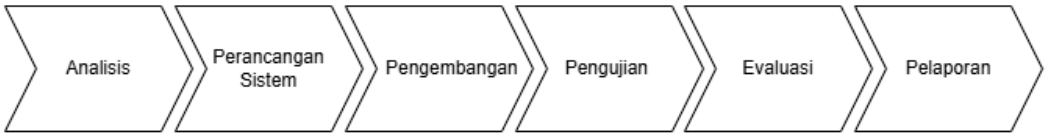
BAB III
METODE PENELITIAN

3.1. Metode Penelitian

Metode *Design and Development* (D&D) dipilih untuk diterapkan pada penelitian ini, karena pendekatan penelitian ini berfokus pada perancangan dan evaluasi aplikasi sebagai solusi terhadap masalah penelitian. Berdasarkan jurnal oleh (J. Ellis & Levy, 2010), metode ini terdiri dari enam tahap utama. Tahap pertama adalah analisis, di mana peneliti menentukan masalah yang layak untuk dikaji menggunakan metode D&D. Selanjutnya, tahap kedua adalah perancangan sistem pada penelitian yang berkaitan dengan permasalahan yang telah diidentifikasi. Tahap ketiga melibatkan pengembangan aplikasi, yang mencakup pembuatan konsep, desain arsitektur, serta sistem berdasarkan kajian literatur dan teori yang ada. Setelah aplikasi dikembangkan, tahap keempat adalah pengujian untuk mengevaluasi efektivitasnya dengan metode yang sesuai, seperti observasi, wawancara, atau eksperimen. Hasil dari pengujian kemudian dievaluasi dan dianalisis pada tahap kelima untuk menilai sejauh mana aplikasi dapat menyelesaikan masalah yang telah diidentifikasi. Terakhir, tahap keenam adalah komunikasi hasil penelitian dalam bentuk laporan atau publikasi ilmiah. Pendekatan ini tidak hanya menghasilkan produk atau model baru, tetapi juga berkontribusi terhadap pengembangan teori dan praktik yang ada. Oleh karena itu, metode D&D banyak digunakan dalam berbagai bidang, termasuk untuk pengembangan aplikasi pada penelitian ini yang menggunakan algoritma AES yang dikombinasikan dengan NoStega untuk mengamankan data teks. Desain penelitian D&D dapat dilihat pada Gambar 3.1.



Gambar 3.1 Tahapan Penelitian Metode *Design and Development*

3.2. Analisis

Tahap pertama dalam metode penelitian *Design and Development* (D&D) adalah analisis, yang diawali dengan studi literatur untuk memahami perkembangan ilmu dan hasil penelitian sebelumnya yang relevan dengan topik yang dikaji. Studi literatur ini bertujuan untuk mengidentifikasi masalah, merumuskan pertanyaan penelitian, serta menetapkan tujuan yang ingin dicapai. Selain itu, peneliti juga mempelajari berbagai pendekatan dan solusi yang telah diuji sebelumnya guna menemukan metode yang paling sesuai dalam pengembangan solusi yang diusulkan.

Dalam konteks penelitian ini, hasil studi literatur mengarahkan peneliti untuk mengembangkan aplikasi berbasis web yang bertujuan mengamankan data teks menggunakan algoritma kriptografi AES yang dikombinasikan dengan teknik *noiseless steganography*. Untuk mendukung pengembangan ini, dilakukan kajian mendalam terhadap algoritma AES guna memahami keunggulan dan kelemahannya dalam konteks keamanan data teks. Selain itu, teknik *noiseless steganography* diteliti sebagai metode penyembunyian data secara efisien tanpa menimbulkan kecurigaan. Penelitian ini juga mencakup eksplorasi terhadap pengembangan aplikasi berbasis web agar solusi yang dihasilkan dapat diimplementasikan secara optimal dan mudah diakses oleh pengguna. Melalui analisis ini, diperoleh pemahaman yang lebih komprehensif mengenai teknologi yang relevan dan cara mengintegrasikannya untuk menciptakan sistem keamanan data teks yang efektif dan andal.

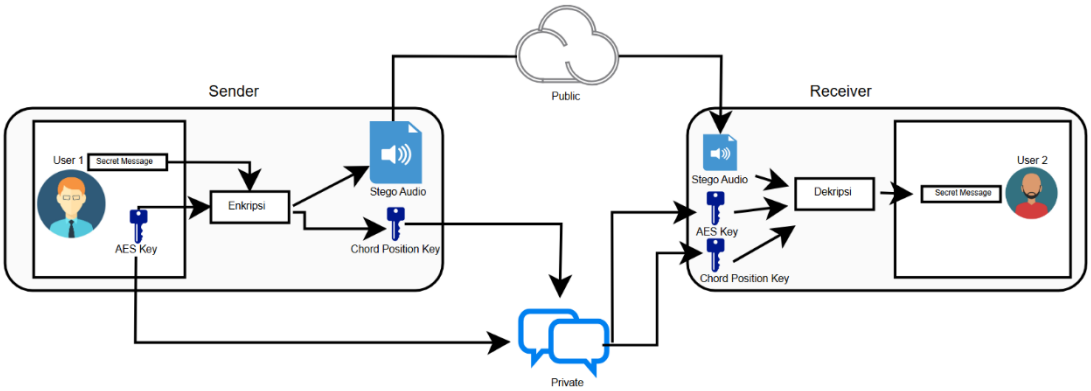
3.3. Perancangan Sistem

Tahap kedua dalam metode *Design and Development* (D&D) adalah perancangan sistem, di mana peneliti mengembangkan kerangka dan rancangan sebagai dasar dalam proses desain aplikasi. Pada tahap ini, dilakukan pemodelan sistem melalui berbagai diagram yang bertujuan untuk memvisualisasikan arsitektur sistem secara keseluruhan. Diagram yang dibuat mencakup *use case diagram* untuk menggambarkan interaksi antara pengguna dan sistem, serta *Flowchart diagram* yang menjelaskan alur kerja atau proses dalam sistem. Penggunaan diagram-diagram ini membantu membuat perancangan sistem lebih terstruktur dan mudah dipahami, sehingga mempermudah

pengembang dalam mengimplementasikan aplikasi sesuai dengan kebutuhan dan spesifikasi yang telah ditetapkan.

3.3.1. Diagram Arsitektur Sistem

Diagram arsitektur sistem berikut menggambarkan sistem aplikasi berbasis web yang akan dikembangkan secara keseluruhan. Gambar 3.2 menunjukkan diagram dengan komponen-komponen utama yang ada pada aplikasi dan interaksi antar komponen tersebut terjadi. Proses pengiriman pesan dimulai dengan *User* 1, sebagai pengirim yang menyiapkan sebuah pesan teks dalam bentuk *input teks*. Pesan teks dienkripsi menggunakan algoritma AES dan pengirim menginput kunci untuk mengunci hasil enkripsi. Hasil dari proses tersebut adalah *ciphertext* yang sudah diubah menjadi bilangan biner yang setelahnya dikonversi menjadi *file* audio stego, dan proses ini juga menghasilkan kunci baru yang disebut *chord position key*. *File* audio stego ini kemudian dikirim ke *User* 2 melalui saluran publik. *User* 2 menggunakan kedua kunci yang sama yang diinput oleh *User* 1 yang didapat melalui saluran pribadi antar *user*. Kedua kunci digunakan untuk mendekripsi *file* audio stego dan mendapatkan kembali pesan teks asli. Berikut Gambar 3.2 menampilkan diagram arsitektur sistem yang digunakan.



Gambar 3.2 Diagram Arsitektur Sistem

3.3.2. *Flowchart* Sistem

Proses enkripsi dimulai dengan pengambilan data teks dari *input* teks yang diunggah pengguna bersama dengan kunci yang di*input* untuk proses enkripsi. Proses ini mencakup ekstraksi pesan teks yang didapat dari *input* pesan yang dilakukan oleh

user yang diunggah lalu data teks diproses oleh algoritma AES. Hasil enkripsi AES yang berbentuk heksadesimal diubah menjadi bilangan biner. Setelahnya, hasil enkripsi yang sudah diubah menjadi bilangan biner, karakter dari pesan teks yang akan diproses harus diubah menjadi bentuk biner 8-bit.

Contoh:

Pesan Rahasia: “Tekkom 2025”

Karakter pada pesan rahasia diubah ke dalam bentuk biner

T = 0101 0100

e = 0110 0101

k = 0110 1011

k = 0110 1011

o = 0110 1111

m = 0110 1101

spasi = 0010 0000

2 = 0011 0010

0 = 0011 0000

2 = 0011 0010

5 = 0011 0101

Kemudian kode biner yang telah didapat, dipecah menjadi kode biner 2-bit seperti berikut:

01 01 01 00 01 10 01 01 01 10 10 11 01 10 10 11 01 10 11 11 01 10 11 01 00 10 00 00 00 11 00 10 00 11 00 00 00 11 00 10 00 11 01 01.

Setiap karakter dipecah menjadi dua bagian 2-bit. Misalnya, karakter T (0101 0100) dipecah menjadi “01”, “01”, “01”, “00”. Proses ini diulang untuk setiap karakter hingga seluruh teks "Tekkom 2025" terkonversi ke dalam bentuk 2-bit. Kemudian setiap pecahan 2-bit dikonversikan ke dalam bentuk audio.

Pada proses ini, sudah disiapkan terlebih dahulu untuk *file* audio untuk setiap kombinasi dari bilangan biner yang sudah dipisah menjadi masing-masing 2-bit, contoh *file* audio yang sudah disiapkan sebagai berikut:

2-bit ‘00’: C-PianoChord.wav

2-bit ‘01’: D-PianoChord.wav

2-bit ‘10’: E-PianoChord.wav

2-bit ‘11’: G-PianoChord.wav

Setelah proses konversi biner selesai, sistem melakukan proses peng-*assign*-an urutan chord pada melodi lagu "Mary Had a Little Lamb" yang telah didefinisikan sebelumnya. Proses ini melibatkan pemetaan setiap segmen biner 2-bit ke posisi nada tertentu dalam pola melodi yang telah ditentukan.

Proses Peng-*assign*-an Urutan Chord:

1. Pemetaan Biner ke Nada:

   Setiap segmen biner 2-bit dipetakan ke nada piano yang sesuai:

   00 → Nada C (C-PianoChord.wav)

   01 → Nada D (D-PianoChord.wav)

   10 → Nada E (E-PianoChord.wav)

   11 → Nada G (G-PianoChord.wav)

2. Pola Melodi "Mary Had a Little Lamb":

   Sistem menggunakan pola melodi yang telah didefinisikan dengan 26 nada. Pola dan durasi tiap Chord dapat terlihat pada Gambar 3.3 berikut.

```
# Definisi pola lagu Mary Had a Little Lamb dengan nada dan ketukan
mary_had_a_little_lamb = [
    ("E", 1.6), ("D", 0.5), ("C", 1.0), ("D", 1.0), ("E", 1), ("E", 1.0), ("E", 2.2),
    ("D", 1.0), ("D", 1.0), ("D", 2.4), ("E", 1.0), ("G", 1.0), ("G", 2.2),
    ("E", 1.6), ("D", 0.5), ("C", 1.0), ("D", 1.0), ("E", 1), ("E", 1.0), ("E", 1.0),
    ("C", 1.1), ("D", 1.1), ("D", 1.1), ("E", 1.1), ("D", 1.1), ("C", 2.2)
]
```

Gambar 3.3 Kode program definisi melodi dan panjang tiap chord

Berdasarkan Gambar 3.3, dalam implementasi sistem *noiseless steganography* audio ini, durasi setiap chord diukur menggunakan satuan unit yang kemudian dikonversi menjadi detik. Sistem menggunakan konversi 1 unit = 0.5 detik. Hal ini dapat dilihat dari perhitungan durasi yang menggunakan rumus *beat_duration = int(sample_rate * 0.5 * duration)*, dimana setiap unit durasi dikalikan dengan 0.5 untuk menghasilkan durasi dalam detik. Implementasi dapat dilihat pada Gambar 3.4.

```
# Hitung durasi berdasarkan ketukan (1.0 = 0.5 detik)
beat_duration = int(sample_rate * 0.5 * duration)
```

Gambar 3.4 Kode program mendefinisikan unit ke durasi satuan detik

3. Proses Penempatan Chord
   a. Sistem mencari posisi dalam pola melodi yang sesuai dengan nada yang dipetakan dari segmen biner
   b. Jika posisi yang sesuai ditemukan dan belum digunakan, sistem menempatkan chord di posisi tersebut
   c. Jika semua posisi dalam satu perulangan sudah terisi, sistem membuat perulangan baru dari pola melodi
   d. Setiap perulangan dipisahkan dengan jeda (*Pause*) selama 2 detik (4.0 unit).

Contoh Proses Peng-assign-an:

Untuk pesan "Tekkom 2025" yang telah dikonversi menjadi biner:

01 01 01 00 01 10 01 01 01 10 10 11 01 10 10 11 01 10 11 11 01 10 11 01 00 10 00 00 00 11 00 10 00 11 00 00 00 11 00 10 00 11 01 01

Sistem akan:

1. Mengambil segmen pertama 01 → mencari posisi nada D dalam pola melodi
2. Menempatkan chord D di posisi yang sesuai (misalnya posisi 1)
3. Mencatat posisi 1 sebagai bagian dari kunci kedua
4. Melanjutkan dengan segmen berikutnya hingga semua segmen biner terpenuhi

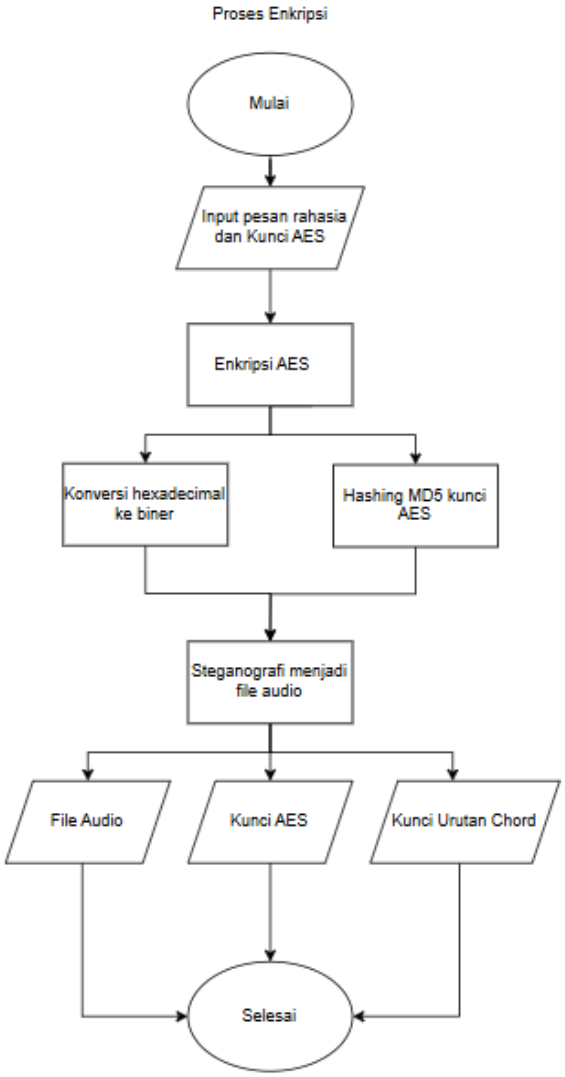
Setelah proses peng-*assign*-an selesai, sistem menghasilkan kunci kedua berupa urutan posisi chord yang dipisahkan dengan koma. Contoh output:

1,3,2,0,15,1,12,4,5,20,3,25,6,38,7,29,10,42,8,47,39,65,13,66,9,52,92,14,17,93,119,16,56,69,120,18,146,19,21,147,74,173,23,79,27,83,96,101,174,31,200,106,201,32,33,227,22,110,123,24,28,228,37,128,133,137,30,34,40,35,150,155,44,254,36,45,46,41,160,50,255,43,54,281,282,164,58,177,59,48,308,182,187,60,191,64,49,309,51,335,204,55,67,57,209,336,214,218,71,231,362,363,72,236,389,61,73,241,245,258,263,77,62,81,85,390,86,268,272

Kunci kedua ini berfungsi sebagai kunci untuk mengakses data tersembunyi dalam file audio steganografi. Tanpa pengetahuan tentang urutan posisi chord yang tepat, pesan rahasia tidak dapat diekstrak dari *file* audio, sehingga menambah lapisan keamanan pada sistem steganografi ini.

Selain kunci kedua berupa urutan chord, sistem juga mengimplementasikan mekanisme keamanan tambahan dengan menyimpan *hash* MD5 dari kunci AES dalam *metadata file* audio. Proses ini dilakukan dengan meng-*hash* kunci AES menggunakan algoritma MD5, kemudian menyimpan hasil *hash* tersebut dalam *chunk metadata* WAV *file*. Pada saat dekripsi, sistem akan membandingkan *hash* MD5 dari kunci AES yang diinputkan pengguna dengan *hash* yang tersimpan dalam metadata *file* audio. Jika *hash* tidak cocok, proses dekripsi akan dihentikan dan menampilkan pesan error. Mekanisme ini memastikan bahwa hanya pengguna yang memiliki kunci AES yang benar yang dapat mengakses pesan rahasia tersembunyi dalam *file* audio steganografi.

Setiap 2-bit kombinasi biner akan direprentasikan menjadi *chord* piano yang akan digabungkan menjadi satu *file* audio hasil dari proses konversi dengan konsep *noiseless steganography,* sehingga menghasilkan *file* akhir yang disebut stego audio. Stego audio ini berfungsi sebagai media penyimpanan pesan teks tersembunyi. Setelah proses ini selesai dan sistem mencapai titik akhir, maka menandakan bahwa proses enkripsi dan steganografi telah berhasil dilakukan. Gambar 3.5 berikut menampilkan detail dari *flowchart* proses enkripsi pada aplikasi.



Gambar 3.5 *Flowchart* sistem enkripsi

Proses dekripsi pada aplikasi ini dimulai dengan penginputan tiga sumber utama, yaitu stego audio yang diunggah oleh pengguna kunci AES, dan kunci urutan posisi chord yang didapat dari pengguna yang melakukan enkripsi pada *file* tersebut. Proses ini meliputi ekstraksi bilangan biner yang didapat dari *file* stego audio, lalu bilangan biner tersebut diubah kembali menjadi heksadesimal.

Sebelum melakukan ekstraksi data, sistem melakukan verifikasi keamanan dengan membaca metadata file audio yang berisi *hash* MD5 dari kunci AES yang digunakan saat enkripsi. Sistem kemudian menghitung hash MD5 dari kunci AES yang diinputkan pengguna dan membandingkannya dengan *hash* yang tersimpan dalam

metadata. Jika *hash* tidak cocok, sistem akan menampilkan pesan error dan menghentikan proses dekripsi. Mekanisme ini memastikan bahwa hanya pengguna yang memiliki password AES yang benar yang dapat mengakses data tersembunyi.

Setelah verifikasi berhasil, sistem memproses stego audio untuk mengekstrak rangkaian chord piano yang mewakili kombinasi 2-bit biner. Setiap chord piano diidentifikasi dan dikonversi kembali ke bentuk 2-bit biner berdasarkan mapping yang telah ditentukan. Proses ekstraksi ini menggunakan kunci kedua berupa urutan posisi chord yang spesifik untuk file audio tersebut. Sistem mengikuti urutan posisi chord yang diberikan untuk mengekstrak data biner dari posisi yang tepat dalam pola melodi "Mary Had a Little Lamb".

Contoh Proses Ekstraksi:

Jika stego audio mengandung chord C-PianoChord.wav pada posisi tertentu, maka sistem akan mengonversinya ke biner 00. Namun, yang lebih penting adalah sistem menggunakan urutan posisi chord yang unik untuk setiap file audio. Misalnya, jika password urutan chord adalah "1,3,2,0,15", maka sistem akan mengekstrak chord dari posisi 1, 3, 2, 0, dan 15 dalam pola melodi, kemudian mengkonversi setiap chord tersebut ke representasi biner 2-bit.

1. Rekonstruksi Data

   Proses selanjutnya, setiap dua kombinasi 2-bit digabungkan kembali menjadi 8-bit (1 byte).

   01 + 01 + 01 + 00 = 01010100 (huruf T). Lalu setiap 8-bit biner dikonversi ke bentuk heksadesimal.

   01010100 (biner) = 54 (heksadesimal).

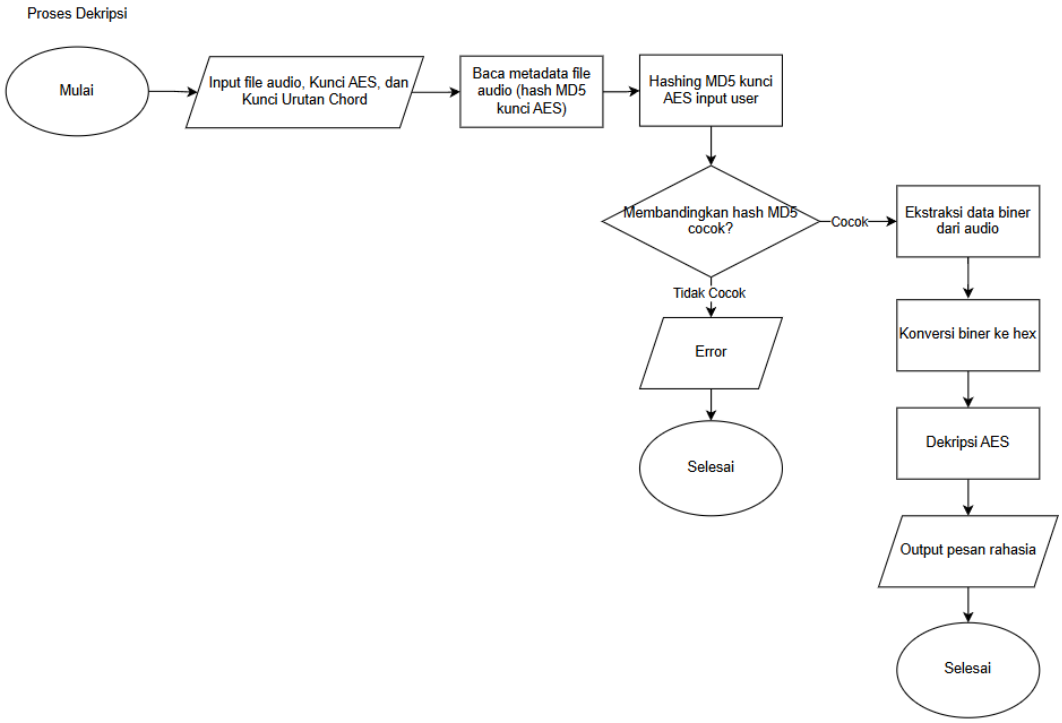
   Sistem melanjutkan proses ini hingga semua posisi chord dalam kunci urutan chord telah diproses dan data biner lengkap telah direkonstruksi.

2. Dekripsi AES dan Pemulihan Pesan

   Setelah data biner berhasil direkonstruksi menjadi string heksadesimal, proses dekripsi AES dijalankan menggunakan kunci AES yang telah diverifikasi sebelumnya. Algoritma AES akan mengembalikan data asli dalam bentuk teks dalam format *string* pesan teks. Sistem juga melakukan penghapusan

padding PKCS#7 dari proses AES yang ditambahkan saat enkripsi untuk mendapatkan pesan asli yang utuh.

Sistem ini mengimplementasikan keamanan berlapis dengan dua kunci yang berbeda. Kunci pertama (AES) diverifikasi melalui *hash* MD5 yang tersimpan dalam *metadata file* audio, sementara Kunci kedua (urutan posisi chord) spesifik untuk setiap *file* audio dan tidak dapat digunakan untuk mengakses data dari *file* audio lainnya. Hal ini memastikan bahwa setiap *file* audio steganografi memiliki karakteristik keamanan yang unik dan tidak dapat diakses tanpa kedua kunci yang benar. Setelah file teks berhasil direkontruksi, menandakan sistem telah mencapai titik akhir dan proses dekripsi telah dijalankan dengan sukses. Diagram *Flowchart* ini menjelaskan tahapan dekripsi dari pengambilan data tersembunyi dalam stego audio hingga proses pemulihan pesan teks asli, berikut ilustrasi proses dekripsi dalam Gambar 3.6.



Gambar 3.6 *Flowchart* sistem dekripsi

3.3.3. Diagram *Use Case*

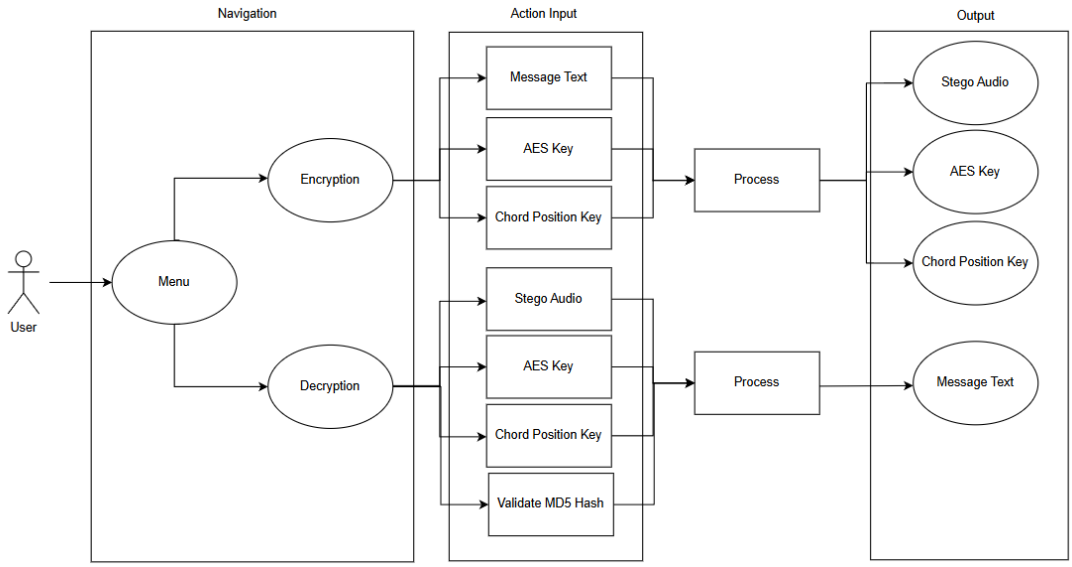
Diagram *Use Case* pada Gambar 3.7 menggambarkan cara pengguna berinteraksi dengan aplikasi, mencakup seluruh proses yang dilakukan untuk enkripsi dan dekripsi

data. Diagram ini dibagi menjadi tiga komponen utama: Navigasi, Input Aksi, dan Output. Bagian Navigasi menunjukkan cara pengguna mengakses aplikasi melalui menu utama, di mana mereka memiliki dua pilihan utama. Pilihan pertama adalah Enkripsi, yang memungkinkan pengguna untuk mengenkripsi pesan dalam bentuk pesan teks dan menyematkannya menjadi *file* audio. Pilihan kedua adalah Dekripsi, yang memungkinkan pengguna untuk mengambil pesan tersembunyi dalam *file* audio yang telah dienkripsi.

Pada bagian input, pada diagram dirinci input apa saja yang diperlukan untuk setiap proses. Untuk proses enkripsi, input yang dibutuhkan meliputi pesan teks (*Message Text*) yang akan disembunyikan, Kunci AES *(AES Key)*, dan Kunci urutan posisi chord *(Position Chord Key)* yang digunakan dalam proses enkripsi. Sementara itu, untuk dekripsi, input yang diperlukan adalah audio steganografi (*Stego Audio*) yang berisi pesan tersembunyi, Kunci AES *(AES Key)*, Kunci urutan posisi chord *(Position Chord Key,* dan juga proses memvalidasi Kunci AES yang sudah diproses *hash* menggunakan MD5 dengan metadata yang ada pada *stego audio*, yang digunakan untuk mendekripsi pesan tersebut.

Pada bagian output, hasil yang diperoleh setelah proses enkripsi yaitu audio steganografi (*Stego Audio*) yang berisi pesan teks terenkripsi. Sedangkan untuk proses dekripsi, output yang dihasilkan adalah teks pesan (*Message Text*) yang telah berhasil didekripsi dari *stego audio*.

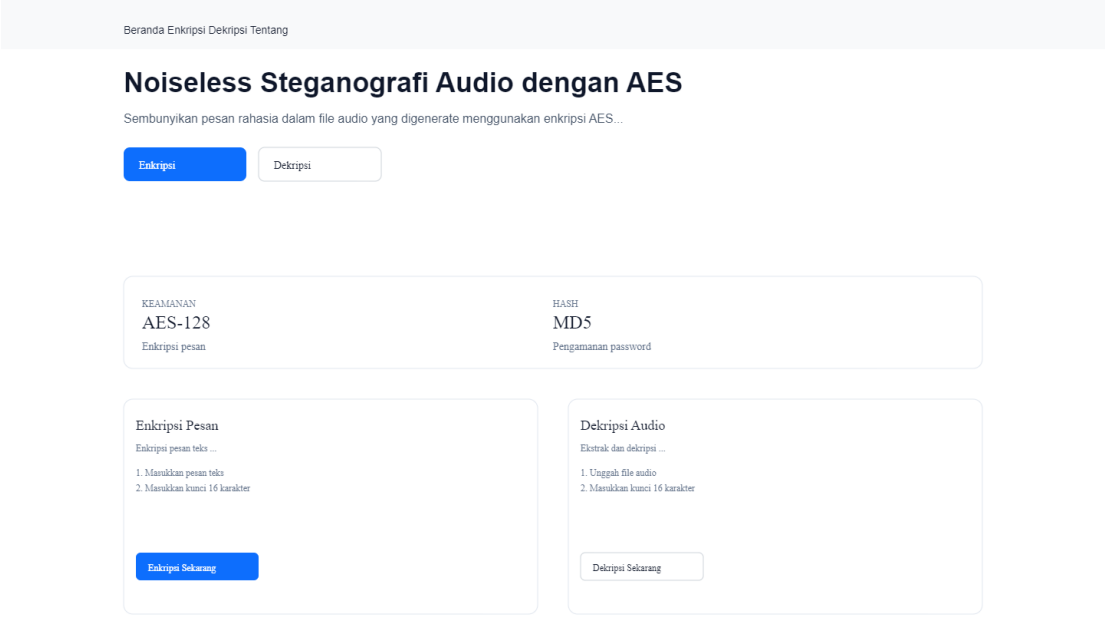
Penjelasan ini memberikan gambaran lengkap mengenai struktur dan fungsionalitas sistem, termasuk alur interaksi yang dilakukan pengguna untuk melakukan tujuan enkripsi dan dekripsi. Diagram *Use Case* pada Gambar 3.7 akan menampilkan visualisasi interaksi antara semua elemen tersebut.



Gambar 3.7 Diagram *Use Case*

3.3.4. Desain *Website*

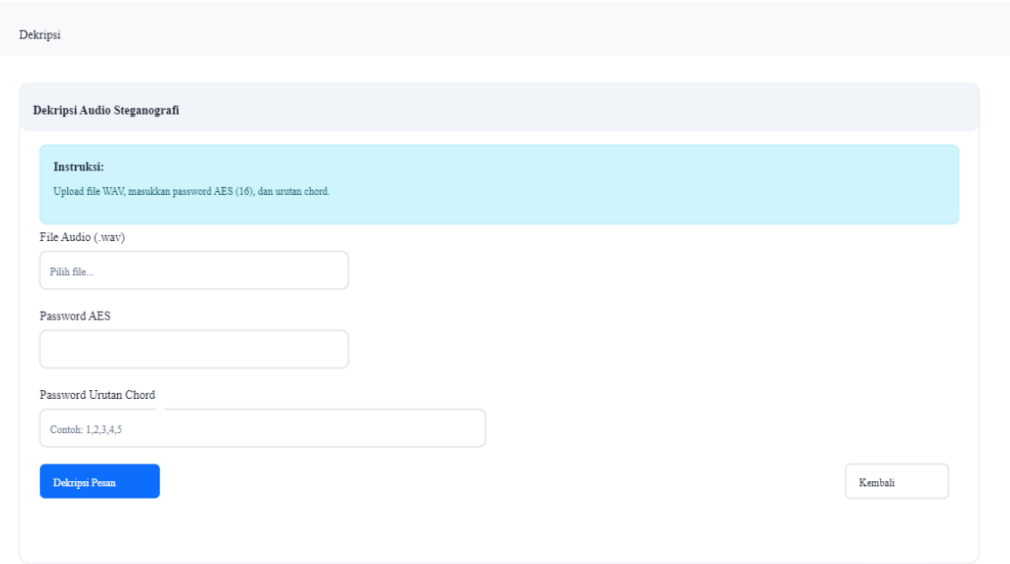
Desain *Website* Halaman beranda menjadi tampilan utama aplikasi, mengenalkan konsep noiseless steganography audio dan enkripsi AES, serta tombol aksi menuju fitur Enkripsi dan Dekripsi, terlihat pada Gambar 3.8.



Gambar 3.8 Desain Menu Utama

Terlihat pada Gambar 3.8, ditampilkan judul utama dan deskripsi singkat. Dua tombol aksi “Enkripsi” dan “Dekripsi” membantu pengguna langsung memulai. Di bawahnya, terdapat informasi yang menyorot konsep enkripsi, AES-128 untuk keamanan dan MD5 untuk verifikasi password. Terakhir, bagian penjelasan secara singkat untuk enkripsi dan dekripsi, disertai tombol navigasi untuk efisiensi dan kenyamanan pengguna.

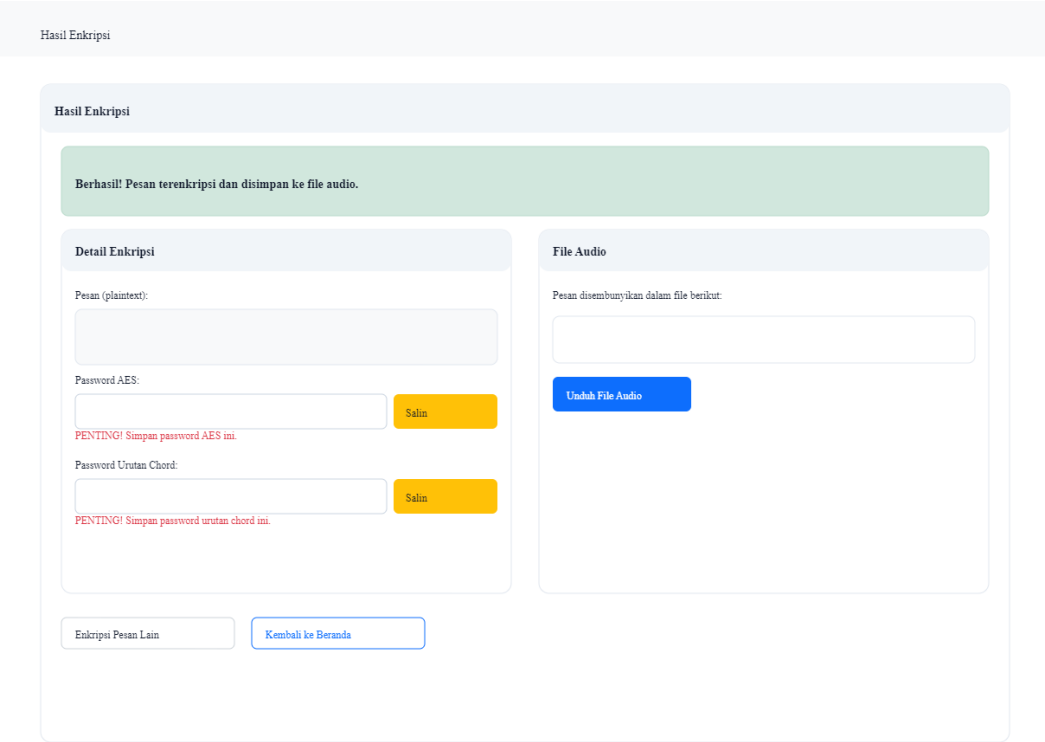
Selanjutnya pada Gambar 3.9, Halaman enkripsi dirancang untuk memandu pengguna mengubah pesan teks menjadi ciphertext dan menyisipkannya ke audio.



Gambar 3.9 Desain Menu Enkripsi

Pada Gambar 3.9 untuk menu enkripsi, bagian atas ditampilkan area progress yang muncul saat proses berjalan dan menampilkan persentase, estimasi waktu, dan deskripsi proses. *Form* dibagi dua kolom: kiri untuk input pesan dengan penghitung karakter, kanan untuk kunci enkripsi 16 karakter beserta catatan pengingat. Di bagian bawah, terdapat tombol “Kembali” dan tombol utama “Enkripsi” yang memulai proses.

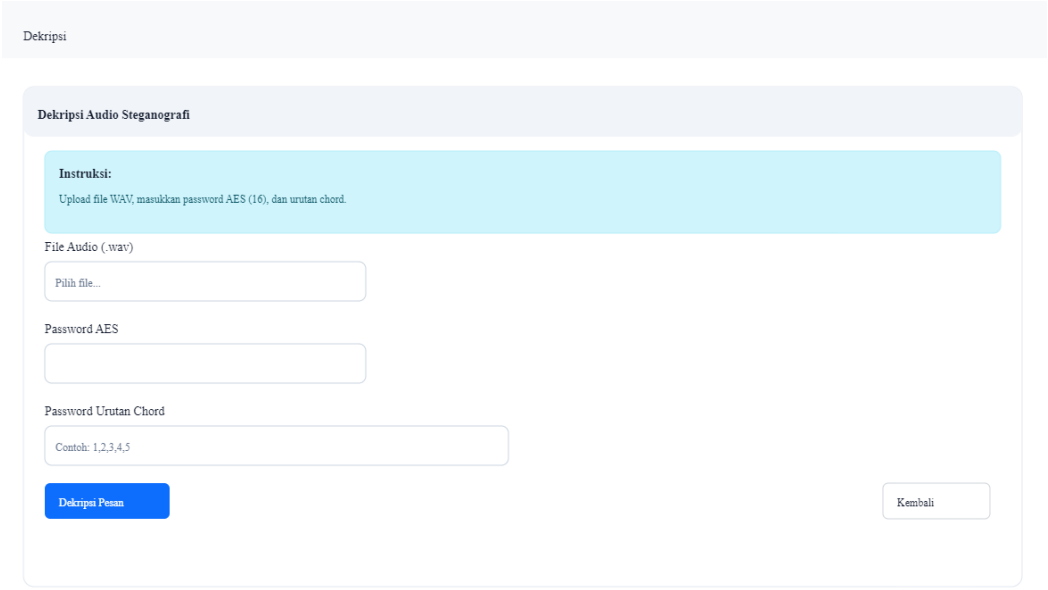
Halaman hasil enkripsi menampilkan konfirmasi keberhasilan, file stego audio, serta seluruh informasi kunci yang dibutuhkan untuk dekripsi, Terlihat pada gambar 3.10.



Gambar 3.10 Desain Menu Hasil Enkripsi

Terlihat pada Gambar 3.10, notifikasi sukses tampil di bagian atas. Kolom kiri berisi detail enkripsi: cuplikan plaintext, “Password AES” dengan tombol salin, dan “Password Urutan Chord” (chord positions) juga dengan tombol salin—keduanya ditandai sebagai informasi penting untuk disimpan. Kolom kanan berfokus pada file audio yang dihasilkan dan tombol unduh. Di bagian bawah, ada tautan cepat untuk melakukan enkripsi ulang atau kembali ke beranda.

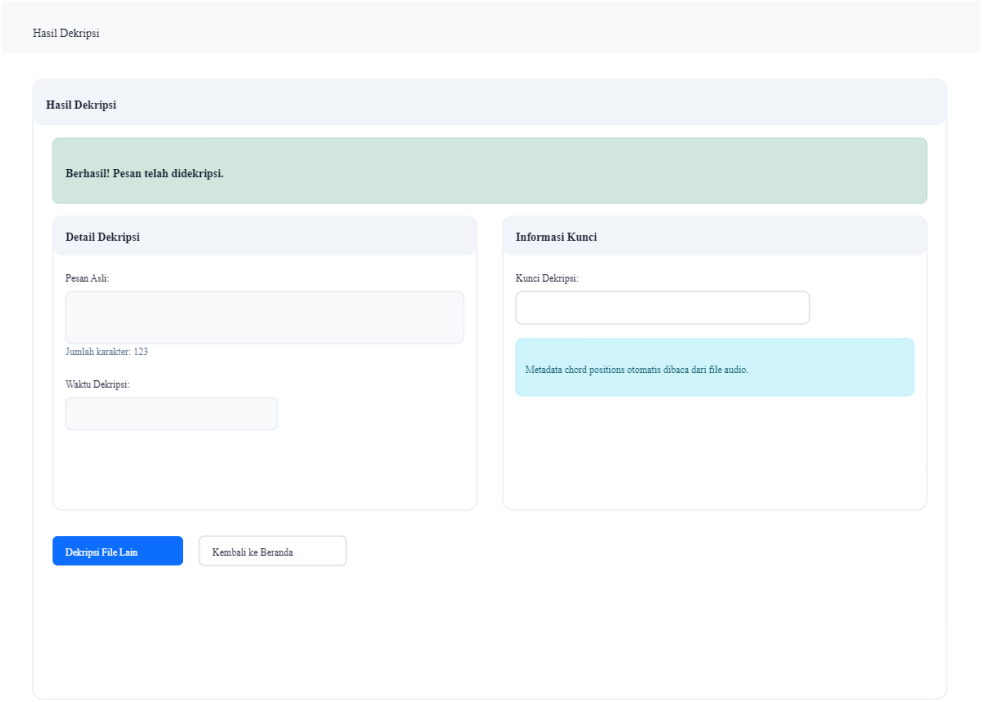
Selanjutnya, halaman dekripsi pada Gambar 3.11 menyediakan antarmuka untuk mengunggah file stego audio dan memasukkan dua kunci yang diperlukan agar pesan dapat diekstrak dan didekripsi.



Gambar 3.11 Desain Menu Dekripsi

Bagian instruksi *header* memperjelas kebutuhan input untuk pengguna: file WAV, password AES 16 karakter, dan "Password Urutan Chord". *Form* memuat tiga *field* sesuai urutan proses, masukkan password AES, lalu password urutan chord. Tombol utama "Dekripsi Pesan" berada di bawah *form*, diikuti tombol "Kembali" untuk navigasi.

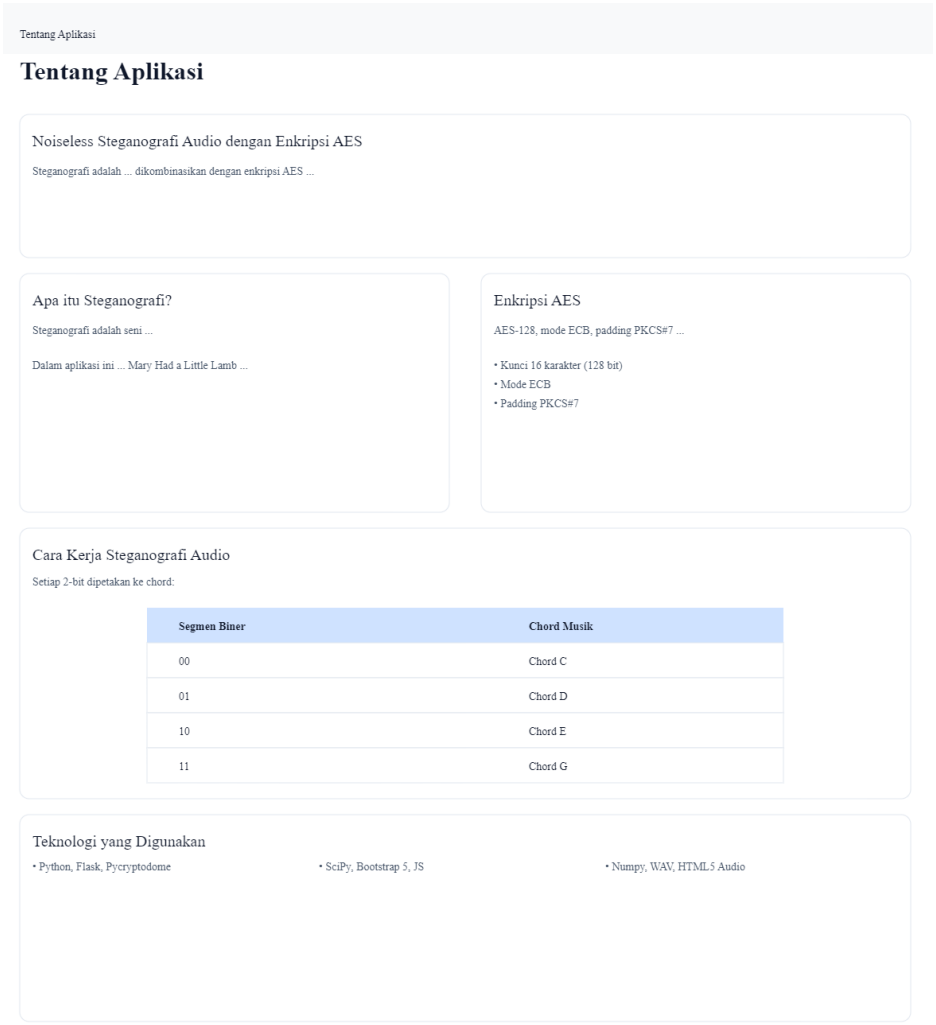
Halaman hasil dekripsi pada Gambar 3.12 menampilkan pesan asli yang berhasil diekstrak dari audio, beserta informasi waktu dekripsi dan kunci yang digunakan.



Gambar 3.12 Desain Menu Hasil Dekripsi

Terlihat pada Gambar 3.12, notifikasi sukses ditampilkan di atas. Kolom kiri menampilkan "Pesan Asli" (plaintext) dan jumlah karakter, serta *timestamp* waktu dekripsi. Kolom kanan memuat "Informasi Kunci" yang dipakai saat proses. Catatan informatif menjelaskan bahwa metadata posisi chord dibaca/diolah untuk rekonstruksi data. Tersedia tombol untuk mendekripsi file lain atau kembali ke beranda.

Pada Gambar 3.13, halaman tentang memberikan konteks, tujuan, serta ringkasan teknis dari metode *noiseless steganography* audio yang dikombinasikan dengan enkripsi AES.



Gambar 3.13 Desain Menu Tentang

Terlihat pada Gambar 3.13, bagian pembuka menjelaskan gambaran umum sistem pada web. Dua kolom konten menjelaskan definisi steganografi beserta penggunaan pola “Mary Had a Little Lamb”, dan ringkasan AES-128. Di bagian tengah, tabel pemetaan 2-bit ke chord memperlihatkan mekanisme konversi biner ke nada. Terakhir, daftar teknologi yang digunakan sebagai ringkasan *stack* yang digunakan pada web.

3.4. Pengembangan

Tahap ketiga dalam penelitian ini merupakan tahap pengembangan aplikasi. Pada tahap ini, penulis akan menguraikan metode yang digunakan dalam pengembangan

aplikasi serta menjelaskan alat dan bahan yang digunakan selama proses pengembangan.

3.4.1. Alat Penelitian

Pada penelitian ini, berikut berupa alat yang terbagi menjadi perangkat keras dan perangkat lunak yang digunakan selama proses penelitian berlangsung:

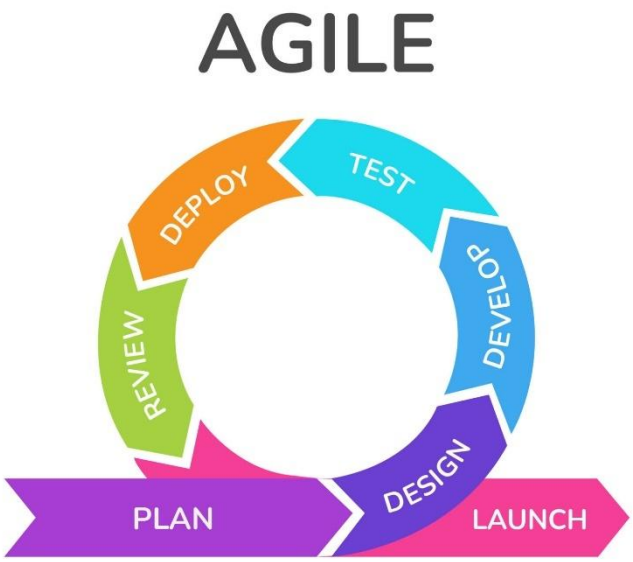
1. Perangkat Keras
   a. Laptop dengan spesifikasi prosesor AMD Ryzen 5 3500U, RAM 12 GB, dan SSD 512 GB.
2. Perangkat Lunak
   a. Microsoft Windows 11
   b. Visual Studio Code
   c. Google Chrome
   d. Draw.io

3.4.2. Metode Pengembangan Aplikasi

Pada penelitian ini, pengembangan aplikasi metode *Agile* digunakan pada proses pengembangan aplikasi. Metode *Agile* adalah pendekatan pengembangan perangkat lunak yang berfokus pada prinsip-prinsip Pengembangan sistem dengan waktu yang efisien, dengan menekankan interaksi responsif terhadap perubahan dalam berbagai bentuk pada aplikasi. Tahapan yang diterapkan dalam Metode *Agile* meliputi perencanaan (*planning*), implementasi (*Implementation)*, pengujian (*testing*), dokumentasi (*documentation)*, peluncuran (deployment), dan pemeliharaan (maintenance) (Handayani dkk., 2023).

Penulis mengadopsi metode *Agile* dalam pengembangan aplikasi web yang menggunakan algoritma AES serta teknik NoStega untuk melindungi data teks. Metode Agile dipilih karena memberikan tingkat fleksibilitas yang tinggi. Dengan metode *Agile*, proses pengembangan aplikasi dapat kembali ke tahap sebelumnya jika diperlukan perubahan, yang merupakan salah satu keunggulannya. Fleksibilitas ini menjadikan Agile pilihan yang sangat tepat dalam pengembangan perangkat lunak, terutama dalam proyek yang menghadapi perubahan kebutuhan dan kondisi selama

proses pengembangan (Prastowo dkk., 2023). Berikut pada Gambar 3.14. menunjukan diagram ilustrasi tahapan proses pengembangan aplikasi dengan metode *Agile.*



Gambar 3.14 Tahapan Metode Agile (Badiwibowo Atim & Korespondensi, 2024)

1. Planning

Pada tahap Perancangan (*Plan)*, proses dimulai dengan mengidentifikasi dan mendefinisikan tujuan serta kebutuhan aplikasi yang akan dibangun. Tujuan utama proyek ini adalah mengembangkan aplikasi web yang menggabungkan algoritma enkripsi AES dengan teknik NoStega untuk melindungi data teks. Langkah pertama mencakup identifikasi fitur-fitur yang perlu diterapkan dalam aplikasi, seperti antarmuka pengguna untuk mengunggah *file* teks dan kunci enkripsi.

2. Design

Pada tahap desain (*Design)* dalam metodologi *Agile*, proses difokuskan pada perancangan awal aplikasi yang akan dikembangkan. Dalam penelitian ini, berbagai diagram dibuat untuk memvisualisasikan desain dan spesifikasi aplikasi, termasuk diagram arsitektur, diagram *use case*, serta diagram alir. Diagram-diagram tersebut digunakan untuk menggambarkan proses dalam setiap modul utama, seperti enkripsi AES dan teknik *Noiseless Steganografi* yang diterapkan pada pengamanan data teks. Rancangan ini bertujuan untuk memastikan bahwa setiap komponen aplikasi terstruktur dengan baik sebelum masuk ke tahap implementasi.

3. Develop

Setelah tahap desain selesai, proyek berlanjut ke tahap pengembangan (*Develop)*, yang menjadi inti dari metodologi *Agile*. Pada tahap ini, aplikasi mulai dibangun berdasarkan rancangan yang telah dibuat sebelumnya. Pengembangan dilakukan dalam iterasi atau sprint, yang memiliki durasi lebih panjang dibandingkan tahapan lain dalam metodologi *Agile*. Setiap iterasi difokuskan pada pengembangan fitur tertentu yang telah ditentukan sebelumnya. tahap pertama difokuskan pada implementasi algoritma enkripsi AES. Setelah algoritma enkripsi berhasil diterapkan, tahap berikutnya berfokus pada integrasi teknik *Noiseless Steganografi* pada data teks. Pendekatan ini memungkinkan pengembangan yang fleksibel dan bertahap, sehingga setiap fitur dapat diuji dan disempurnakan sebelum melanjutkan ke tahap berikutnya.

4. Test

Setiap iterasi atau sprint dalam proyek ini diakhiri dengan tahap pengujian untuk memastikan bahwa setiap modul dan fitur yang dikembangkan berfungsi dengan baik serta sesuai dengan kebutuhan yang telah ditetapkan. Pengujian dilakukan secara berkelanjutan pada setiap iterasi guna mendeteksi dan memperbaiki kesalahan secepat mungkin.

Dalam penelitian ini, metode pengujian yang digunakan adalah *Black Box Testing*, yang berfokus pada validasi *input* dan *output* tanpa menguji struktur internal sistem. Pengujian ini bertujuan untuk memastikan bahwa semua fitur aplikasi berfungsi sesuai spesifikasi serta untuk mengidentifikasi dan mengatasi *bug* atau masalah yang mungkin terjadi sebelum dilanjutkan ke tahap selanjutnya.

5. Deploy

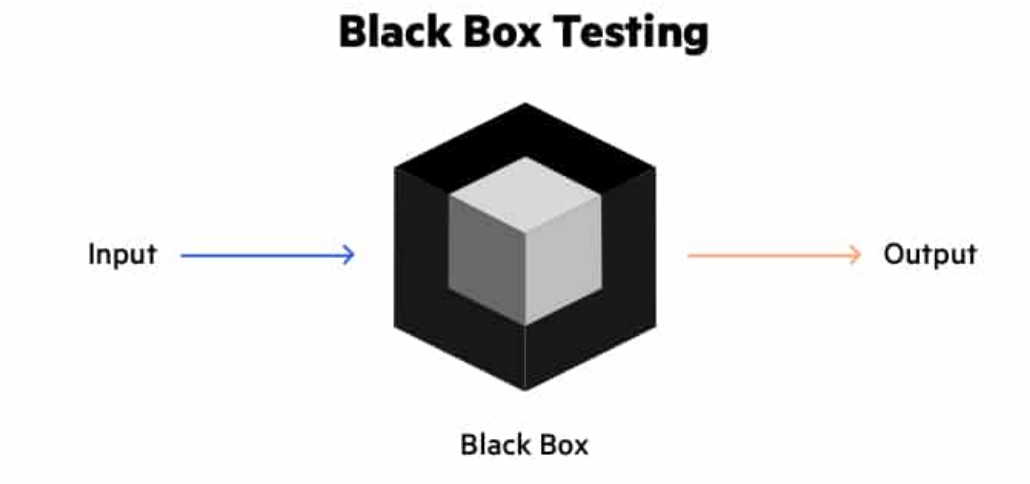
Pada tahap *deployment* dalam penelitian ini, aplikasi tidak dipublikasikan secara luas atau diimplementasikan melalui internet, karena aplikasi yang dikembangkan hanya untuk komputer lokal. Sebagai alternatif, deployment dilakukan dengan mengunggah *source code* dan aplikasi ke *platform* repositori *online* seperti GitHub. Dengan cara ini, aplikasi dapat diakses oleh pengguna atau developer yang ingin memanfaatkan atau mengembangkan proyek ini lebih lanjut.

6. Review

Pada tahap akhir *Agile*, yaitu tahap *review*, penulis melakukan evaluasi terhadap hasil setiap iterasi dan mengumpulkan umpan balik (*feedback*) dari pengguna atau pengembang aplikasi. Umpan balik ini digunakan untuk mengidentifikasi aspek-aspek yang perlu diperbaiki atau ditingkatkan dari fitur yang telah dikembangkan. Melalui proses ini, penulis dapat terus memperoleh masukan yang berharga selama proses pengembangan aplikasi, memastikan perbaikan dan peningkatan berkelanjutan.

3.5. Pengujian dan Evaluasi

Setelah fase pengembangan selesai, langkah berikutnya adalah melakukan pengujian dan evaluasi terhadap aplikasi yang telah dibangun. Tahap ini sangat krusial untuk memastikan bahwa aplikasi yang dikembangkan sesuai dengan spesifikasi dan kebutuhan yang telah ditentukan sebelumnya. Dalam proses ini, metode pengujian *black box*, atau yang juga dikenal sebagai *behavioral testing*, digunakan untuk fokus pada persyaratan fungsional perangkat lunak. Pengujian *black box* dirancang untuk mengidentifikasi berbagai jenis kesalahan, seperti fitur yang tidak berjalan atau tidak ada, kesalahan antarmuka, kesalahan dalam struktur data, serta kesalahan perilaku atau kinerja Aplikasi (Setiyani, 2018). Untuk ilustrasi metode pengujian *black box*, dapat terlihat pada Gambar 3.15.



Gambar 3.15 Pengujian Black Box

Melalui metode pengujian *black box testing*, peneliti dapat menyusun berbagai skenario untuk menguji sistem yang telah dikembangkan. Fokus utama dari metode ini

adalah memvalidasi apakah sistem dapat menjalankan fungsi-fungsinya sesuai dengan yang telah ditentukan dalam spesifikasi awal. Pengujian dilakukan dengan memberikan *input* tertentu dan kemudian mengevaluasi *output* yang dihasilkan, untuk memastikan bahwa hasilnya sesuai dengan yang diharapkan. Dalam konteks penelitian ini, pengujian mencakup beberapa aspek penting, seperti fungsi utama dari sistem, proses enkripsi data, serta keakuratan hasil yang ditampilkan. Dengan pendekatan ini, peneliti dapat mengidentifikasi adanya kesalahan logika, ketidaksesuaian *output*, atau potensi kerentanan tanpa harus memeriksa struktur kode internal. Ruang lingkup pengujian tersebut dijabarkan lebih lanjut pada Tabel 3.1.

Tabel 3.1 Skenario uji coba *Black Box* Aplikasi Web

Proses	Jenis Fitur	Parameter	Hasil yang Diharapkan
Enkripsi	1. Tampilan *input* pesan teks (Enkripsi) 2. Tampilan *input* kunci AES (Enkripsi) 3. Tampilan *output* stego audio (Enkripsi) 4. Tampilan *output* Kunci AES 5. Tampilan *output* Kunci	1. User dapat menginput pesan teks yang akan dienkripsi 2. User dapat menginput kunci AES untuk proses enkripsi 3. Sistem dapat menghasilkan file audio hasil steganografi 4. Sistem menampilkan kembali kunci AES yang digunakan 5. Sistem menampilkan urutan posisi chord s ebagai password ked ua	1. Pesan teks berhasil diinput dan ditampilkan pada form enkripsi 2. Kunci AES berhasil diinput dan digunakan untuk proses enkripsi 3. File stego audio dapat diunduh dan diputar tanpa error 4. Kunci AES tampil jelas dan dapat disalin oleh user 5. Kunci urutan posisi chord tampil jelas dan dapat disalin oleh user

Proses	Jenis Fitur	Parameter	Hasil yang Diharapkan
	urutan posisi chord		
Dekripsi	1. Tampilan *input file* stego audio (Dekripsi) 2. Tampilan *input* Kunci AES (Dekripsi) 3. Tampilan *input* Kunci urutan posisi chord (Dekripsi) 4. Tampilan *output* pesan teks (Dekripsi)	1. User dapat mengunggah file stego audio yang akan didekripsi 2. User dapat menginput kunci AES untuk proses dekripsi 3. User dapat menginput urutan posisi chord untuk proses dekripsi 4. Sistem menampilkan hasil dekripsi berupa pesan teks asli	1. File stego audio berhasil diunggah dan diproses oleh sistem 2. Kunci AES berhasil diinput dan diverifikasi oleh sistem 3. Kunci urutan posisi chord berhasil diinput dan digunakan untuk ekstraksi pesan 4. Pesan teks asli berhasil ditampilkan jika semua input benar

Proses pengujian algoritma enkripsi pada penelitian ini dengan menguji algoritma AES untuk memastikan keberhasilannya dalam melakukan enkripsi. Pengujian dilakukan dengan korelasi Pearson, yaitu metode statistik yang digunakan untuk mengukur tingkat hubungan linear antara dua variabel, dalam hal ini antara *plaintext* dan *ciphertext*. Nilai koefisien Korelasi Pearson berkisar antara -1 hingga 1, di mana nilai mendekati 0 menunjukkan tidak adanya hubungan *linear* yang signifikan antara kedua variabel. Dalam konteks enkripsi, semakin rendah nilai korelasi antara *plaintext* dan *ciphertext*, maka semakin baik algoritma dalam mengacak data sehingga hasil enkripsi sulit ditebak atau dianalisis. Pengujian ini juga digunakan untuk mengamati

pada enkripsi AES, yaitu sejauh mana perubahan satu bit dalam *plaintext* atau kunci dapat mempengaruhi banyak bit dalam *ciphertext*. Dengan demikian, Korelasi Pearson menjadi salah satu indikator penting untuk menilai kekuatan dan keamanan algoritma enkripsi yang digunakan.

Tahap berikutnya, hasil *noiseless steganography* diuji menggunakan kuesioner dengan parameter skala Likert untuk mengukur tingkat kecurigaan serta kualitas audio pada hasil *noiseless steganography*. Metode kuesioner ini dipilih agar dapat memperoleh penilaian subjektif dari responden terkait seberapa mencurigakan dan seberapa baik kualitas audio yang dihasilkan setelah proses penyisipan pesan rahasia. Skala Likert digunakan untuk memberikan rentang penilaian, mulai dari sangat tidak setuju hingga sangat setuju, sehingga dapat mengurangi kemungkinan bias dan memberikan gambaran yang lebih objektif mengenai persepsi responden. Melalui pengujian ini, diperoleh skor penilaian yang mencerminkan tingkat kecurigaan dan kualitas audio berdasarkan pengalaman langsung para responden saat mendengarkan *file* audio *noiseless steganography*. Hasil dari kuesioner ini menjadi salah satu indikator penting dalam menilai keberhasilan teknik *noiseless steganography* yang diterapkan, khususnya dalam aspek penyamaran pesan rahasia dan kualitas audio yang dihasilkan.

Untuk mengukur tingkat kecurigaan dan kualitas audio secara terstruktur melalu kuesioner, maka disusun instrumen penerimaan responden seperti yang terlihat pada Tabel 3.2. Instrumen ini terdiri dari tujuh pernyataan yang dirancang untuk menilai persepsi responden secara subjektif. Setiap pernyataan, dinilai menggunakan skala Likert dari 1 hingga 5, di mana 5 menunjukkan Sangat Setuju (SS) dan 1 menunjukkan Sangat Tidak Setuju (STS).

Tabel 3.2 Instrumen Penerimaan Responden Terhadap Stego Audio

No	Pertanyaan	Penilaian				
		SS (5)	S (4)	N (3)	TS (2)	STS (1)
1	Apakah musik "Mary Had A Little Lamb" pada file audio tersebut **tidak** menimbulkan kecurigaan?					
2	Apakah musik "Mary Had A Little Lamb" pada file audio tersebut **tidak** terdengar suara *noise*?					
3	Apakah Anda merasa audio tersebut terdengar seperti musik biasa pada umumnya?					
4	Apakah tempo dan nada lagu terdengar wajar dan tidak berubah?					
5	Apakah kualitas suara audio terdengar jernih dan nyaman didengarkan?					
6	Apakah volume audio terdengar stabil dari awal hingga akhir lagu?					
7	Musik "Mary Had A Little Lamb" pada file audio berukuran 10.3MB dengan durasi 2 menit, apakah hal tersebut wajar untuk sebuah file musik?					

Tujuan dari instrumen pada Tabel 3.2 ini adalah untuk memperoleh data kuantitatif dari penilaian responden. Skor yang dihasilkan dari instrumen inilah yang menjadi dasar untuk mengukur seberapa berhasil teknik *Noiseless Steganography* dalam menyembunyikan pesan tanpa menimbulkan rasa kecurigaan terhadap file stego audio.

Menurut Sugiyono, 2019. Instrumen penilaian disusun dengan perhitungan skala Likert dengan lima tingkat penilaian, yaitu:

1. Sangat Setuju (SS): 5 poin
2. Setuju (S): 4 poin
3. Netral (N): 3 poin
4. Tidak Setuju (TS): 2 poin
5. Sangat Tidak Setuju (STS): 1 poin

Setelah data hasil kuesioner dari para responden sudah diterima, maka selanjutnya bisa melakukan perhitungan untuk mencari persentase dari total semua jawaban responden dengan persamaan berikut:

$$Persentase = \frac{Skor\ Aktual}{Skor\ Maks}\ x\ 100\% \quad (2)$$

Skor Aktual: Jumlah skor yang didapat berdasarkan jawaban responden

Skor Maks: Jumlah skor maksimum yang mungkin didapatkan dari jawaban responden

Setelah mendapatkan hasil akhir kuesioner dengan perhitungan skala Likert dalam bentuk presentase, langkah selanjutnya adalah menginterpretasikan nilai persentase tersebut. Tingkat keberhasilan metode *Noiseless Steganography* dalam menyembunyikan pesan dan menjaga kualitas audio dapat dinilai berdasarkan tabel interpretasi persentase. Pada Tabel 3.3 ini mengategorikan hasil yang diperoleh ke dalam tingkatan yang terukur, memberikan gambaran yang jelas mengenai tingkat penerimaan responden.

Tabel 3.3 Interpretasi Persentase Besar dari Kuesioner (Sugiyono, 2019)

Persentase	**Keterangan**
1% - 20%	Sangat Tidak Setuju
21% - 40%	Tidak Setuju
41% - 60%	Netral
61 % - 80%	Setuju

Persentase	Keterangan
81% - 100%	Sangat Setuju

Dengan mengacu pada Tabel 3.3, kita dapat menentukan sejauh mana aplikasi ini memenuhi tujuan penelitian. Jika persentase hasil berada di rentang 81% - 100%, maka dapat disimpulkan bahwa metode ini layak dan memenuhi ekspektasi. Kategori interpretasi ini akan menjadi dasar kuat untuk menarik kesimpulan akhir tentang keberhasilan pengujian *Noiseless Steganography*.

Pengujian dilakukan menggunakan perangkat Lenovo S340 dengan spesifikasi prosesor AMD Ryzen 5 3500U, RAM 12 GB, dan sistem operasi Windows 11. Spesifikasi ini berperan penting dalam menentukan kualitas aplikasi, khususnya dalam hal proses enkripsi, dekripsi, dan ekstraksi pesan.

Evaluasi dilakukan setelah seluruh pengujian selesai. Penulis menganalisis hasil pengujian guna memastikan bahwa aplikasi telah memenuhi spesifikasi dan kebutuhan yang telah ditetapkan pada tahap perencanaan dan desain. Selain itu, proses evaluasi juga berfungsi untuk mengidentifikasi area yang memerlukan perbaikan.

3.6. Pelaporan

Setelah tahap pengujian dan evaluasi aplikasi selesai, langkah selanjutnya adalah pelaporan. Tahap ini mencakup penyusunan laporan hasil penelitian dalam bentuk skripsi, yang bertujuan untuk mendokumentasikan seluruh proses, temuan, dan metode yang digunakan secara sistematis dan terperinci. Dengan adanya dokumentasi ini, penelitian diharapkan dapat memberikan kontribusi signifikan bagi pengembangan ilmu di bidang keamanan data serta menjadi referensi bagi penelitian lain yang berfokus pada topik serupa. Selain itu, laporan ini dapat menjadi dasar bagi studi selanjutnya, memungkinkan peneliti lain untuk memperluas atau mengembangkan penelitian ini dengan pendekatan yang lebih mendalam berdasarkan metodologi dan temuan yang telah dipaparkan.