw);
            bw.write(content);
            JOptionPane.showMessageDialog(null, "Proses Simpan Log berhasil");
        } catch (IOException e) {
            e.printStackTrace();

        } finally {

            try {

                if (bw != null) {
                    bw.close();
                }

                if (fw != null) {
                    fw.close();
                } catch (IOException ex) {
                ex.printStackTrace();

                public String pilihRadioButton(ButtonGroup btngroup) {
        String nama = "";
        Enumeration<AbstractButton> allRadioButton = btngroup.getElements();
        while (allRadioButton.hasMoreElements()) {
            JRadioButton temp = (JRadioButton) allRadioButton.nextElement();
            if (temp.isSelected()) {
                nama = temp.getName();
            }
        return nama;
    }
}
```