# Analisis Security Voice Authenticator pada Sistem Login Two Factor Authentication

**oleh**

**Gilbert Alexandro Onggo**

**535170006**



**Skripsi yang Ditulis untuk Memenuhi Sebagian Persyaratan**

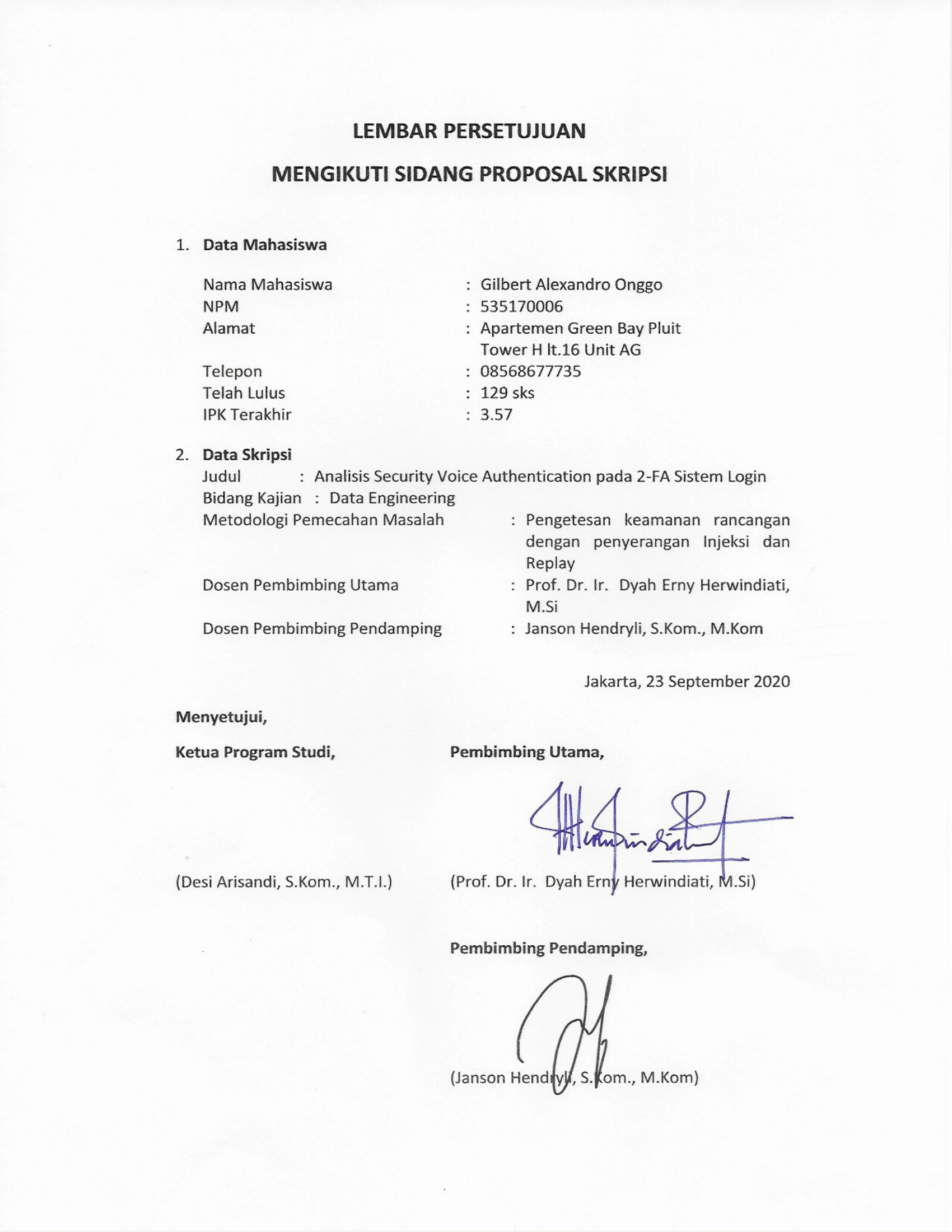
**Dalam Mendapatkan Gelar Sarjana Komputer**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS TARUMANAGARA

AGUSTUS 2020



# DAFTAR ISI

[Judul i](#_Toc52100386)

[Lembar persetujuan mengikuti sidang proposal skripsi ii](#_Toc52100387)

[DAFTAR ISI III](#_Toc52100387)

[DAFTAR TABEL V](#_Toc52100388)

[DAFTAR GAMBAR VI](#_Toc52100389)

[DAFTAR LAMPIRAN viii](#_Toc52100390)

[BAB I PENDAHULUAN 1](#_Toc52100391)

[1.1. Latar Belakang 1](#_Toc52100392)

[1.2. Rumusan Rancangan 6](#_Toc52100393)

[1.3. Batasan Rancangan 7](#_Toc52100394)

[1.4. Spesifikasi Rancangan 7](#_Toc52100395)

[1.5. Tujuan dan Manfaat Rancangan 8](#_Toc52100396)

[1.6. Rancangan Yang Sudah Pernah Dibuat 8](#_Toc52100397)

[BAB II LANDASAN TEORETIK 10](#_Toc52100398)

[2.1. Sistem yang Dirancang 10](#_Toc52100399)

[2.2. Landasan Teori 13](#_Toc52100400)

[2.2.1. Hash 13](#_Toc52100401)

[2.2.1.1 Algoritma Enkripsi MD5 13](#_Toc52100402)

[2.2.2. Audio 16](#_Toc52100403)

[2.2.2.1. Data Audio 17](#_Toc52100404)

[2.2.2.2. Waveform Audio File Format 20](#_Toc52100405)

[2.2.3. Mel Frequency Cepstrum Cepstral Coefficient 21](#_Toc52100406)

[2.2.3.1. Frame Blocking 22](#_Toc52100404)

[2.2.3.2. Windowing 22](#_Toc52100405)

[2.2.3.3. Discrete Fourier Transform 23](#_Toc52100404)

[2.2.3.4. Mel-Frequency Wrapping 25](#_Toc52100405)

[2.2.3.5. Discrete Cosine Transform 28](#_Toc52100405)

[2.2.4. Gaussian Mixture Model 29](#_Toc52100406)

[2.2.5. Application Programming Interface (API) 31](#_Toc52100406)

[2.2.6. Serangan API 32](#_Toc52100407)

[2.2.6.1. Replay Attacks 32](#_Toc52100408)

[2.2.6.1. Injection Attacks 33](#_Toc52100409)

[2.2.7. Keamanan API 33](#_Toc52100410)

[2.2.7.1. API Token (OAuth 2) 33](#_Toc52100411)

[2.2.7.1. API Key 34](#_Toc52100412)

[2.2.8. Black Box Testing 34](#_Toc52100413)

[2.2.9. C# (Visual C-Sharp) 35](#_Toc52100414)

[2.2.10. ASP.NET 35](#_Toc52100415)

[2.2.11. Javascript 36](#_Toc52100416)

[2.2.12. HTML 36](#_Toc52100417)

[2.2.13. MySQL 37](#_Toc52100418)

[2.3 Rekayasa Perangkat Lunak 37](#_Toc52100419)

[2.3.1. System Development Life Cycle (SDLC) 38](#_Toc52100420)

[2.3.1.1. Waterfall 38](#_Toc52100421)

[BAB III RANCANGAN DAN PEMBUATAN 40](#_Toc52100422)

[3.1. Rancangan Sistem 40](#_Toc52100423)

[3.1.1. Tahap Analisis 40](#_Toc52100424)

[3.1.1.1. Analisis Kebutuhan Khusus 40](#_Toc52100425)

[3.1.1.2. Analisis Fitur Aplikasi Sejenis 41](#_Toc52100426)

[3.1.1.3. Analisis Kebutuhan Perangkat 43](#_Toc52100427)

[3.1.2. Tahap Perancangan 44](#_Toc52100428)

[3.1.2.1. Rancangan Skema Sistem 44](#_Toc52100429)

[3.1.2.2. Rancangan Diagram Alur (Flowchart) 44](#_Toc52100430)

[3.1.2.3. Rancangan State Transition Diagram (STD) 45](#_Toc52100431)

[3.1.2.4. Rancanagan Data Flow Diagram (DFD) 46](#_Toc52100432)

[3.1.2.5. Rancangan Diagram Hierarki Sistem 47](#_Toc52100433)

[3.1.2.6. Rancangan Basis Data 47](#_Toc52100434)

[3.1.2.7. Rancangan Tampilan Antarmuka 49](#_Toc52100435)

[DAFTAR PUSTAKA 51](#_Toc52100436)

# DAFTAR TABEL

[Tabel 1 Contoh Sampling pada Audio Input 1](#_Toc52095426)7

[Tabel 2 Perbandingan Penelitian](#_Toc52095427) 29

[Tabel 3 Contoh Sampling untuk Perhitungan 41](#_Toc52095427)

[Tabel 4 Tabel Black Box Testing 52](#_Toc52095427)

[Tabel 5 Tabel User 52](#_Toc52095427)

[Tabel 6 Tabel API Server 52](#_Toc52095427)

# DAFTAR GAMBAR

[Gambar 1 Diagram Alur Penyerangan 1](#_Toc52095426)1

[Gambar 2 Diagram Alur Program Backend Otentikasi Suara 1](#_Toc52095426)2

[Gambar 3 Illustrasi proses operasi MD5 dalam satu putaran 1](#_Toc52095426)6

[Gambar 4 Contoh Data Audio Analog dan Digital](#_Toc52095427) 18

[Gambar 5 Contoh Besar Sampel 20](#_Toc52095427)

[Gambar 6 Skema API 3](#_Toc52095427)2

[Gambar 7 Skema Rancangan Sistem *Login* 88](#_Toc52095427)

[Gambar 8 Diagram Alur Sistem *Login* 89](#_Toc52095427)

[Gambar 9 Diagram STD Sistem *Login* dan Registrasi. 90](#_Toc52095427)

[Gambar 10 Diagram STD API Sistem *Login* 91](#_Toc52095427)

[Gambar 11 Context Diagram Sistem *Login* 92](#_Toc52095427)

[Gambar 12 DFD Sistem *Login* 93](#_Toc52095427)

[Gambar 13 Diagram Hierarki Sistem *Login* 94](#_Toc52095427)

[Gambar 14 Hubungan Antar Entitas Sistem *Login* 95](#_Toc52095427)

[Gambar 15 Hubungan Antar Tabel Sistem *Login* 96](#_Toc52095427)

[Gambar 16 Rancangan Halaman *Login* untuk Sistem *Login* 98](#_Toc52095427)

[Gambar 17 Rancangan Halaman Registrasi untuk Sistem *Login* 99](#_Toc52095427)

[Gambar 18 Rancangan Halaman *Input* Suara *login* untuk Sistem *Login* 100](#_Toc52095427)

[Gambar 19 Rancangan Halaman *input* Suara Registrasi untuk Sistem *Login* 101](#_Toc52095427)

[Gambar 20 Rancangan Halaman Hasil *Login* untuk Sistem *Login* 102](#_Toc52095427)

# DAFTAR LAMPIRAN

[Lampiran 1 Contoh Perhitungan Kriptografi MD5 54](#_Toc52095426)

[Lampiran 2 Contoh Perhitungan Data Audio 74](#_Toc52095426)

[Lampiran 3 Contoh Perhitungan MFCC 76](#_Toc52095427)

[Lampiran 4 Tabel Black Box Testing 87](#_Toc52095427)

[Lampiran 5 Skema Rancangan Sistem *Login* 88](#_Toc52095427)

[Lampiran 6 Diagram Alur Sistem *Login* 89](#_Toc52095427)

[Lampiran 7 Diagram STD Sistem *Login* dan Registrasi. 90](#_Toc52095427)

[Lampiran 8 Diagram STD API Sistem *Login* 91](#_Toc52095427)

[Lampiran 9 Context Diagram Sistem *Login* 92](#_Toc52095427)

[Lampiran 10 DFD Sistem *Login* 93](#_Toc52095427)

[Lampiran 11 Diagram Hierarki Sistem *Login* 94](#_Toc52095427)

[Lampiran 12 Hubungan Antar Entitas Sistem *Login* 95](#_Toc52095427)

[Lampiran 13 Hubungan Antar Tabel Sistem *Login* 96](#_Toc52095427)

[Lampiran 14 Spesifikasi Tabel Use 97](#_Toc52095427)

[Lampiran 15 Rancangan Halaman *Login* untuk Sistem *Login* 98](#_Toc52095427)

[Lampiran 16 Rancangan Halaman Registrasi untuk Sistem *Login* 99](#_Toc52095427)

[Lampiran 17 Rancangan Halaman *Input* Suara *login* untuk Sistem *Login* 100](#_Toc52095427)

[Lampiran 18 Rancangan Halaman *input* Suara Registrasi untuk Sistem *Login* 101](#_Toc52095427)

[Lampiran 19 Rancangan Halaman Hasil *Login* untuk Sistem *Login* 102](#_Toc52095427)

# BAB I PENDAHULUAN

## Latar Belakang

Pengenalan secara biometrik, menawarkan solusi yang andal untuk masalah otentikasi pengguna dalam sistem manajemen identitas[[1]](#footnote-1). Ada banyak alat dan teknik yang dapat mendukung pengelolaan keamanan informasi. Tetapi sistem berbasis biometrik telah berkembang untuk mendukung beberapa aspek keamanan informasi. Otentikasi biometrik mendukung aspek identifikasi, otentikasi, dan non-penolakan dalam keamanan informasi.[[2]](#footnote-2) Otentikasi biometrik menggunakan metode matematika otomatis untuk mengenali seseorang berdasarkan karakteristik fisiologis atau perilaku, seperti sidik jari, geometri wajah, atau ucapan. Kemampuan suatu aplikasi yang memungkinkan pengenalan berdasarkan lebih dari satu karakteristik manusia disebut multi-modal. Multi-modalitas diakui sebagai metodologi yang ampuh untuk mengatasi beberapa kelemahan sistem mono-modal, dengan memberikan bobot pada setiap karakteristik biometrik menurut faktor-faktor yang dapat mempengaruhi perolehan dan pemrosesan karakteristik biometrik yang berbeda secara tidak merata.[[3]](#footnote-3)

Menyusul ini, beberapa perusahaan besar telah mengambil sendiri untuk bertindak sebagai pelopor untuk otentikasi biometrik suara. Perusahaan finansial dari inggris seperti bank Barclays telah meluncurkan layanan pengenalan suara untuk melindungi diri mereka dari pemalsuan identitas menlalui aplikasi pada *smartphone* mereka. Bank lain seperti HSBC juga telah mengimplementasikan layanan serupa untuk pelanggan *smartphone* dan *mobile banking* yang memungkinkan nasabah untuk mengakses berbagai layanan di rekening bank mereka. Namun, agar teknologi ini menjadi umum di semua industri, ada kebutuhan yang lebih besar akan solusi yang berhubung, andal, dan aman. Misalnya, teknologi ini harus mampu beradaptasi dengan fakta bahwa suara manusia berubah dari waktu ke waktu, harus bebas dari kebisingan latar belakang dan dapat mengidentifikasi pengguna dengan akurat. Untuk melakukan otentikasi biometrik, dibutuhkannya API yang akan digunakan pada sistem *login* yang digunakan sebagai penguhubung antar aplikasi UI dengan program *backend*.

API merupakan kepanjangan dari *Application Programming Interface* yang berupa antarmuka komputasi yang mendefinisikan interaksi antara beberapa perantara perangkat lunak. API mendefinisikan jenis panggilan atau permintaan yang dapat dibuat, bagaimana membuatnya, format data yang harus digunakan, konvensi yang harus diikuti, dan lain-lain. API juga dapat menyediakan mekanisme ekstensi sehingga pengguna dapat memperluas fungsionalitas yang ada dengan berbagai cara dan untuk berbagai tingkat.[[4]](#footnote-4) API dapat diintegrasikan pada sistem *login* agar dapat dipadukan bersama otentikasi biometrik, tentunya hal ini menambah kesulitan untuk melakukan serangan yang bertujuan untuk mengambil informasi pengguna. Namun, masih ada cara bagi penyerang untuk dapat menembus atau memecah sistem *login* dan mengambil data sensitif yang berasal dari suatu aplikasi. Terutama perusahaan besar yang memilih untuk menggunakan otentikasi suara sebagai fitur keamanan mereka seperti bank, yang dimana menjadi sasaran utama dalam serangan digital*.* Untuk itu dibutuhkannya sebuah enkripsi untuk melindungi API juga agar sebuah rancangan dapat menjadi lebih aman dan rentan dari serangan.

Untuk melakukan suatu otentikasi suara, sebuah sistem *login* harus menggunakan API sebagai sarana untuk mengirim sebuah data dari suara tersebut agar dapat diproses pada program eksternal. Setelah data telah diproses hasil dari data tersebut akan dikirim kembali melalui API kepada sistem *login*. API harus bertahan dari serangan agar dapat mencegah penyerang untuk mengambil data pengguna yang dikirim. Pada tanggal 17 April 2020 riset menunjukkan bahwa bertambahnya lalu lintas koneksi yang berbahaya sebesar 40% hingga 28 juta panggilan dalam satu minggu pada API milik IBM. Kemudian pada awal Mei 2020, Panggilan tersebut bertambah menjadi 85% hingga 138 juta panggilan berbahaya pada satu titik akhir dalam API. Hal ini dikarenakan karena API tersebut pernah berhasil dibobol oleh penyerang dengan cara seperti itu. Terkadang titik akhir API yang difokuskan penyerang sudah usang, baru dipelajari, atau dilupakan sehingga tidak di pantau lagi.[[5]](#footnote-5)

Batasan yang selalu muncul pada sistem *login* dikarenakan mengikatnya mekanisme tradisional *password-only* dalam suatu server. Server harus menyimpan tabel verifikasi sensitif yang berisi sandi (atau sandi dalam *hash*) dari semua pengguna terdaftar. Jika server otentikasi berhasil disusupi, semua sandi pengguna akan terungkap. Karena keterbatasan memori dan anggaran keamanan manusia, distribusi sandi yang dipilih pengguna sangat tidak seimbang. Akibatnya, bahkan jika kata sandi disimpan dalam *salted* *hash*, ini tidak menimbulkan hambatan nyata bagi penyerang untuk memulihkannya dengan persentase yang luar biasa dengan menggunakan teknik pemecahan probabilistik modern dan perangkat keras umum seperti GPU.[[6]](#footnote-6) Untuk membuat API yang aman dibutuhkan beberapa tahap keamanan seperti, penggunaan Transport Layer Security (TLS) untuk mengenkripsi paket data yang dikirim, Penggunaan API *Key* yang digunakan sebagai kunci untuk mengakses API tersebut dan disimpan seaman mungkin.

Otentikasi biometrik membutuhkan model yang dapat membedakan input fisiologis dengan akurasi yang akurat. Model yang dipakai untuk otentikasi tersebut memerlukan waktu yang cukup lama atau kekuatan komputasi besar, agar dapat memberikan hasil dengan akurasi yang memuaskan. Karena ini, sebagian besar sistem *login* yang menggunakan *voice authenticator*, menggunakan program terpisah yang sudah ada dan dikelola oleh pihak ketiga dibanding dibuat dari awal. Hal ini menyebabkan rentannya sistem tersebut karena membutuhkan koneksi diatara sistem *login* dengan program otentikasi. Pada zaman ini, banyak pengembang membuat otentikasi suara milik sendiri walaupun mengetahui bahwa otentikasi tersebut memiliki kerentanan. Karena itu pengembang kecil menggunakan otentikasi yang tersedia di publik untuk menghemat biaya dan waktu untuk dapat menambah keamanan pada aplikasi yang mereka kembangkan.

Skripsi ini membahas mengenai kerentanan yang dapat dijumpai sistem *login* yang diintegrasikan dengan API *voice authenticator* menggunakan Aplikasi Web. Sistem Login dengan API *voice authenticator* merupakan tambahan keamanan login yang awalnya pengguna cukup melakukan otentikasi dengan password. Dengan API ini pengguna wajib menginput suara agar dapat dilakukan otentikasi pada program yang terhubung dengan API tersebut.

Pengamatan dilakukan pada penyerangan sistem *login* bertujuan untuk menganalisis kerentanan koneksi antar sistem dengan penyedia layanan otentikasi biometrik suara sehingga dapat mencari cara untuk mencegah kerentanan tersebut. Pengguna akan diminta untuk memasukkan *username* dan *password,* jika user ditemukan pada database maka pengguna akan melanjutkan ke tahap berikutnya yaitu berupa *input* suara pengguna untuk dapat di verifikasi menggunakan API untuk dapat di verifikasi pada program backendnya. Hasil evaluasi dari sistem yang dibuat diukur dari ketangguhan terhadap gangguan dari serangan-serang yang dilakukan, seperti *Replay* dan *Injection*. Metode-metode tersebut merupakan metode yang sering digunakan oleh penyerang karena kemudahan akses atas alat dan kemudahan untuk dipelajari.

## Rumusan Rancangan

Berdasarkan latar belakang permasalahan di atas, maka akan dibuat rancangan sistem login seaman mungkin. Untuk keamanan, program akan menggunakan metode MD5 *password hashing* dan Integrasi dengan API untuk otentikasi suara sebagai otentikasi kedua.

Dengan metode dan integrasi ini, Rancangan kemudian akan di tes keamanannya menggunakan metode *Inject* dan *Replay* karena metode tersebut merupakan serangan umum yang kerap terjadi pada sebuah API.

## Batasan Rancangan

Sistem *login* yang dirancang memiliki batasan-batasan sebagai berikut:

1. Data User yang digunakan untuk menyimpan data pada database user seperti userid, username dan password akan digabung bersama rancangan website sementara penyimpanan suara di simpan pada server API.
2. Otentikasi suara akan menggunakan kalimat “Nama saya adalah …”.
3. Rancangan hanya melingkupi sistem login yang berbasis website.
4. Masukan pada aplikasi berupa teks *username* dan *password.* setelah melewati tahap pertama pengguna akan diminta masukan berupa suara sesuai dengan kalimat yang dipakai saat pembuatan akun.
5. Program otentikasi suara menggunakan program *open source* yang dibuat dengan model *Gaussian Mixture Model*.
6. Rancangan ini dibuat sebagai usulan bagi pengembang website dengan sistem login yang menggunakan program otentikasi suara.

## Spesifikasi Rancangan

Pada rancangan sistem *login* *Two Factor Authentication* dengan *Voice Authentication* hanya memiliki tiga modul yaitu:

1. Modul Sistem *Login*

Modul ini merupakan modul dengan dimana rancangan akan mengecek apabila *username* dan *password* yang pengguna masukkan ada dalam database dan benar.

1. Modul Registerasi

Modul ini merupakan proses untuk membuat user baru dimana rancangan akan memasukan username, password dan suara pengguna untuk dikirim menggunakan API.

1. Modul API

Modul ini merupakan lanjutan dari modul sistem *login* dan registerasi sebagai sarana untuk mengirim suara pengguna ke program.

## Tujuan dan Manfaat Rancangan

Beberapa kegunaan dari perancangan sistem yang dibuat adalah sebagai berikut:

1. Membangun sistem yang dapat menjaga keamanan data pengguna.
2. Menerapkan metode keamanan dalam pembuatan sistem.
3. Menganalisis titik lemah sistem dengan beberapa metode serangan.

## Rancangan Yang Sudah Pernah Dibuat

Berikut beberapa penelitian relevan yang pernah dibuat:

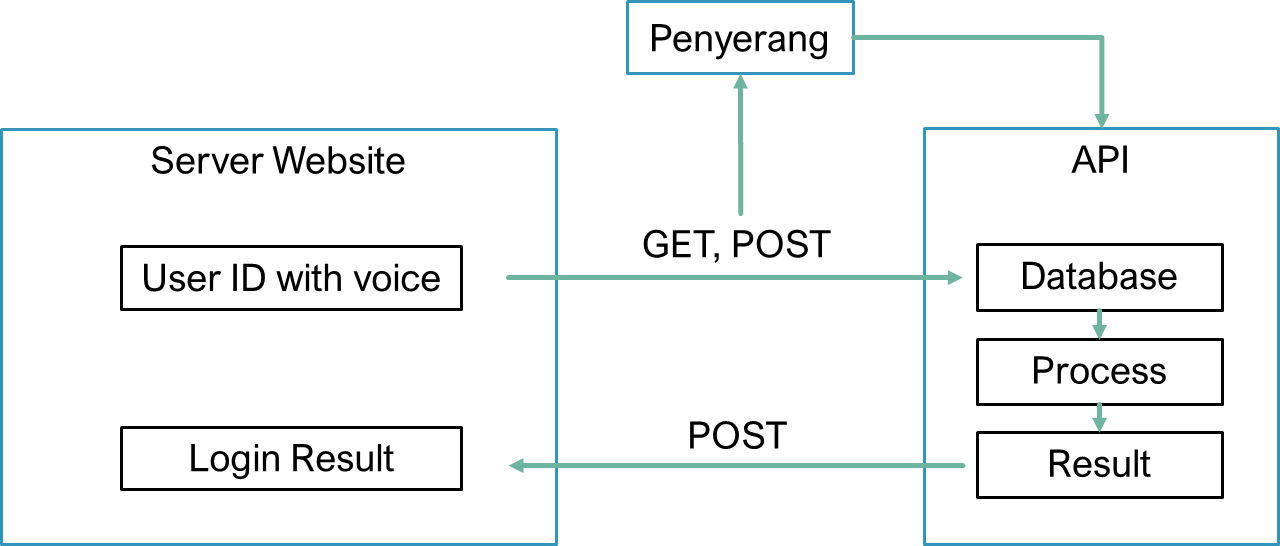
1. Penelitian yang dilakukan oleh Christian Zeitza , Tobias Scheidata , Jana Dittmanna , Claus Vielhauera , Elisardo González Agullab , Enrique Otero Murasb , Carmen García Mateob dan José L. Alba Castrob dengan judul “*Security issues of Internet-based biometric authentication systems: risks of Man-in-the-Middle and BioPhishing on the example of BioWebAuth*”. Penelitian ini menunjukan bahwa terdapat banyak metode yang dapat dilakukan untuk meluncurkan serangan terhadap koneksi dalam proses otentikasi biometrik berbasis web.[[7]](#footnote-7)
2. Penelitian yang dilakukan oleh Battista Biggio, Zahid Akhtar, Giorgio Fumera, Gian Luca Marcialis, dan Fabio Roli dengan judul “*Security Evaluation of Biometric Authentication Systems Under Real Spoofing Attacks*”. Penelitian ini menunjukkan bahwa serangan *spoofing* dapat menembus sistem keamanan yang menggunakan otentikasi biometrik.[[8]](#footnote-8)

# BAB II LANDASAN TEORETIK

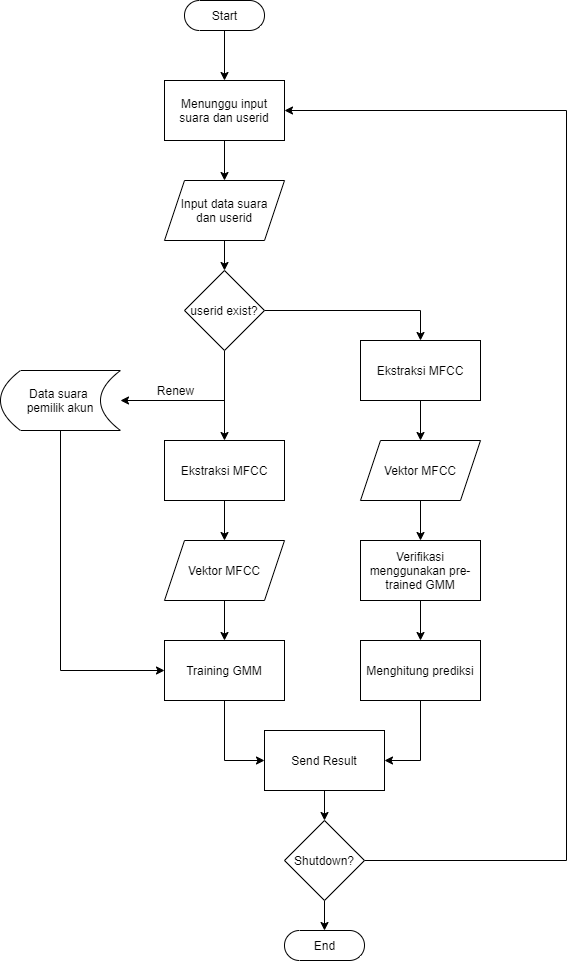
## 2.1. Sistem yang Dirancang

Sistem yang di rancang adalah sistem keamanan yang kuat terhadap serangan *digital.* Sistem tersebut adalah sistem login yang diintegrasikan dengan API otentikasi suara yang berguna untuk memperkuat keamanan data, sehingga pengguna dapat *login* tanpa khawatir atas kebobolan data. Sistem ini akan dibuat menggunakan metode kriptografi hashing MD5 untuk menyimulasikan sistem login yang umum dengan integrasi *Voice Authenticator* menggunakan metode GMM sebagai otentikasi ke-2. Algoritma hashing MD5 adalah fungsi kriptografi satu arah yang menerima teks dengan panjang berapa pun sebagai *input* dan mengembalikan sebagai *output* nilai inti sari dengan panjang tetap yang akan digunakan untuk mengautentikasi teks asli.[[9]](#footnote-9)

Beberapa tahap yang harus dilakukan sistem yaitu, Menyimpan data pengguna, mengenkripsi kata sandi pengguna, validasi pada database sistem dan validasi pada *voice* *authentication*, sebelum memberikan akses kepada pengguna. Tahap pertama tentunya dimulai dengan penyimpanan data pengguna dalam database aplikasi seperti *username* dan *password*, sedangkan suara pengguna akan dikirim langsung ke API agar bisa dipakai untuk verifikasi nantinya. Kemudian dilanjutkan ke tahap selanjutnya yaitu enkripsi kata sandi pengguna menggunakan algoritma enkripsi MD5. Setelah melakukan kedua tahap tersebut, pengguna dapat memasukkan data *username* dan *password* pada rancangan yang kemudian akan di validasi pada database yang terhubung dengan rancangan. Setelah validasi selesai, pengguna akan diminta melakukan validasi selanjutnya menggunakan suaranya. Suara tersebut akan di kirim ke program otentikasi melalui API dan rancangan akan menerima *feedback* berupa positif atau negatif dalam format JSON. Rancangan kemudian akan dites keamananya menggunakan beberapa penyerangan pada API tersebut menggunakan metode blackbox testing. Diagram alur penyerangan dapat dilihat pada **Gambar 1**.

  
**Gambar 1** Diagram Alur Penyerangan

Pemrosesan otentikasi suara menggunakan Gaussian Mixture Model yang dilengkapi dengan API untuk dapat dipakai oleh website *login* dapat dilihat pada **Gambar 2**.

  
**Gambar 2.** Diagram Alur Program Backend Otentikasi Suara

## 2.2. Landasan Teori

Pada landasan teori ini akan dijelaskan mengenai teori dan metode yang digunakan dalam membangun sistem *Login*. Berikut ini adalah beberapa bahasan mengenai teori dan metode yang digunakan.

### **2.2.1. Hash**

Fungsi *hash* adalah fungsi enkripsi satu arah yang mengambil teks biasa masukan ukuran variabel m dan menghasilkan keluaran *hash* ukuran tetap. Secara komputasi sulit untuk menguraikan *hash* dan setiap upaya untuk memecahkannya secara praktis tidak layak. Fungsi *hash* yang "aman" harus mampu menahan serangan pra-gambar dan serangan tabrakan. Karena prinsip *pigeonhole* dan *birthday paradox*, maka akan ada beberapa masukan yang akan menghasilkan hasil *hash* yang sama. Panjang keluaran adalah 128 bit ukuran tetap, membuat total 2128 kemungkinan nilai *output* dari *hash*. Nilai ini, sebesar yang terlihat, tidak terbatas, sehingga mengakibatkan tabrakan.[[10]](#footnote-10)

#### **2.2.1.1 Algoritma Enkripsi MD5**

Algoritma MD5 dirancang untuk bekerja cukup cepat pada mesin 32-bit. Selain itu, algoritma MD5 tidak memerlukan tabel substitusi yang besar sehingga algoritma dapat dikodekan dengan cukup kompak. Algoritma MD5 adalah perpanjangan dari algoritma inti sari teks MD4. MD5 sedikit lebih lambat dari MD4 tetapi lebih "konservatif" dalam desain. MD5 dirancang karena dirasa bahwa MD4 mungkin diadopsi untuk digunakan lebih cepat daripada yang dibenarkan oleh tinjauan kritis yang ada, karena MD4 dirancang untuk menjadi sangat cepat, algoritma ini "berada di ujung" dalam hal serangan kriptanalitik yang berhasil. MD5 mundur sedikit, memberikan sedikit kecepatan untuk kemungkinan keamanan tertinggi yang jauh lebih besar. Ini menggabungkan beberapa saran yang dibuat oleh berbagai pengulas dan berisi pengoptimalan tambahan. Algoritme MD5 ditempatkan di domain publik untuk ditinjau dan kemungkinan diadopsi sebagai standar.[[11]](#footnote-11)

Algoritma MD5 pertama-tama membagi masukan dalam blok masing-masing 512 bit. 64 Bit disisipkan di akhir blok terakhir. 64-bit ini digunakan untuk mencatat panjang input asli. Jika blok terakhir kurang dari 512 bit, beberapa bit tambahan '*Padded*' sampai akhir. Selanjutnya, setiap blok dibagi menjadi 16 kata dengan masing-masing 32 bit. Ini dilambangkan sebagai .

MD5 menggunakan buffer yang terdiri dari empat kata yang masing-masing memiliki panjang 32 bit. Kata-kata ini disebut A, B, C dan D.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Word A: | 01 | 23 | 45 | 67 |
| Word B: | 89 | ab | cd | ef |
| Word C: | fe | dc | ba | 98 |
| Word D: | 76 | 54 | 32 | 10 |

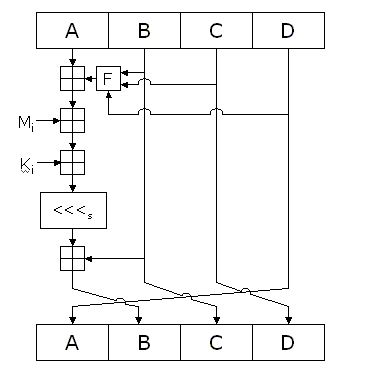
MD5 selanjutnya menggunakan tabel K yang memiliki 64 elemen. Elemen nomor i diindikasikan sebagai . Tabel dihitung sebelumnya untuk mempercepat penghitungan. Unsur-unsur tersebut dihitung menggunakan fungsi sin matematika:

(1)

Selain itu, MD5 menggunakan empat fungsi (F, G, H, I) tambahan yang masing-masing mengambil tiga kata 32-bit (X, Y, Z) sebagai masukan dan menghasilkan satu kata 32-bit sebagai keluaran. Mereka menerapkan operator logika *and*, or, *not* dan xor ke bit *input*.

|  |  |
| --- | --- |
| F(X, Y, Z) | = (X and Y) or (not(X) and Z) |
| G(X, Y, Z) | = (X and Z) or (Y and not (Z)) |
| H(X, Y, Z) | = X xor Y xor Z |
| I(X, Y, Z) | = Y xor (X or not(Z)) |

Isi dari empat *buffer* (A, B, C dan D) sekarang dicampur dengan kata-kata *input*, menggunakan empat fungsi tambahan (F, G, H dan I). Ada empat putaran, masing-masing melibatkan 16 operasi dasar. Salah satu operasi diilustrasikan pada **gambar 1**.

  
**Gambar 3** Illustrasi proses operasi MD5 dalam satu putaran  
**Sumber:** Anonymous, The MD5 cryptographic hash function, https://www.iusmentis.com/technology/hashfunctions/md5/, Oktober 2005

Gambar tersebut menunjukkan bagaimana fungsi bantu F diterapkan pada empat buffer (A, B, C dan D), menggunakan kata pesan dan konstanta . Item "" menunjukkan pergeseran kiri biner sebesar s bit. Setelah semua putaran dilakukan, buffer A, B, C dan D berisi intisari MD5 dari input asli.[[12]](#footnote-12) Contoh perhitungan kriptografi MD5 dapat dilihat pada **Lampiran 1.**

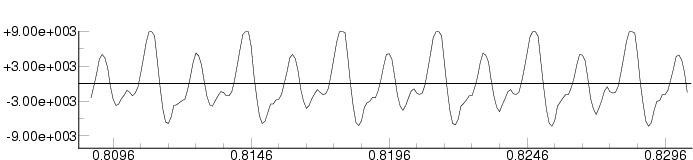
### **2.2.2. Audio**

Audio merupakan gelombang suara yang dihasilkan dari getaran pada rentang frekuensi audio yang dapat didengar oleh manusia. Telinga manusia hanya dapat mendengar bunyi dengan rentang frekuensi antara 20 Hz hingga 20 KHz (20.000Hz). Angka 20 Hz sebagai frekuensi suara terendah yang dapat didengar, sedangkan 20 KHz merupakan frekuensi tertinggi yang dapat didengar.[[13]](#footnote-13)

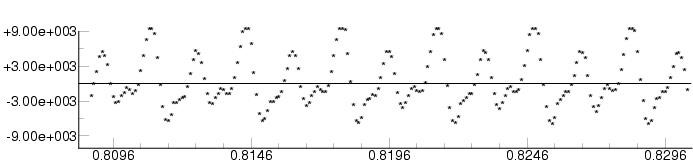
Gelombang suara terdiri dari beberapa kompenen penting diantaranya yakni amplitude, panjang gelombang dan frekuensi. Komponen ini yang merupakan penyebab setiap suara memiliki perbedaan yang unik. Suara dengan amplitudo yang besar dapat dinilai sebagai suara dengan volume yang keras. Frekuensi merupakan jumlah dari siklus yang terjadi dalam satu detik dan memiliki satuan Hertz (Hz). Getaran gelombang suara yang cepat membuat frekuensi semakin tinggi. Misalnya, bernyanyi dengan nada tinggi membuat tali suara pada pita suara bergetar dengan cepat.[[14]](#footnote-14)

#### **2.2.2.1. Data Audio**

Data audio dapat dibagi menjadi 2 jenis yaitu data analog dan data digital. Data analog merupakan data yang berbantuk gelombang suara secara kontinu. Sedangkan untuk data digital merupakan pengubahan data audio analog menjadi data digital dengan menggunakan alat yang disebut dengan *Analog To Digital Converter*(ADC). ADC memiliki cara kerja yang dimana alat ini mengubah gelombang analog ke dalam waktu internval (sampel) sehingga menghasilkan penyajian digital dari suara. Sehingga audio digital dapat disebut sebagai sampel suara [[15]](#footnote-15) data audio digital adalah data audio yang dapat diolah oleh komputer. Contoh data audio digital dan data audio analog dapat dilihat pada **Gambar 2.**



(a)



(b)

**Gambar 4*.***Contoh data audio digital (b) dan data audio analog (a)

**Sumber:**Bill Poser, Audio data, <http://billposer>.org/Linguistics/Computation/LectureNotes/AudioData.html, 9 September 2020.

Untuk melakukan digitalisasi dari bentuk audio analog, dibutuhkan konversi dengan menggunakan *Sampling* yang merupakan rekaman atas sinyal dalam interval secara reguler. *Sampling* dilakukan secara umum dengan mengambil sampel dari amplitudo dari gelombang suara, dengan interval regular. Nilai biner kemudian akan diberikan untuk setiap hasil. Contoh dari sampling dapat dilihat pada **tabel 1**.

|  |  |
| --- | --- |
| Waktu (s) | Amplitudo |
| 0 | 100 |
| 0.5 | 101 |
| 1 | 110 |
| 1.5 | 111 |

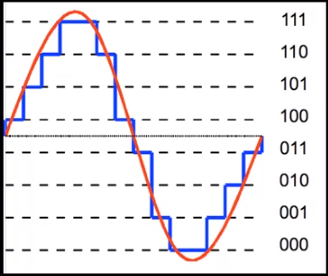
**Tabel 1.** Contoh Sampling pada Audio *input*

Frekuensi Sampel merupakan banyak sampel yang terekam tiap detik, dihitung dalam satuan Hertz (Hz).

(2)

Semakin besar frekuensi sampel semakin mirip sinyal digital dengan original, namun semakin besar besar *file.*

Besar sampel merupakan berapa banyak bits yang tersedia pada tiap sampel. Besar sampel diketahui juga sebagai *Bit Depth*. Contoh dari besar sample dapat dilihat pada **gambar 3**.

  
**Gambar 3.** Contoh besar sampel  
Sumber: Miguel Gudino, Engineering Resources: Basics of Analog-to-Digital Converters, https://www.arrow.com/en/research-and-events/articles/engineering-resource-basics-of-analog-to-digital-converters, April 2018.

Pada gambar tersebut besar sampel adalah 3-bit yang menghasilkan hanya 8 level diskrit () berupa jarak volume yang rendah. Data bit depth disimpan dalam bentuk biner kemudian menjadi reproduksi menjadi gelombang suara digital dari data bit tersebut.

*Bitrate* merupakan banyaknya bit yang digunakan per detik.

(3)

Semakin tinggi *Bitrate* menyebabkan semakin detil suara, namun membutuhkan besar penyimpanan yang lebih besar. Untuk menghitung besar file dapat menggunakan rumus:

(4)

= Sampling Frequency (Hz)

= Besar Sampel

= Waktu (detik)

Contoh perhitungan konversi pada *input* suara dari pengguna menjadi sinyal digital dapat dilihat pada **Lampiran 2.**

#### **2.2.2.2. Waveform Audio File Format**

*Waveform Audio File Format* (WAV) merupakan salah satu file audio yang memilki kualitas yang sangat baik. WAV merupakan format *file*yang dikembangkan oleh Microsoft dan IBM dengan tujuan menyimpan audio dalam komputer berbasis Pulse Code Modulation (PCM). PCM merupakan sebuah metode yang mengkonversi suara analog ke digital dan digunakan dalam format digital audio processing dalam bentuk biner. Bentuk format dari PCM adalah uncompressed, yang berarti ukuran file tersebut besar dan memiliki kualitas yang sangat tinggi, serta informasi yang detil. Format file WAV sangat cocok digunakan untuk implementasi data audio di berbagai bidang.[[16]](#footnote-16)

2.2.3. Mel Frequency Cepstrum Cepstral Coefficient

*Mel Frequency Cepstrum Cepstral Coefficient* (MFCC) merupakan metode ekstraksi fitur dalam bidang *signal processing* dan merupakan cara yang paling sering digunakan untuk ekstraksi ciri pada berbagai bidang area pemrosesan suara, karena dianggap cukup baik dalam merepresentasikan ciri sebuah sinyal. Skala mel yang linier pada frekuensi rendah dan logaritmik pada frekuensi tinggi ditemukan baik sebagai kandidat untuk merepresentasikan sifat tidak uniform tersebut[[17]](#footnote-17). Tujuan utama dari proses MFCC adalah menirukan perilaku pendengaran manusia[[18]](#footnote-18). Beberapa tahapan yang digunakan pada metode MFCC adalah Frame blocking, Windowing, Discrete Fourrier Transform, Mel-Frequency Wrapping, dan Discrete Cosine Transform.

2.2.3.1. Frame blocking

Pada proses Frame blocking sinyal suara memiliki panjang N melalui proses sampling diubah menjadi beberapa frames. Umumnya ukuran frame yang digunakan adalah 20 hingga 40 ms, dengan jumlah frame yang dipakai tergantung dari sampling rate pada data. Masing-masing frame yang digunakan overlap untuk menghindari hilangnya informasi pada sinyal data[[19]](#footnote-19).

2.2.3.2. Windowing

Proses selanjutnya adalah pemrosesan data dengan window pada masing-masing frame individual untuk meminimalisasi sinyal non-kontinu pada awal dan akhir masing-masing frame. Berikut ini adalah rumus untuk menghitung window [[20]](#footnote-20):

(5)

Window yang digunakan adalah Hamming Window dan dihitung menggunakan rumus berikut[[21]](#footnote-21):

(6)

Keterangan :

= Panjang setiap frame

= Sinyal masukan ke-n pada frame ke-t

= Fungsi hamming window

= Nilai hasil windowing untuk sinyal masukan ke-n pada frame ke-t

2.2.3.3. Discrete Fourier Transform

Bentuk sinyal data suara biasanya dalam domain waktu. Data sinyal domain waktu merupakan data mentah sehingga harus diubah menjadi domain frekuensi. Proses konversi sinyal dalam domain waktu menjadi domain frekuensi dapat dilakukan menggunkan metode Discrete Fourier Transform (DFT).[[22]](#footnote-22) Pada tahap ini setiap frame yang terdiri dari N sampel dikonversi dari domain waktu ke domain frekuensi.[[23]](#footnote-23) Hasil dari proses ini disebut dengan nama spektrum atau periodogram. Rumus DFT adalah sebagai berikut[[24]](#footnote-24).

(7)

Keterangan:

= Nilai transformasi Fourier (FT) ke-k

= Input signal ke-n

= Bilangan imajiner

= Panjang input Signal

Rumus DFT di atas dioperasikan pada hasil proses windowing sebelumnya yakni , sehingga rumus DFT pada input adalah[[25]](#footnote-25):

(8)

berdasarkan rumus Euler[[26]](#footnote-26):

(9)

dengan e adalah logaritma natural, i adalah bilangan imajiner, sedangkan cos dan sin adalah fungsi trigonometri. Jika persamaan 5 disubstitusikan ke persamaan 4 maka rumus DFT menjadi:

(10)

Keterangan:

= Nilai FT ke-k pada frame ke-t

= Nilai hamming window ke-n pada frame ke-t

= bilangan imajiner

Hasil perhitungan FT mengandung bilangan riil dan imajiner. Oleh karena itu, nilai Fourier spektrum didapat dari nilai magnitude yang dibentuk kedua bilangan rill dan imajiner. Besar nilai magnitude dari persamaan tersebut dihitung dengan rumus

(11)

Keterangan:

= Nilai FT yang mengandung bilangan kompleks

= Nilai magnitude

= Bilangan rill

= Koefisien dari bilangan imajiner

= Bilangan imajiner

2.2.3.4. Mel-Frequency Wrapping

Tahap ini merupakan proses filter dari spektrum setiap frame yang diperoleh dari tahapan sebelumnya, menggunakan sejumlah M filter segitiga dengan tinggi satu.[[27]](#footnote-27) Filter ini dibuat dengan mengikuti persepsi telinga manusia[[28]](#footnote-28). Studi psikofisikal menunjukkan bahwa persepsi manusia dari kandungan frekuensi suara pada sinyal wicara tidak mengikuti skala linier[[29]](#footnote-29). Persepsi ini dinyatakan dalam skala ‘mel’ yang mempunyai hubungan tidak linier dengan frekuensi suara. Dalam hal ini skala mel-frequency bersifat linier untuk frekuensi kurang dari 1000 Hz dan logaritmik untuk frekuensi di atas 1000 Hz[[30]](#footnote-30). Hubungan antara skala mel dengan frekuensi dinyatakan sebagai berikut:[[31]](#footnote-31)

(12)

Sedangkan inversnya adalah:[[32]](#footnote-32)

(13)

Triangular filter bank sebanyak P dibangun dengan tinggi yang sama. Nilai dari filter bank sendiri didefinisikan sebagai berikut:[[33]](#footnote-33)

(14)

Keterangan:

= 1, 2, …., N

= 1, 2, …, P

= titik batas dari filter

Dengan nilai adalah:[[34]](#footnote-34)

(15)

Keterangan:

= banyaknya sampel sinyal setiap frame

= titik sampel frekuensi

= frekuensi tertinggi pada keseluruhan filter

= frekuensi terendah pada keseluruhan filter

Nilai titik pusat filter ke-I yakni dirumuskan dengan:[[35]](#footnote-35)

(16)

Sehingga didapatkan nilai koefisien spektrum mel (mel Spectrum coefficients)[[36]](#footnote-36):

(17)

Keterangan:

= koefisien spektrum mel ke-I (i=1, 2, …, P)

= filter ke-i (i=1, 2, …, P) pada frekuensi n

= nilai *magnitude dari* yaitu

2.2.3.5. Discrete Cosine Transform

Koefisien spektrum mel yakni pada tahap sebelumnya masih dalam domain frekuensi. Oleh karena itu nilai koefisien spektrum mel diubah ke domain waktu.[[37]](#footnote-37) Hasil akhir dari MFCC berupa bilangan rill, oleh karena itu dilakukan Discrete Cosine Transform (DCT) untuk mengubah dari domain frekuensi ke domain waktu. MFCC didapat dari rumus:[[38]](#footnote-38)

(18)

Keterangan:

= Nilai MFCC ke-j

= Jumlah MFCC

= Koefisien spectrum mel ke-I (i=1, 2, …, P)

= jumlah filter

Contoh perhitungan MFCC dapat dilihat pada **Lampiran 3.**

### **2.2.4. Gaussian Mixture Model**

Gaussian Mixture Model (GMM) adalah sebuah tipe density model yang terdiri dari komponen fungsifungsi Gaussian.[[39]](#footnote-39) Gaussian mixture model adalah suatu himpunan komponen model fungsi kerapatan yang dapat menghitung dan melacak ruas suatu model spektrum. Mixture model juga merupakan metode yang efektif untuk perubahan dari model yang bergerak lambat karena merupakan semi parameter alternatif untuk histogram tanpa parameter dan memberikan fleksibilitas lebih serta ketelitian dalam memodelkan statistik dari data, dengan kata lain untuk mevisualkan suatu dynamic scene.

GMM juga merupakan model statistik dari distibusi probabilitas yang didapatkan dari nilai bobot setiap distribusi Gaussian sehingga GMM merupakan metode yang sangat tepat untuk perhitungan, baik dengan parameter maupun tidak. Bila model telah dihasilkan, syarat peluang dapat dihitung dan GMM juga dapat ditampilkan sebagai bentuk fungsi hubungan dasar network, seperti berikut ini[[40]](#footnote-40). Hasil prediksi dari model ini akan diberikan threshold tertentu, agar dapat memberikan output yang dapat dikirimkan ke sistem *login* melalui API.

Rumus dari fungsi ketebalan probabilitas dari distribusi gauss dengan 1 dimensi adalah:

(19)

Keterangan:  
 = nilai untuk mean yang memaksimalkan kecocokan  
 = Apriori dari Gauss

= Standar Deviasi

= mean gauss

= indeks gauss

= indeks data point

= data point

Kemudian akan dihitung kecocokan dengan rumus sebagai berikut:

(20)

Untuk Algoritma maksimalisasi ekspektasi untuk model campuran Gaussian dimulai dengan langkah inisialisasi, yang menetapkan parameter model ke nilai yang wajar berdasarkan data. Kemudian, model mengulangi langkah ekspektasi (E) dan maksimisasi (M) sampai estimasi parameter bertemu.

Langkah E dimana menghitung probabilitas menggunakan rumus sebagai berikut:

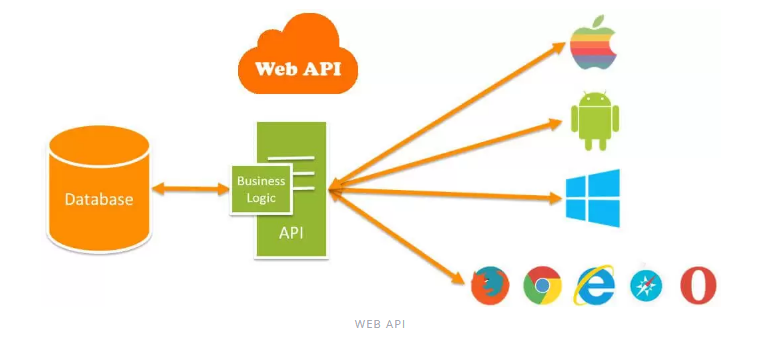
(21)

Langkah M dimana menggunakan hasil ekspektasi, menghitung mean dari tiap iterasi.

(22)

### **2.2.5. Application Programming Interface (API)**

Web API adalah sebuah software yang menyediakan layanan (fungsi-fungsi) spesifik yang dapat dipanggil melalui Web oleh berbagai jenis software lain. Web service adalah salah satu bentuk dari tradisional Web API, umumnya terdapat kontrak berupa sebuah dokumen Web Service Definition Language (WSDL) yang berisi fungsi-fungsi apa saja yang dapat dipanggil dan dieksekusi oleh Web service tersebut. Perkembangan selanjutnya adalah arsitektur REST (Representational State Transfer) API yang diperkenalkan oleh Fielding dan Taylor (2002). Sampai saat ini REST API sangat populer dan diimplementasikan pada banyak Web API.[[41]](#footnote-41) Arsitektur REST API ini akan dipakai untuk membuat API otentikasi suara untuk dites keamanannya. Skema API dapat dilihat pada **Gambar 4**.



**Gambar 6.** Skema API   
Sumber: Anugrah Sandi, Mengenal Apa itu Web API, https://www.codepolitan.com/mengenal-apaitu-web-api-5a0c2855799c8, 17 Mei 2020

### **2.2.6. Serangan API**

Penggunaan API semakin populer dan semakin juga penyerangan terhadap API, berikut merupakan contoh-contoh dari penyerangan API:

#### **2.2.6.1. Replay Attacks**

Serangan replay terjadi ketika penyerang menguping komunikasi jaringan yang aman, mencegatnya, dan kemudian secara curang menunda atau mengirimnya kembali untuk menyesatkan penerima agar melakukan apa yang diinginkan penyerang. Bahaya tambahan dari serangan replay adalah bahwa seorang penyerang bahkan tidak membutuhkan keterampilan tingkat lanjut untuk mendekripsi pesan setelah menangkapnya dari jaringan. Serangan itu bisa berhasil hanya dengan mengirim ulang semuanya.

#### **2.2.6.1. Injection Attacks**

Injection merupakan metode serangan dimana penyerang melakukan panggilan pada API yang memiliki SQL, NoSQL, LDAP, OS, atau perintah lain yang API atau program backend dibelakang-Nya menjalankan panggilan tersebut secara buta. Untuk melakukan serangan ini, penyerang harus menebak perintah apa yang dapat digunakan untuk API tersebut, bentuk panggilan, tipe data, dan banyak data.

### **2.2.7. Keamanan API**

Tentunya ada beberapa cara untuk meningkatkan untuk mempersulit melakukan penyerangan, diantaranya sebagai berikut:

#### **2.2.7.1. API Token (OAuth 2)**

OAuth2 memiliki kepanjangan open authorization yang dimana OAuth2 banyak digunakan dikalangan developer sebagai proses authorization sebuah aplikasi. Ketika ada aplikasi pihak ketiga akan mengakses API, API akan terlebih dahulu bertanya kepada authorization server, apakah token yang dikirim valid atau tidak, jika valid maka resource server akan memberikan resource yang dibutuhkan dan jika teryata tidak valid maka akan menampilkan bahwa aplikasi tersebut tidak berhak melakukan akses terhadap resource tersebut.

#### **2.2.7.1. API Key**

API Key merupakan metode pengaman lain yang dapat diterapkan untuk API. Metode ini menggunakan kode yang dienkripsi oleh penyedia API sebagai tanda pengenal untuk pengembang program agar dapat memakai servis yang diberikan oleh API. API Key ini harus di simpan seaman mungkin karena apabila penyerang mendapakan kode tersebut maka penyerang dapat melakukan panggilan ke API sebanyak penyerang inginkan.

### **2.2.8. Black Box Testing**

Untuk pengetesan perancangan ini, digunakan metode pengetesan *Black Box*. Dimana pengetesan fungsionalitas aplikasi perangkat lunak diuji tanpa memiliki pengetahuan tentang struktur kode internal, detail implementasi, dan jalur internal. Pengujian *Black Box* terutama berfokus pada input dan output aplikasi perangkat lunak dan sepenuhnya didasarkan pada persyaratan dan spesifikasi perangkat lunak. Metode ini juga dikenal sebagai Pengujian perilaku program.[[42]](#footnote-42) Tabel pengetesan menggunakan metode *Black Box* dapat dilihat pada **Lampiran 4.**

Kasus pengetesan pada rancangan ini dibuat berdasarkan pada data yang dibutuhkan program dan akan kemudian di analisa hasil output yang diberikan oleh rancangan tersebut.

### **2.2.9. C# (Visual C-Sharp)**

Visual C# atau C-Sharp adalah bahasa pemrograman sederhana yang digunakan untuk tujuan umum, dalam artian bahasa pemrograman ini dapat digunakan untuk berbagai fungsi misalnya untuk pemrograman server-side pada website, membangun aplikasi desktop ataupun mobile, pemrograman game dan sebagainya. Visual C# merupakan bahasa pemrograman utama dalam platform Microsoft.NET framework. Visual C# dianggap sebagai kombinasi antara efisiensi pemrograman C++, kesederhanaan pemrograman Java, dan penyederhanaan dari pemrograman Visual Basic.[[43]](#footnote-43)

### **2.2.10. ASP.NET**

ASP.Net adalah kumpulan teknologi dalam Framework .Net untuk membantu pengembangan aplikasi web yang menggunakan Object – Oriented secara dinamis, teknologi yang diciptakan oleh Microsoft untuk pemograman Internet yang lebih efisien. [[44]](#footnote-44)

### **2.2.11. Javascript**

Javascript adalah bahasa pemograman yang digunakan untuk berbagai tujuan, mulai dari meningkatkan fungsionalitas website hingga mengaktifkan permainan (games) dan software berbasis web.[[45]](#footnote-45) Kelebihan JavaScript tidak membutuhkan compiler karena web browser mampu menginterpretasikannya dengan HTML, website menjadi lebih interaktif dan juga mampu menarik perhatian lebih banyak pengunjung, Error atau kesalahan lebih mudah dicari dan diperbaiki. Kekurangan dari JavaScript yaitu, beresiko tinggi terhadap eksploitasi, dapat dimanfaatkan untuk mengaktifkan kode berbahaya di komputer pengguna Tidak selalu didukung oleh berbagai browser dan perangkat.

### **2.2.12. HTML**

HTML adalah singkatan dari HyperText Markup Language yaitu bahasa pemrograman standar yang digunakan untuk membuat sebuah halaman web, yang kemudian dapat diakses untuk menampilkan berbagai informasi di dalam sebuah penjelajah web Internet (Browser). HTML dapat juga digunakan sebagai *link* antara file-file dalam situs atau dalam komputer dengan menggunakan localhost, atau *link* yang menghubungkan antar situs dalam dunia internet.[[46]](#footnote-46)

Secara umum, fungsi dari HTML adalah untuk mengelola serangkaian data dan informasi sehingga suatu dokumen dapat diakses dan ditampilkan di internet melalui layanan web. Fungsi HTML yang lebih spesifik adalah untuk membuat halaman web, menampilkan berbagai informasi di dalam sebuah browser internet, dan membuat *link* menuju halaman web lain dengan kode tertentu (*hypertext*).

### **2.2.13. MySQL**

MySql adalah sebuah manajemen basis data yang menggunakan perintah dasar SQL (*Structured Query Language*). SQL (*Structured Query Language)* adalah bahasa pemrograman yang penting untuk dipahami karena dapat menjadi relasi bagi beberapa tabel dengan database maupun antar basis data itu sendiri.[[47]](#footnote-47)

## 2.3 Rekayasa Perangkat Lunak

Menurut Pressman Rekayasa Perangkat Lunak adalah pembuatan dan penggunaan prinsip-prinsip keahlian teknik untuk mendapatkan perangkat lunak yang ekonomis, handal, dan bekerja secara efisien pada mesin yang sesungguhnya. Rekayasa Perangkat Lunak mendirikan suatu fondasi untuk suatu proses perangkat lunak yang lengkap dengan mengidentifikasi sejumlah aktivitas kerangka kerja yang berlaku untuk semua proyek perangkat lunak, terlepas dari hal ukuran dan kompleksitas.[[48]](#footnote-48)

### **2.3.1. System Development Life Cycle (SDLC)**

*Systems development life cycle* (SDLC) adalah siklus atau tahapan yang digunakan dalam pembuatan/pengembangan suatu sistem informasi agar pengerjaan sistem berjalan secara terstruktur, efektif dan sesuai dengan tujuan yang diinginkan. Fungsi dari SDLC adalah sebagai sarana komunikasi antara tim pengembang dengan pemegang kepentingan dan dapat memberikan gambaran input dan output yang jelas dari satu tahap menuju tahap selanjutnya.[[49]](#footnote-49)

#### **2.3.1.1. Waterfall**

Model ini merupakan proses sekuensial linear. Waterfall model menawarkan sebuah pendekatan yang sistematis dan sekuensial dalam pengembangan piranti lunak yang dimulai dari label sistem dan perkembangannya dengan melalui tahap analisis, desain, coding dan testing.[[50]](#footnote-50)

Model waterfall adalah versi paling popular dari system development lifecycle model untuk sofware engineering. Model ini sering dianggap sebagai pendekatan klasik dalam daur hidup pengembangan sistem. Ada empat tahap utama dalam model ini, yaitu:

1. Analisis

Analisis adalah sebuah proses pengumpulan kebutuhan yang dikhususkan dan difokuskan dalam pembuatan piranti lunak.

1. Desain

Desain adalah sebuah proses yang menerjemahkan hasil dari analisis dalam bentuk representasi piranti lunak sehingga dapat dinilai kualitasnya sebelum proses *coding* dimulai.

1. *Coding*

*Coding* adalah proses dimana hasil dari desain diterjemahkan kembali dalam bentuk bahasa pemrograman yang dapat dimengerti oleh mesin.

1. *Testing*

*Testing* dilakukan setelah *coding* selesai dilakukan. Testing digunakan untuk mengetahui kesalahan yang tidak terdeteksi sebelum atau hasil dari proses yang tidak diinginkan.

# BAB III RANCANGAN

## 3.1. Rancangan Sistem

Sistem yang dirancang adalah sistem *login* yang dapat bertahan dari serangan digital. Sistem ini diintegrasikan dengan API otentikasi suara dengan guna menambah tingkat kesulitan bagi penyerang, sehingga data yang berada dalam rancangan tetap aman.

Proses Perancangan sistem ini menggunakan metode *System DevelopmentLife Cycle* (SDLC) dengan model Waterfall. Pada sub-bab berikut ini akan dijelaskan mengenai Tahap perancangan

### **3.1.1. Tahap Analisis**

Dalam tahap analisis untuk perancangan Sistem Rekomendasi dan Reservasi Hotel berbasis web dibagi dalam beberapa analisis seperti analisis kebutuhan kasus, analisis fitur aplikasi sejenis, dan analisis kebutuhan perangkat yang akan dijelaskan sebagai berikut:

#### **3.1.1.1. Analisis Kebutuhan Khusus**

Perancangan Sistem *login* dengan integrasi API otentikasi suara bertujuan untuk menambah keamanan untuk aplikasi website dengan masukan suara pengguna sehingga pengguna memerlukan mikrofon untuk memberikan input suaranya.

#### **3.1.1.2. Analisis Fitur Aplikasi Sejenis**

Pada saat ini sudah terdapat banyak aplikasi atau sistem yang menggunakan sistem *login* menggunakan otentikasi *multi-factor*. Fitur-fitur yang dimiliki oleh aplikasi tersebut juga memiliki enkripsi, otentikasi, dan pengiriman data. Dalam perancangan Sistem *login* ini memiliki fitur enkripsi password, fitur otentikasi, dan fitur pengiriman data. Fitur otentikasi untuk rancangan ini menggunakan suara pengguna menggunakan API otentikasi suara.

Untuk sistem keamanan yang sudah pernah dirancangan oleh peneliti lain dan perbandingannya dengan sistem yang akan dibuat akan dijelaskan dalam table berikut:

**Tabel 2**. Perbandingan Penelitian

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Judul Penelitian** | **Nama Peneliti** | **Metode** | **Kriteria** | **Fitur** |
| 1 | Security issues of Internet-based biometric authentication systems: risks of Man-in-the-Middle and BioPhishing on the example of BioWebAuth | Christian Zeitza , Tobias Scheidata , Jana Dittmanna , Claus Vielhauera , Elisardo González Agullab , Enrique Otero Murasb , Carmen García Mateob dan José L. Alba Castrob | Man, in the Middle | Username, Password. | Memberikan penelitian keamanan yang beresiko tinggi terhadap serangan tertentu. |
| 2 | Security Evaluation of Biometric Authentication Systems Under Real Spoofing Attacks | Battista Biggio, Zahid Akhtar, Giorgio Fumera, Gian Luca Marcialis, dan Fabio Roli | Spoofing menggunakan biometrik yang di generate. | Muka dan suara | Memberikan penelitian ketahanan program otentikasi terhadap serangan. |
| 3 | Analisis Security Voice Authenticator pada Sistem Login Two Factor Authentication | Gilbert Alexandro Onggo | Serangan Injeksi dan Replay | Username, Enkripsi Password, suara pengguna. | Memberikan Analisa kerentanan rancangan API terhadap serangan Injeksi dan Replay. |

#### **3.1.1.3. Analisis Kebutuhan Perangkat**

Dalam perancangan Sistem Login berbasis web, perangkat keras dan perangkat lunak yang dibutuhkan oleh perancang untuk melakukan proses perancangan adalah sebagai berikut:

* 1. Personal Computer dengan spesifikasi prosesor intel core i7-8700K 3.70GHz, GTX 1080 8GB,Ram 16GB, SSD 250GB, dan HDD 2TB
  2. Windows 10 Enterprise 64-bit
  3. Bahasa pemrograman C# (Visual C-Sharp)
  4. Framework ASP.NET
  5. MySQL
  6. Javascript
  7. Google Chrome

Sedangkan untuk pihak pengguna, diperlukan perangkat keras dan perangkat lunak sebagai berikut:

1. Smartphone atau perangkat komputer lain
2. Browser
3. Jaringan Internet

### **3.1.2. Tahap Perancangan**

Pada Tahap ini di jelaskan pembentukan rancangan program yang akan dites keamanannya.

#### **3.1.2.1. Rancangan Skema Sistem**

Skema Sistem atau Diagram Skema adalah gambar konsep desain yang merepresentasikan unsur-unsur sistem yang ditampilkan dengan menggunakan simbol grafis bukan dengan bentuk yang sesungguhnya.[[51]](#footnote-51) Skema sistem pada perancangan sistem ini dapat dilihat pada **Gambar 7 Lampiran 5**

#### **3.1.2.2. Rancangan Diagram Alur (Flowchart)**

Rancangan Diagram Alur atau *Flowchart* adalah suatu bagan dengan simbol-simbol tertentu yang memperlihatkan urutan atau langkah-langkah dari suatu program dan hubungan antar proses beserta pernyataannya.[[52]](#footnote-52) Flowchart sistem pada perancangan Sistem Rekomendasi dan Reservasi Hotel dapat dilihat pada **Gambar 8 Lampiran 6.**

#### **3.1.2.3. Rancangan State Transition Diagram (STD)**

*State Diagram* atau *State Transition Diagram* adalah diagram yang digunakan untuk mendeskripsikan semua kondisi yang mungkin muncul pada perilaku sistem.[[53]](#footnote-53) Rancangan State Transition Diagram pada rancangan akan dijelaskan sebagai berikut:

##### **3.1.2.3.1 Modul *Login* dan Registerasi**

Dalam modul *login*, setiap pengguna dapat melakukan proses *login* menggunakan basis data yang berisi data pengguna dengan melakukan pencocokan data. Jika pengguna tidak memiliki akun, maka pengguna harus melakukan registrasi. Pengguna akan diminta memasukkan suara pada saat proses *login* atau registrasi. State Transition Diagram modul ini dapat dilihat pada **Gambar** **9 Lampiran 7.**

##### **3.1.2.3.2 Modul API**

Dalam modul API, *input* suara dari modul *login* dan registrasi akan diterima dan akan dikirim ke program otentikasi secara otomatis. Modul ini bertanggung jawab atas pengiriman data suara pengguna ke program. State Transition Diagram modul ini dapat dilihat pada **Gambar 10 Lampiran 8.**

#### **3.1.2.4. Rancanagan Data Flow Diagram (DFD)**

Data Flow Diagram (DFD) adalah suatu langkah atau metode untuk membuat sebuah perancangan sistem yang berorientasi pada alur data yang bergerak kesebuah sistem lainnya.[[54]](#footnote-54) Rancangan Data Flow Diagram pada perancangan sistem ini terdiri dari 2 diagram, yaitu sebagai berikut:

##### **3.1.2.4.1. Context Diagram**

*Context Diagram* atau Diagram Konteks adalah sebuah bagian level dari *Data Flow Diagram* yang digunakan untuk menetapkan kontesk serta batasan sistem pada sebuah pemodelan. Context Diagram pada perancangan sistem ini dapat dilihat pada **Gambar 11 Lampiran 9**.

##### **3.1.2.4.2. Data Flow Diagram**

*Data Flow Diagram* (DFD)merupakan pengembangan dari *Context Diagram* yang bertujuan untuk menggambarkan secara lebih rinci proses-proses yang terdapat pada *Context Diagram*. *Data Flow Diagram* (DFD)pada perancangan sistem ini dapat dilihat pada **Gambar 12 Lampiran 10**.

#### **3.1.2.5. Rancangan Diagram Hierarki Sistem**

Diagram Hierarki Sistem adalah suatu tingkatan atau urutan yang abstraksi menjadi struktur pohon yang bertujuan untuk menunjukan keseluruhan bagian sistem dari tingkat yang paling umum ke tingkat yang paling spesifik. Diagram Hierarki Sistem pada perancangan sistem ini dapat dilihat pada **Gambar 13 Lampiran 11.**

#### **3.1.2.6. Rancangan Basis Data**

Basis Data atau *Database* adalah kumpulan informasi yang disimpan dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari database.[[55]](#footnote-55) Basis data berfungsi untuk menampung atau menyimpan data-data dimana masing-masing data yang ada pada tabel atau file tersebut saling berhubungan satu sama lainnya. Rancangan basis data yang akan ditampilkan akan dibagi menjadi tiga bagian sebagai berikut:

##### **3.1.2.6.1. Hubungan Antar Entitas**

Hubungan Antar Entitas atau *Entity Relationship Diagram* (ERD) adalah suatu model informasi untuk menjelaskan suatu hubungan antar data dan basis data yang digambarkan dengan grafik dan juga notasi dengan model data konseptual.[[56]](#footnote-56) Hubungan antar entitas yang akan dibuat pada perancangan ini terdiri hanya dua tabel yaitu tabel User dan tabel API dapat dilihat pada **Gambar 14 Lampiran 12.**

##### **3.1.2.6.2.Hubungan antar Tabel**

Hubungan Antar Tabel atau Relasi merupakan hubungan yang terjadi pada suatu tabel dengan lainnya yang mempresentasikan hubungan antar objek di dunia nyata dan berfungsi untuk mengatur mengatur operasi suatu database.[[57]](#footnote-57) Hubungan antar tabel pada perancangan sistem ini dapat dilihat pada **Gambar 15 Lampiran 13**.

##### **3.1.2.6.3. Spesifikasi Tabel**

Rancangan basis data pada Sistem *Login* dengan integrasi API otentikasi suara akan dibuat dalam bentuk tabel yang menampilkan spesifikasi tabel. Kedua tabel yang telah dijelaskan pada bagian sebelumnya akan dijelaskan dengan rinci beserta spesifikasi tabel seperti nama field, user field, dan atribut lainnya dari tabel tersebut. Spesifikasi tabel untuk perancangan sistem ini dapat dilihat pada **Tabel 5 Lampiran 14** sampai dengan **Tabel 6 Lampiran 14.**

#### **3.1.2.7. Rancangan Tampilan Antarmuka**

Rancangan tampilan antar muka pada perancangan Sistem Login dengan integrasi API otentikasi suara akan dijelaskan sebagai berikut:

##### **3.1.2.7.1. Rancangan Tampilan Halaman Login**

Halaman *Login* merupakan halaman yang digunakan untuk mengecek apakah email dan password yang dimasukkan oleh pengguna benar atau tidak. Jika email dan password tersebut valid, maka akan langsung dialihkan ke halaman Input Suara. Tampilan halaman Login dapat di lihat pada **Gambar 16 Lampiran 15**.

##### **3.1.2.7.2. Rancangan Tampilan Halaman Register**

Halaman Register merupakan halaman yang digunakan untuk membuat sebuah akun jika pengguna belum memiliki akun untuk *login*. Tampilan halaman Register dapat dilihat pada **Gambar 17 Lampiran 16**.

##### **3.1.2.7.3. Rancangan Tampilan Halaman Input Suara Login**

Halaman Input Suara merupakan halaman yang digunakan untuk melakukan input suara user dengan text yang akan digunakan oleh program otentikator suara. Tampilan halaman Input suara dapat dilihat pada **Gambar 18 Lampiran 17**.

##### **3.1.2.7.4. Rancangan Tampilan Halaman Input Suara Registrasi**

Halaman Input Suara merupakan halaman yang digunakan untuk melakukan input suara user dengan text yang akan digunakan oleh program otentikator suara. Tampilan halaman Input suara dapat dilihat pada **Gambar 19 Lampiran 18**.

##### **3.1.2.7.5. Rancangan Tampilan Halaman Hasil *Login***

Halaman Hasil *Login* merupakan halaman yang pakai untuk memberikan informasi bahwa user berhasil *login* dan terotentikasi oleh program otentikasi suara. Tampilan halaman Berhasil Login dapat dilihat pada **Gambar 20 Lampiran 19**.

## 3.2. Pembuatan Sistem

Tahap yang dilakukan setelah membuat rancangan sistem adalah pembuatan sistem. Urutan pembuatan sistem akan dibagi menjadi tiga langkah utama yaitu:

1. Membuat sistem login berbasis website yang dapat digunakan untuk melakukan penyerangan. Basis data menggunakan perangkat lunak MoggoDB untuk menyimpan data username dan password.
2. Membuat API yang memberikan servis otentikasi dan membuat model untuk melakukan pengenalan pada setiap suara oleh user. Model dibuat menggunakan ekstraksi fitur MFCC untuk pemrosesan data audio agar dapat dilakukan pelatihan pada GMM, yang kemudian di simpan.
3. Membuat jembatan koneksi antara sistem login dengan API agar dapat mengirim data dengan satu sama lain. Fitur keamanan yang dipakai merupakan token yang dibatasi dengan waktu.

Pembuatan sistem ini dilakukan menggunakan perangkat keras dan perangkat lunak. Perangkat keras yang digunakan adalah sebagai berikut:

1. Processor : Intel Core i7-8700K @3.70GHz, 3696 Mhz, 6 Core(s), 12 Logical Processor(s)
2. Hard Drives : SSD 250GB + HDD 2TB
3. RAM : DDR4 16 GB
4. VGA : NVIDIA GeForce GTX 1080 8 GB
5. Layar : Monitor 27.0”

Perangkat lunak yang digunakan adalah sebagai berikut:

1. MonggoDB
2. Postman
3. WireShark
4. SQLAlchemy
5. Flask
6. Python 3.7
7. Microsoft Visual Studio Code

### **3.2.1. Tahap Perancangan**

c

# DAFTAR PUSTAKA

Anil K. Jain, Karthik Nandakumar, Abhishek Nagar, Biometric template security, *EURASIP Journal on Advances in Signal Processing*, Januari 2008.

Debnath Bhattacharyya, Rahul Ranjan , Farkhod Alisherov A. , dan Minkyu Choi., Biometric Authentication: A Review, *International Journal of u- and e- Service, Science and Technology Vol. 2, No. 3*, September 2009.

Christian Zeitz , Tobias Scheidat , Jana Dittmann , Claus Vielhauer , Elisardo González Agull, Enrique Otero Muras , Carmen García Mateo dan José L. Alba Castro, “Security issues of Internet-based biometric authentication systems: risks of Man-in-the-Middle and BioPhishing on the example of BioWebAuth”, Proceedings of SPIE - The International Society for Optical Engineering, Februari 2008

Fisher, Sharon, “OS/2 EE to Get 3270 Interface Early”, San Francisco: Info World, Vol 10, Nomor 34, Agustus 1988.

Sarah Coble, 10 September 2020, API Attacks Increase During Lockdown, https://www.infosecurity-magazine.com/news/api-attacks-increase-during/, 15 Mei 2020.

Ding Wang dan Ping Wang, “Two Birds with One Stone: Two-Factor Authentication with Security Beyond Conventional Bound”, IEEE Transactions on Dependable and Secure Computing, Vol. 5, Nomor 4, 1 Juli 2018.

Battista Biggio, Zahid Akhtar, Giorgio Fumera, Gian Luca Marcialis, dan Fabio Roli., “Security Evaluation of Biometric Authentication Systems”, IET Biometrics, Vol 1, Maret 2012.

Margaret Rouse, Accessed 15 September 2020, MD5 Definition, https://searchsecurity.techtarget.com/definition/MD5,.

Mary Cindy, Zhao Shun, Shubra Deb, “Security Analysis of MD5 Algorithm in Password Storage”, Applied Mechanics and Materials, Vol 347, Agustus 2013.

R. Rivest, “The MD5 Message-Digest Algorithm”, RFC1321, April 1992.

Anonymous, The MD5 cryptographic hash function, https://www.iusmentis.com/technology/hashfunctions/md5/, 20 September 2020.

Sri Waluyanti, Buku Direktorat PSMK Untuk Tehnik Audio Video, (Jakarta: Direktorat Pembinaan SMK, 2008)

Bill Poser, Audio Data, http://billposer.org/Linguistics/Computation/LectureNotes/AudioData.html, 9 September 2020.

Cliff Truesdell, Mastering Digital Audio Production: The Professional Music Workflow with Mac OS X, (Vancouver: Wiley Publishing Inc, 2007).

Herru Darmadi, Merancang Web API yang Mudah Dieksplorasi, https://binus.ac.id/knowledge/2018/05/merancang-web-api-yang-mudah-dieksplorasi/, 31 Mei 2018.

Luqman Hakim, Bahasa Pemrograman C# dan EmguCV, (Yogyakarta: DEEPUBLISH, 2018)

Hanry Ham, S.Kom., M.Eng., ASP.Net, https://socs.binus.ac.id/2018/12/20/asp-net/, 18 September 2020.

Ariata C., Apa Itu JavaScript, https://www.hostinger.co.id/tutorial/apa-itu-javascript/, Januari 2019.

Admin Nawadwipa, Pengertian Dan Fungsi HTML¸ https://www.nawadwipa.co.id/pengertian-dan-fungsi-html-hypertext-markup-language/, Agustus 2020.

Anonimous, Mengenal Apa Itu MySQL, Fungsi, dan Cara Kerjanya, https://idcloudhost.com/mengenal-apa-itu-mysql-fungsi-dan-cara-kerjanya-lengkap/, 18 September 2020.

Roger S.Pressman, Software Engineering, 7th Edition, (New York: McGraw-Hill, 2008).

Anonimus, Gambar Skematik Diagram LRT Bandung Raya, https://kampungdrafter.com/gambar-skematik-diagram-lrt-bandung-raya/#:~:text=Gambar%20digaram%20skematik%20adalah%20gambar,bukan%20dengan%20bentuk%20yang%20sesungguhnya., 25 September 2020.

Ali Ridho Barakbah, Tita Karlita, dan Ahmad Syauqi Ahsan, Logika dan Algoritma, (Surabaya: Departemen Teknik Informatika dan Komputer Politeknik Elektronika Negeri Surabaya, 2013).

Nova V. Siringoringo, Pengertian, State Diagram, https://medium.com/state-diagram/pengertian-state-diagram-67d3a4069da3, 3 Februari 2020.

Ansori, Pengertian DFD (Data Flow Diagram): Fungsi, Simbol, dan Contohnya, https://www.ansoriweb.com/2020/03/pengertian-dfd.html, 25 September 2020.

Dosen Pendidikan 3, Basis Data Adalah, https://www.dosenpendidikan.co.id/basis-data/, 25 September 2020.

Nabil Tediamaja, Pengertian, Fungsi, Contoh dan Cara Membuat ERD (Entity Pelationship Diagram), https://bilabil.com/contoh-erd/#Pengertian\_ERD\_Entity\_Pelationship\_Diagram, 25 September 2020.

Fatkhan Amirul Huda, Relasi Antar Tabel dalam Database, http://fatkhan.web.id/relasi-antar-tabel-dalam-database/#:~:text=Mempunyai%20pengertian%20%E2%80%9CSetiap%20baris%20data,orang%20tua%20begitu%20juga%20sebaliknya., 25 September 2020.

Cliff Truesdell, Mastering Digital Audio Production: The Professional Music Workflow with Mac OS X, (Vancouver: Wiley Publishing Inc, 2007).

Abdel-rahman Mohamed, Deep Neural Network Model for ASR,Toronto: Department of Computer Science University Toronto (Thesis of Doctor Philosophy Graduate), 2014.

Ali Mustofa, Sistem Pengenalan Penutur dengan Metode Mel-Frequency Wrapping, Jurnal Teknik Elektro, Vol.7, No.2, (September, 2007).

Marwa A.Nasr, et.al., Speaker identification based on ormalized pitch frequency and Mel Frequency Cepstral Coefficient, International Journal of Speech Technology , Vol.21, Issue 4,(Desember, 2018).

Agus Buono, Wisnu Jatmiko, dan Benyamin Kusumoputro, Perluasan Metode MFCC 1D ke 2D sebagai Ekstraksi Ciri pada Sistem Identifikasi Pembicara Menggn Hidden Markov Model (HMM), Makara, Sains, Vol. 13, Nomor 1, (April, 2009).

Bella, M. Prof. Dr. Ir. Dyah Erny Herwindiati and S. M. Janson Hendryli, Aplikasi Mobile Electronic Wallet Berbasis Android Dengan Fitur Voice Authentication, 2018.

Martin A Moskowitz, A Course in Complex Analysis in One Variable, (Singapore: World Scientific Publishing Co., 2002).

Gu, Juan, Jun Chen, Qiming Zhou, Hongwei Zhang. *Gaussian Mixture Model of Texture for Extracting Residential Area from High-Resolution Remotely Sensed Imagery*, (ISPRS Workshop on Updating Geo-spatial Databases with Imagery & The 5th ISPRS Workshop on DMGISs), Agustus 2007.

# LAMPIRAN 1

Pesan: **KRIPTOGRAFI**

Konversi setiap 4 byte kedalam word:

KRIP : 5049524b

TOGR : 52471554

Mengolah final bits, menambahkan padding bit 1 kemudian sisanya bit 0

Panjang pesan = 11

panjang pesan % 4 = 3, Sisa Byte: 'AFI' (1 byte padding) = 80494641

Proses: padding bit (dalam decimal):

Menghadle akhir dari teks dan memulai pading bit: = 1346982475, 1380405076, -2142681535

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

padding ke 448 bits (mod 512 bits) = 14 word (mod 15 word) = 1346982475, 1380405076, -2142682535, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

Menambahkan pesan dengan Panjang 64-bit = 1346982475, 1380405076, -2142681535, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 88, 0

88 muncul karena panjang pesan KRIFTOGRAFI = 11 karakter, Ialu dikalikan dengan 8 bit = 88

Block untuk setiap Variabel:

Permulaan, a (0x067452301) = 67452301

b (0x0efcdab89) = efcdab89

c (0x098badcfe) = 98badcfe

d (0x010325476) = 10325476

Padding bit dimasukkan kedalam array, konversikan ke hexa:

x[0]=5049524b

x[1]=52474f54

x[2]=80494641

x[3]=00000000

x[4]=00000000

x[5]=00000000

x[6]=00000000

x[7]=00000000

x[8]=00000000

x[9]=00000000

x[10]=00000000

x[11]=00000000

x[12]=00000000

x[13]=00000000

x[14]=00000058

x[15]=00000000

Konstanta:

S11 = 7 S12 = 12 S13 = 17 S14 = 22

S21 = 5 S22 = 9 S23 = 14 S24 = 20

S31 = 4 S32 = 11 S33 = 16 S34 = 23

S41 = 6 S42 = 10 S43 = 15 S44 = 21

Round 1:

Lakukan operasi [abcd k s i]

a = b + ((a+F(b,c,d) + X[k] + T[i]) <<< s)

Lakukan 16 operasi berikut ini setiap roundnya:

[ABCD 0 7 1 ][DABC 1 12 2 ][CDAB 2 17 3 ][BCDA 3 22 4 ]

[ABCD 4 7 5 ][DABC 5 12 6 ][CDAB 6 17 7 ][BCDA 11 22 8 ]

[ABCD 8 7 9 ][DABC 9 12 10][CDAB 10 17 11 ][BCDA 11 22 12]

[ABCD 12 7 13][DABC 13 12 14][CDAB 14 17 15 ][BCDA 15 22 16]

Lakukan a = b +((a + F(b,c,d) + X[k] + T[i]) <<< s). (Catatan: <<< Berarti rotate left)

Fungsi yang digunakan pada masukan F(b, c, d) = (b and C) or ((not b)and d)

FF (a, b, c, d, x[0], S11, 0xd76aa478)

FF: a=67452301, b=efcdab89, c=98badcfe, d=10325476, xk=5049524b, s=7, ti=d76aa478

F = (efcdab89 & 98badcfe)|((~efcdab89) & 10325476)

= (88888888) | (10325476)

= 98BADCFE

A+F(b, c, d)+ xk + ti = 227B3F6C2

* 227B3F6C2 = 1000100111101100111111011011000010

S^n(X) = (X <<n) OR (X>>32-n)  
X= 1000100 111101100111111011011000010

100010011 1101100111111011011000010

X << 7 = 111101100111111011011000010 0000000  
X >> 25 = 0000000000000000000000000 100010011

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ OR

= 1111011001111110110110000100010011

= 3D9FB6113

Efcdab89 + 3D9FB6113 = C9C90C9C

A = b + ((a + F(b, C, d) + X[k] + T[i]) <<< s)

FF: b + (... <<< 7) = c9c90c9c

FF (d, a, b, c, x[1], S12, 0xe8c7b756)

FF: a=10325476, b = c9c90c9c, c=efcdab89, d=98badcfe, xk=52474f54, s=12, ti=e8c7b756

F(c9c90c9c, efcdab89, 98badcfe) = d9fbd8ea

a + F(b, c, d) + xk + ti = 253d340a

* 2253D340A = 1000100101001111010011010000001010  
  S^n(X) = (X <<n) OR (X >> 32-n)  
  n = 7

X = 100010010100 1111010011010000001010

10001001010011 11010011010000001010

X << 12 = 1111010011010000001010 000000000000

X >> 20 = 00000000000000000000 10001001010011

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ OR

= 1111010011010000001010001001010011

= 3D340A253

C9c90c9c + 3D340A253 = 9D09AEEF

FF: b + (... <<< 12) = 9d09aeef

FF (c, d, a, b, x[2], S13, 0x242070db)

FF: a=98badcfe, b=9d09aeef, c=c9c90c9c, d=efcdab89, xk=80494641, s=17, ti=24070db

F(9d09aeef, c9c90c9c, efcdab89) = ebcd0d8c

a+F(b, c, d) + xk + ti = 28f1a1a6 228F1A1A6

FF:b+(... <<< 17) = e05600d2

FF (b, c, d, a, x[3], S14, 0xc1bdceee)

FF: a=efcdab89, b=e05600d2, c=9d09aeef, d=c9c90c9c, xk=00000000, s=22, ti=c1bdceee

F(e05600d2, 9d09aeef, c9c90c9c) = 89890cce

a+F(b,c,d) + xk+ti=3b148745

FF: b+(... <<< 22) = b1a4c5f3

FF (a, b, c, d, x[4], S11, 0xf57c0faf)

FF: a=c9c90c9c, b=b1a4c5f3, c=e05600d2, d=9d09aeef, xk=00000000, s=7, ti=f57c0faf

F(b1a4c5f3, e05600d2, 9d09aeef) = ac0d2ade

a+F(b,c,d) + xk+ti=6b524729

FF: b+(... <<< 7) = 5ac85aa8

FF (d, a, b, c, x[5], S12, 0x4787c62a)

FF: a=9d09aeef, b=5ac85aa8, c=b1a4c5f3, d=e05600d2, xk=00000000, s=12, ti=4787c62a

F(5ac85aa8, b1a4c5f3, e05600d2) = b09640f2

a+F(b,c,d) + xk+ti=9527b60b

FF: b+(... <<< 12) = d62913fa

FF (c, d, a, b, x[6], S13, 0xa8304613)

FF: a=e05600d2, b=d62913fa, c=5ac85aa8, d=b1a4c5f3, xk=00000000, s=17, ti=a8304613

F(d62913fa, 5ac85aa8, b1a4c5f3) = 738cd6a9

a+F(b,c,d) + xk+ti= fc131d8e

FF: b+(... <<< 17) = 11470c20

FF (b, c, d, a, x[7], S14, 0xfd469501)

FF: a=b1a4c5f3, b=11470c20, c=d62913fa, d=5ac85aa8, xk=00000000, s=22, ti=fd469501

F(11470c20, d62913fa, 5ac85aa8) = 5a8952a8

a+F(b,c,d) + xk+ti= 0974ad9c

FF: b+(... <<< 22) = 7849694b

FF (a, b, c, d, x[8], S11, 0x698098d8)

FF: a=5ac85aa8, b=7849694b, c=11470c20, d=d62913fa, xk=00000000, s=7, ti=698098d8

F(7849694b, 11470c20, d62913fa) = 96611ab0

a+F(b,c,d) + xk+ti= 5aaa0e30

FF: b+(... <<< 7) = cd508178

FF (d, a, b, c, x[9], S12, 0x8b44f7af)

FF: a=d62913fa, b=cd508178, c=7849694b, d=11470c20, xk=00000000, s=12, ti=8b44f7af

F(cd508178, 7849694b, 11470c20) = 58470d48

a+F(b,c,d) + xk+ti= b9b518f1

FF: b+(... <<< 12) = 1edf9d13

FF (c, d, a, b, x[10], S13, 0xffff5bb1)

FF: a=11470c20, b=1edf9d13, c=cd508178, d=7849694b, xk=00000000, s=17, ti=ffff5bb1

F(1edf9d13, cd508178, 7849694b) = 6c50e158

a+F(b,c,d) + xk+ti= 7d974929

FF: b+(... <<< 17) = b1329841

FF (b, c, d, a, x[11], S14, 0x895cd7be)

FF: a=7849694b, b=b1329841, c=1edf9d13, d=cd508178, xk=00000000, s=22, ti=895cd7be

F(b1329841, 1edf9d13, cd508178) = 5c529939

a+F(b,c,d) + xk+ti= 5df8da42

FF: b+(... <<< 22) = 41ca1677

FF (a, b, c, d, x[12], S11, 0x6b901122)

FF: a=cd508178, b=41ca1677, c=b1329841, d=1edf9d13, xk=00000000, s=7, ti=6b901122

F(41ca1677, b1329841, 1edf9d13) = 1f179941

a+F(b,c,d) + xk+ti= 57f82bdb

FF: b+(... <<< 7) = 3de00422

FF (d, a, b, c, x[13], S12, 0xfd987193)

FF: a=1edf9d13, b=3de00422, c=41ca1677, d=b1329841, xk=00000000, s=12, ti=fd987193

F(3de00422, 41ca1677, b1329841) = 81d29c63

a+F(b,c,d) + xk+ti= 9e4aab09

FF: b+(... <<< 12) = e8909e06

FF (c, d, a, b, x[14], S13, 0xa679438e)

FF: a=b1329841, b=e8909e06, c=3de00422, d=41ca1677, xk=00000058, s=17, ti=a679438e

F(e8909e06, 3de00422, 41ca1677) = 29ca0473

a+F(b,c,d) + xk+ti= 8175e09a

FF: b+(... <<< 17) = a9c5a0f1

FF (b, c, d, a, x[15], S14, 0x49b40821)

FF: a=41ca1677, b=a9c5a0f1, c=e8909e06, d=3de00422, xk=00000000, s=22, ti=49b40821

F(a9c5a0f1, e8909e06, 3de00422) = bca08402

a+F(b,c,d) + xk+ti= 481ea29a

FF: b+(... <<< 22) = 5057a899

Round 2:

Lakukan operasi [abcd k s i]

a = b + ((a+G(b,c,d) + X[k] + T[i]) <<< s)

Lakukan 16 operasi berikut ini setiap roundnya:

[ABCD 1 5 17][DABC 6 9 18 ][CDAB 11 14 19 ][BCDA 0 20 20 ]

[ABCD 5 5 21][DABC 10 9 22 ] [CDAB 15 14 23 ][BCDA 4 20 24 ]

[ABCD 9 5 25][DABC 14 9 26 ][CDAB 3 14 27 ][BCDA 8 20 28 ]

[ABCD 13 5 29][DABC 2 9 30 ][CDAB 17 14 31 ][BCDA 12 20 32 ]

Lakukan a = b +((a + G(b,c,d) + X[k] + T[i]) <<< s). (Catatan: <<< Berarti rotate left)

Fungsi yang digunakan pada masukan G(b, c, d) = (b and d) or (c and (not d))

GG (a, b, c, d, x[1], S21, 0xf61e2562)

GG: a=3de00422, b=5057a899, c=a9c5a0f1, d=e8909e06, xk=52474f54, s=5, ti=f61e2562

G(5057a899, a9c5a0f1, e8909e06) = 4155a8f1

a+G(b,c,d) + xk+ti= c79b21c9

GG: b+(... <<< 5) = 43bbe1d1

GG (d, a, b, c, x[6], S22, 0xc040b340)

GG: a=e89090e06, b=43bbe1d1, c=5057a899, d=a9c5a0f1, xk=00000000, s=9, ti=c040b340

G(43bbe1d1, 5057a899, a9c5a0f1) = 5193a8d9

a+G(b,c,d) + xk+ti= fa64fa1f

GG: b+(... <<< 8) = 0db021c5

GG (c, d, a, b, x[11], S23, 0x265e5a51)

GG: a=a9c5a0f1, b=0db021c5, c=43bbe1d1, d=5057a899, xk=00000000, s=14, ti=265e5a51

G(0db021c5, 43bbe1d1, 5057a899) = 03b861c1

a+G(b,c,d) + xk+ti= d3dc5d03

GG: b+(... <<< 14) = 24f116bc

GG (b, c, d, a, x[0], S24, 0x9b6c7aa)

GG: a=5057a899, b=24f116bc, c=0db021c5, d=43bbe1d1, xk=5049524b, s=20, ti=e9b6c7aa

G(24f116bc, 0db021c5, 43bbe1d1) = 0cd10094

a+G(b,c,d) + xk+ti= 9708c322

GG: b+(... <<< 20) = 571a8748

GG (a, b, c, d, x[5], S21, 0xd62f105d)

GG: a=43bbe1d1, b=571a8748, c=24f116bc, d=0db021c5, xk=00000000, s=5, ti=d62f105d

G(571a8748, 24f116bc, 0db021c5) = 25511778

a+G(b,c,d) + xk+ti= 3f3c09a6

GG: b+(... <<< 5) = 3e9bbc0f

GG (d, a, b, c, x[10], S22, 0x2441453)

GG: a=0db021c5, b=3e9bbc0f, c=571a8748, d=24f116bc, xk=00000000, s=9, ti=02441453

G(3e9bbc0f, 571a8748, 24f116bc) = 779b954c

a+G(b,c,d) + xk+ti= 878fcb64

GG: b+(... <<< 9) = 5e32851e

GG (c, d, a, b, x[15], S23, 0xd8ale681)

GG: a=24f116bc, b=5e32851e, c=3e9bbc0f, d=571a8748, xk=00000000, s=14, ti=d8a1e681

G(5e32851e, 3e9bbc0f, 571a8748) = 7e93bd0f

a+G(b,c,d) + xk+ti= 7c26ba4c

GG: b+(... <<< 14) = 0cc5a427

GG (b, c, d, a, x[4], S24, 0xe7d3fbc8)

GG: a=571a8748, b=0cc5a427, c=5e32851e, d=3e9bbc0f, xk=00000000, s=20, ti=e7d3fbc8

G(0cc5a427, 5e32851e, 3e9bbc0f) = 0ef72529

a+G(b,c,d) + xk+ti= 6f74af1e

GG: b+(... <<< 20) = 7dd440f6

GG (a, b, c, d, x[9], S21, 0x21e1cde6)

GG: a=3e9bbc0f, b=8f3e5d29, c=0cc5a427, d=5e32851e, xk=00000000, s=5, ti=21e1cde6

G(8f3e5d29, 0cc5a427, 5e32851e) = 0ef72529

a+G(b,c,d) + xk+ti= 6f74af1e

GG: b+(... <<< 5) = 7dd440f6

GG (d, a, b, c, x[14], S22, 0xc33707d6)

GG: a=5e32851e, b=7dd440f6, c=8f3e5d29, d=0cc5a427, xk=00000058, s=9, ti=c33707d6

G(7dd440f6, 8f3e5d29, 0cc5a427) = 8ffe592e

a+G(b,c,d) + xk+ti= b167e67a

GG: b+(... <<< 9) = 4da13658

GG (c, d, a, b, x[3], S23, 0xf4d50d87)

GG: a=0cc5a427, b=4da13658, c=7dd440f6, d=8f3e5d29, xk=00000000, s=14, ti=f4d50d87

G(4da13658, 7dd440f6, 8f3e5d29) = 7de014de

a+G(b,c,d) + xk+ti= 7f7ac68c

GG: b+(... <<< 14) = ff445636

GG (b, c, d, a, x[8], S24, 0x455a14ed)

GG: a=8f3e5d29, b=ff445636, c=4da13658, d=7dd440f6, xk=00000000, s=20, ti=455a14ed

G(ff445636, 4da13658, 7dd440f6) = 7d65763e

a+G(b,c,d) + xk+ti= 51fde854

GG: b+(... <<< 20) = 84897614

GG (a, b, c, d, x[13], S21, 0xa9e3e905)

GG: a=7dd440f6, b=84897614, c=ff445636, d=4da13658, xk=00000000, s=5, ti=a9e3e905

G(84897614, ff445636, 4da13658) = b6c57636

a+G(b,c,d) + xk+ti= de7da031

GG: b+(... <<< 5) = 543d7c4f

GG (d, a, b, c, x[2], S22, 0xfcefa3f8)

GG: a=4da13658, b=543d7c4f, c=84897614, d=ff445636, xk=80494641, s=9, ti=fcefa3f8

G(543d7c4f, 84897614, ff445636) = 548d7406

a+F(b,c,d) + xk+ti= 1f679497

GG: b+(... <<< 9) = 2366aa8d

GG (c, d, a, b, x[7], S23, 0x676f02d9)

GG: a=ff445636, b=2366aa8d, c=543d7c4f, d=84897614, xk=00000000, s=14, ti=676f02d9

G(5057a899, a9c5a0f1, e8909e06) = 50342a4f

a+G(b,c,d) + xk+ti= b6e7835e

GG: b+(... <<< 14) = 043e5846

GG (b, c, d, a, x[12], S23, 0x8d2a4c8a)

GG: a=84897614, b=043e5846, c=543d7c4f, d=543d7c4f, xk=00000000, s=20, ti=8d2a4c8a

G(043e5846, 236aa8d, 543d7c4f) = 277edac6

a+G(b,c,d) + xk+ti= 39329d64

GG: b+(... <<< 20) = da81eb6f

Round 3:

Lakukan operasi [abcd k s i]

a = b + ((a+H(b,c,d) + X[k] + T[i]) <<< s)

Lakukan 16 operasi berikut ini setiap roundnya:

[ABCD 5 4 33][DABC 8 11 34 ] [CDAB 11 16 35 ][BCDA 14 23 36 ]

[ABCD 1 4 37][DABC 4 11 38 ] [CDAB 7 16 39 ][BCDA 10 23 40 ]

[ABCD 13 4 41][DABC 0 11 42 ][CDAB 3 16 43 ][BCDA 6 23 44 ]

[ABCD 9 4 45][DABC 12 11 46 ][CDAB 15 16 47 ][BCDA 2 23 48 ]

Lakukan a = b +((a + H(b,c,d) + X[k] + T[i]) <<< s). (Catatan: <<< Berarti rotate left)

Fungsi yang digunakan pada masukan H(b, c, d) = b^c^d

HH (a, b, c, d, x[5], S31, 0xfffa3942)

HH: a=543d7c4f, b=da81eb6f, c=043e5846, d=2366aa8d, xk=00000000, s=4, ti=fffa3942

H(5057a899, a9c5a0f1, e8909e06) = fdd919a4

a+H(b,c,d) + xk+ti= 5210cf35

HH: b+(... <<< 4) = fb8edec4

HH (d, a, b, c, x[8], S32, 0x8771f681)

HH: a=2366aa8d, b=fb8edec4, c=da81eb6f, d=043e5846, xk=00000000, s=11, ti=8771f681

H(fb8edec4, da81eb6f, 043e5846) = 25316ded

a+H(b,c,d) + xk+ti= d00a0efb

HH: b+(... <<< 11) = 4c06bd44

HH (c, d, a, b, x[11], S33, 0x6d9d6122)

HH: a=043e5846, b=4c06bd44, c=fb8edec4, d=da81eb6f, xk=00000000, s=16, ti=6d9d6122

H(4c06bd44, fb8edec4, da81eb6f) = 6d0988ef

a+H(b,c,d) + xk+ti= dee54257

HH: b+(... <<< 16) = 8e5e9c29

HH (b, c, d, a, x[14], S34, 0xfde5380c)

HH: a=da81eb6f, b=8e5e9c29, c=4c06bd44, d=fb8edec4, xk=00000058, s=23, ti=fde5380c

H(8e5e9c29, 4c06bd44, fb8edec4) = 39d6ffa9

a+H(b,c,d) + xk+ti= 123e237c

HH: b+(... <<< 23) = 4c67bb3a

HH (a, b, c, d, x[1], S31, 0xa4beea44)

HH: a=fb8edec4, b=4c67bb3a, c=8e5e9c29, d=4c06bd44, xk=52474f54, s=4, ti=a4beea44

H(4c67bb3a, 8e5e9c29, 4c06bd44) = 8e3f9a57

a+H(b,c,d) + xk+ti= 80d4b2b3

HH: b+(... <<< 4) = 59b2e672

HH (d, a, b, c, x[4], S32, 0x4bdecfa9)

HH: a=4c06bd44, b=59b2e672, c=4c67bb3a, d=8e5e9c29, xk=00000000, s=11, ti=4bdecfa9

H(59b2e672, 4c67bb3a, 8e5e9c29) = 9b8bc161

a+H(b,c,d) + xk+ti= 33714e4e

HH: b+(... <<< 11) = e425580d

HH (c, d, a, b, x[7], S33, 0xf6bb4b60)

HH: a=8e5e9c29, b=e425580d, c=59b2e672, d=4c67bb3a, xk=00000000, s=16, ti=f6bb4b60

H(e425580d, 59b2e672, 4c67bb3a) = f1f00545

a+H(b,c,d) + xk+ti= 7709ecce

HH: b+(... <<< 16) = d0f3cf16

HH (b, c, d, a, x[10], S34, 0xbebfbc70)

HH: a=4c67bb3a, b=d0f3cf16, c=e425580d, d=59b2e672, xk=00000000, s=23, ti=bebfbc70

H(d0f3cf16, e425580d, 59b2e672) = 6d647169

a+H(b,c,d) + xk+ti= 788be913

HH: b+(... <<< 23) = 5ab0150a

HH (a, b, c, d, x[13], S31, 0x289b7ec6)

HH: a=59b2e672, b=5ab0150a, c=d0f3cf16, d=e425580d, xk=00000000, s=4, ti=289b7ec6

H(5ab0150a, d0f3cf16, e425580d) = 6e668211

a+H(b,c,d) + xk+ti= f0b4e749

HH: b+(... <<< 4) = 65fe89a9

HH (d, a, b, c, x[0], S32, 0xeaa127fa)

HH: a=e425580d, b=65fe89a9, c=5ab0150a, d=d0f3cf16, xk=5049524b, s=11, ti=eaa127fa

H(65fe89a9, 5ab0150a, d0f3cf16) = efbd53b5

a+H(b,c,d) + xk+ti= 0ecd2607

HH: b+(... <<< 11) = cf2ec21f

HH (c, d, b, a, x[3], S33, 0xd4ef3085)

HH: a=d0f3cf16, b=cf2ec21f, c=65fe89a9, d=5ab0150a, xk=00000000, s=16, ti=d4ef3085

H(cf2ec21f, 65fe89a9, 5ab0150a) = f0605ebc

a+H(b,c,d) + xk+ti= 96435e57

HH: b+(... <<< 14) = 2d865862

HH (b, c, d, a, x[6], S34, 0x4881d06)

HH: a=5ab015a, b=2d865862, c=cf2ec21f, d=65fe89a9, xk=00000000, s=16, ti=d4ef3085

H(2d865862, cf2ec21f, 65fe89a9) = 875613d4

a+H(b,c,d) + xk+ti= 68e45e3

HH: b+(... <<< 14) = 1f799f84

HH (a, b, c, d, x[9], S31, 0xd9d4d039)

HH: a=65fe89a9, b=1f799f84, c=2d865862, d=cf2ec21f, xk=00000000, s=4, ti=d9d4d039

H(1f799f84, 2d865862, cf2ec21f) = fdd105f9

a+H(b,c,d) + xk+ti= 3da45fdb

HH: b+(... <<< 4) = f9bf9d37

HH (d, a, b, c, x[12], S32, 0xe6db99e5)

HH: a=cf2ec21f, b=f9bf9d37, c=1f799f84, d=2d865862, xk=00000000, s=11, ti=e6db99e5

H(f9bf9d37, 1f799f84, 2d865862) = cb405ad1

a+H(b,c,d) + xk+ti= 814ab6d5

HH: b+(... <<< 14) = 4f764941

HH (c, d, a, b, x[15], S33, 0x1fa27cf8)

HH: a=2d865862, b=4f764941, c=f9bf9d37, d=1f799f84, xk=00000000, s=16, ti=1fa27cf8

H(4f764941, f9bf9d37, 1f799f84) = a9b04bf2

a+H(b,c,d) + xk+ti= f6d9214c

HH: b+(... <<< 14) = 70c3401a

HH (b, c, d, a, x[2], S34, 0xc4ac5665)

HH: a=1f799f84, b=70c3401a, c=4f764941, d=f9bf9d37, xk=80494641, s=23, ti=c4ac5665

H(70c3401a, 4f764941, f9bf9d37) = c60a946c

a+H(b,c,d) + xk+ti= 2a79d096

HH: b+(... <<< 14) = bbd87d02

Round 4:

Lakukan operasi [abcd k s i]

a = b + ((a+I(b,c,d) + X[k] + T[i]) <<< s)

Lakukan 16 operasi berikut ini setiap roundnya:

[ABCD 1 5 17][DABC 6 9 18 ][CDAB 11 14 19 ][BCDA 0 20 20 ]

[ABCD 5 5 21][DABC 10 9 22 ][CDAB 15 14 23 ][BCDA 4 20 24 ]

[ABCD 9 5 25][DABC 14 9 26 ][CDAB 3 14 27 ][BCDA 8 20 28 ]

[ABCD 13 5 29][DABC 2 9 30 ][CDAB 17 14 31 ][BCDA 12 20 32 ]

Lakukan a = b +((a + I(b,c,d) + X[k] + T[i]) <<< s). (Catatan: <<< Berarti rotate left)

Fungsi yang digunakan pada masukan I(b, c, d) = C xor (b or (not d))

II (a, b, c, d, x[0], S41, 0xf4292244)

II: a=f9bf9d37, b=bbd87d02, c=70c3401a, d=4f764941, xk=5049524b, s=6, ti=f4292244

I(bbd87d02, 70c3401a, 4f764941) = cb1abfa4

a+I(b,c,d) + xk+ti= 094cd16a

II: b+(... <<< 6) = 0f0cd784

II (a, b, c, d, x[7], S42, 0x432aff97)

II: a=4f764941, b=0f0cd784, c=bbd87d02, d=70c3401a, xk=00000000, s=10, ti=432aff97

I(0f0cd784, bbd87d02, 70c3401a) = 34e482e7

a+I(b,c,d) + xk+ti= c785cbbf

II: b+(... <<< 10) = 263bd6a2

II (a, b, c, d, x[14], S43, 0xab9423a7)

II: a=70c3401a, b=263bd6a2, c=0f0cd784, d=bbd87d02, xk=00000058, s=15, ti=ab9423a7

I(263bd6a2, 0f0cd784, bbd87d02) = 6933017b

a+I(b,c,d) + xk+ti= 858a6594

II: b+(... <<< 15) = 59061967

II (a, b, c, d, x[5], S44, 0xfc93a039)

II: a=bbd87d02, b=59061967, c=263bd6a2, d=0fcd784, xk=00000000, s=21, ti=fc93a039

I(59061967, 263bd6a2, 0fcd784) = dfccefdd

a+I(b,c,d) + xk+ti= 98390d18

II: b+(... <<< 21) = fc192088

II (a, b, c, d, x[12], S41, 0x655b59c3)

II: a=0f0cd784, b=fc192088, c=59061967, d=263bd6a2, xk=00000000, s=6, ti=655b59c3

I(fc192088, 59061967, 263bd6a2) = a4db30ba

a+I(b,c,d) + xk+ti= 19436201

II: b+(... <<< 6) = 4cf1a0ce

II (a, b, c, d, x[3], S42, 0x8f0ccc92)

II: a=263bd6a2, b=4cf1a0ce, c=fc192088, d=59061967, xk=00000000, s=10, ti=8f0ccc92

I(4cf1a0ce, fc192088, 59061967) = 12e0c656

a+I(b,c,d) + xk+ti= c829698a

II: b+(... <<< 10) = f297cbee

II (a, b, c, d, x[10], S43, 0xffeff47d)

II: a=59061967, b=f297cbee, c=4cf1a0ce, d=fc192088, xk=00000000, s=15, ti=ffeff47d

I(f297cbee, 4cf1a0ce, fc192088) = bf067f31

a+I(b,c,d) + xk+ti= 17fc8d15

II: b+(... <<< 15) = 392257ec

II (a, b, c, d, x[1], S44, 0x85845dd1)

II: a=fc192088, b=392257ec, c=f297cbee, d=4cf1a0ce, xk=52474f54, s=21, ti=8584dd1

I(392257ec, f297cbee, 4cf1a0ce) = 49b99413

a+I(b,c,d) + xk+ti= 1d9e61c0

II: b+(... <<< 21) = 71260bb8

II (a, b, c, d, x[8], S41, 0x6fa87e4f)

II: a=4cf1a0ce, b=71260bb8, c=392257ec, d=f297cbee, xk=00000000, s=6, ti=fe2ce6e0

I(71260bb8, 392257ec, f297cbee) = 444c6855

a+I(b,c,d) + xk+ti= 00e68772

II: b+(... <<< 6) = aac7e838

II (a, b, c, d, x[15], S42, 0xfe2ce6e0)

II: a=f297cbee, b=aac7e838, c=7126bb8, d=392257ec, xk=00000000, s=10, ti=fe2ce6e0

I(aac7e838, 7126bb8, 392257ec) = 9ff9e383

a+I(b,c,d) + xk+ti= 90be9651

II: b+(... <<< 10) = a5212e7a

II (a, b, c, d, x[6], S43, 0xa3014314)

II: a=392257ec, b=a5212e7a, c=aac7e838, d=71260bb8, xk=00000000, s=15, ti=a3014314

I(a5212e7a, aac7e838, 71260bb8) = 053e1647

a+I(b,c,d) + xk+ti= e161b147

II: b+(... <<< 15) = 7dc51f2a

II (a, b, c, d, x[13], S44, 0x4e0811a1)

II: a=71260bb8, b=7dc51f2a, c=a5212e7a, d=aac7e838, xk=00000000, s=21, ti=4e0811a1

I(7dc51f2a, a5212e7a, aac7e838) = d8dc3195

a+I(b,c,d) + xk+ti= 980a4eee

II: b+(... <<< 21) = 5b982073

II (a, b, c, d, x[4], S41, 0xf7537e82)

II: a=aac7e838, b=5b982073, c=7dc51f2a, d=a5212e7a, xk=00000000, s=6, ti=f7537e82

I(5b982073, 7dc51f2a, a5212e7a) = 261beedd

a+I(b,c,d) + xk+ti= c8375597

II: b+(... <<< 6) = 696d8665

II (a, b, c, d, x[11], S42, 0xbd3af235)

II: a=a5212e7a, b=696d8665, c=5b982073, d=7dc51f2a, xk=00000000, s=10, ti=bd3af235

I(696d8665, 5b982073, 7dc51f2a) = b0e7c686

a+I(b,c,d) + xk+ti= 1343e735

II: b+(... <<< 10) = 790a5ab2

II (a, b, c, d, x[2], S43, 0x2ad7d2bb)

II: a=7dc51f2a, b=790a5ab2, c=696d8665, d=5b982073, xk=80494641, s=15, ti=2ad7d2bb

I(bbd87d02, 70c3401a, 4f764941) = 940259db

a+I(b,c,d) + xk+ti= bce89201

II: b+(... <<< 15) = c20b3926

II (a, b, c, d, x[9], S44, 0x eb86d391)

II: a=5b982073, b= c20b3926, c=790a5ab2, d=696d8665, xk=00000000, s=21, ti=eb86d391

I(bbd87d02, 70c3401a, 4f764941) = af91230c

a+I(b,c,d) + xk+ti= f6b01710

II: b+(... <<< 6) = a42a0f28

Lakukan operasi penambahan berikut. (Bahwa tambahkan dari tiap empat register dari nilai yang dimiliki sebelumnya sebelum blok dimulai.)

a = (a + AA) & 0x0fffffff;

b = (b + BB) & 0x0fffffff;

c = (c + CC) & 0x0fffffff;

d = (d + DD) & 0x0fffffff;

5057a899 + 67452301

a (0x067452301) = 67452301

a = 67452301

AA = 69d8665 merupakan hasil dari II (a, b, c, d, x[4], S41, 0xf7537e82)

= D0B2A966 & FFFFFFFF

= D0B2A966

BB merupakan hasil dari II (b, c, d, a, x[9], S44, 0xeb86d391)

b (0x0efcdab89) = efcdab89

BB = a42a0f28

b = (b + BB) & 0x0ffffffff;

= (efcdab89 + a42a0f28) & 0x0ffffffff

= 93F7BAB1

CC merupakan hasil dari II (c, d, a, b, x[2], S43, 0x2ad7d2bb)

c (0x098badcfe) = 98badcfe

CC = c20b3926

c = (c + CC) & 0x0ffffffff;

= (98badcfe + c20b3926) & 0x0ffffffff

= 5AC61624

DD merupakan hasil dari II (d, a, b, c, x[11], S42, 0xbd3af235)

D (0x010325476) = 10325476

DD = 790a5ab2

D = (d + DD) & 0x0ffffffff;

= (10325476 + 790a5ab2) & 0x0ffffffff

= 893CAF28

Round selesai,

a = -793597594, konversi nilai 32-bit ke 8-char hex string d0b2a966, dikembalikan ke nilai hex LSB (Least Significant Bit) menjadi 66a9b2d0

b = -1812481359, konversi nilai 32-bit ke 8-char hex string 93f7bab1, dikembalikan ke nilai hex LSB (Least Significant Bit) menjadi b1baff79300000000

c = 1522931236, konversi nilai 32-bit ke 8-char hex string5ac61624, dikembalikan ke nilai hex LSB (Least Significant Bit) menjadi 2416c65a00000000

d = -1992511704, konversi nilai 32-bit ke 8-char hex string 893caf28, dikembalikan ke nilai hex LSB (Least Significant Bit) menjadi 28af3c8900000000

Hasil hash terakhir: lsb\_hex(a) + lsb\_hex(b) + lsb\_hex(c) + lsb\_hex(d) =

66a9b2d0 b1baf793 2416c65a 28af3c89

d0 b2 a8 66 = 66 a9 b2 d0

93 f7 ba b1 = b1 ba f7 93

5a c6 16 24 = 24 16 c6 5a

89 3c af 28 = 28 af 3c 89

# LAMPIRAN 2

Contoh penyelesaian kasus *input* suara menggunakan perhitungan digitasi suara dapat dilihat pada penjelasan di bawah ini:

Misalkan diberikan data sebagai berikut, tiap barisan merupakan 1 sampel suara dalam sebuah periode menggunakan besar sampel 3-bit.

|  |  |
| --- | --- |
| Waktu | Amplitudo |
| 0.001 | 100 |
| 0.002 | 101 |
| 0.003 | 110 |
| 0.004 | 111 |
| 0.005 | 110 |
| 0.006 | 101 |
| 0.007 | 100 |
| 0.008 | 011 |
| 0.009 | 010 |
| 0.010 | 001 |
| 0.011 | 000 |
| 0.012 | 001 |
| 0.013 | 010 |
| 0.014 | 011 |
| 0.015 | 100 |

**Tabel 3**. Contoh Sampling untuk perhitungan

Data amplitudo disimpan dalam bentuk biner. Dari besar nilai yang disimpan, amplitudo dari suara yang di-digitasikan dapat di buat ulang menggunakan perbedaan voltase analog oleh DAC, namun karena bentuk ini sudah dalam digital maka data ini akan digunakan untuk merepresentasikan data input suara.

Setelah mendapatkan data biner dari satu periode makan dapat dihitung besar file suara tersebut.

Pertama dalam data tersebut kita dapat mengetahui frekuensi sampel menggunakan rumus frekuensi.

Kemudian proses selanjutnya yaitu perhitungan Bitrate, yang merupakan banyaknya bit yang dipakai perdetik.

Ketika Bitrate sudah diketahui maka tahap selanjutnya cukup menghitung berapa Panjang rekaman yang dinotasikan dengan waktu misalkan 5 detik.

Maka diketahui bahwa besar file tersebut dengan 5 detik rekaman akan menghasilkan file sebesar 1000 bits atau 125 bytes. File tersebut akan tersimpan dalam bentuk Waveform.

# LAMPIRAN 3

**Mel Frequency Cepstrum Cepstral Coefficient (MFCC)**

**Diketahui:**

Rekaman Suara A dengan format .wav dimana data sinyal X dengan suara t = 1 detik.

**Dicari:** 5 fitur MFCC

**Jawab:**

**Frame Blocking**

Melalui proses frame blocking dipilih frame ke-1 yang berisikan N=5 samepl sebagai berikut:

**Windowing**

Hitung :

Hasil perhitungan windowing

**DFT**

Hitung nilai DFT dan Magnitude untuk k = 0,1,2,3,4

44.42

**Mel-Frequency Wrapping**

Jumlah filter yang diinginkan

Konversi dan ke dalam domain frekuensi mel

Frekuensi sampling

Menghitung titik batas filter dengan

Dengan adalah dan adalah

Menghitung triangular filter bank

Dan koedisien spektrum mel

0

**DCT**

J=5

Untuk

**Contoh 1 iterasi EM Gaussian Mixture Model**

Misalkan menjadi datapoint, dianggap masing-masing telah dibuat dari salah satu kedua gaussian.

Standar deviasi

apriori gaussian

Mean data dan

Misalkan j adalah index gaussian, i adalah index data point, dan k adalah index EM.

Maka dapat dihitung pada iterasi ke-1

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1 | 2 | 3 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sehingga

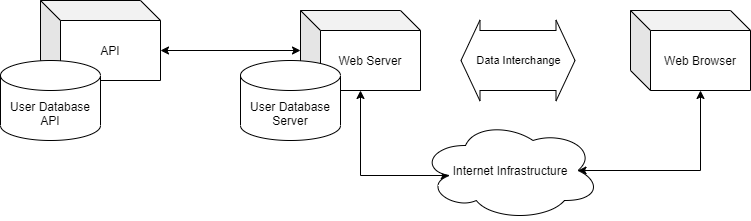
dan

# LAMPIRAN 4

**Tabel 4.** Black Box testing

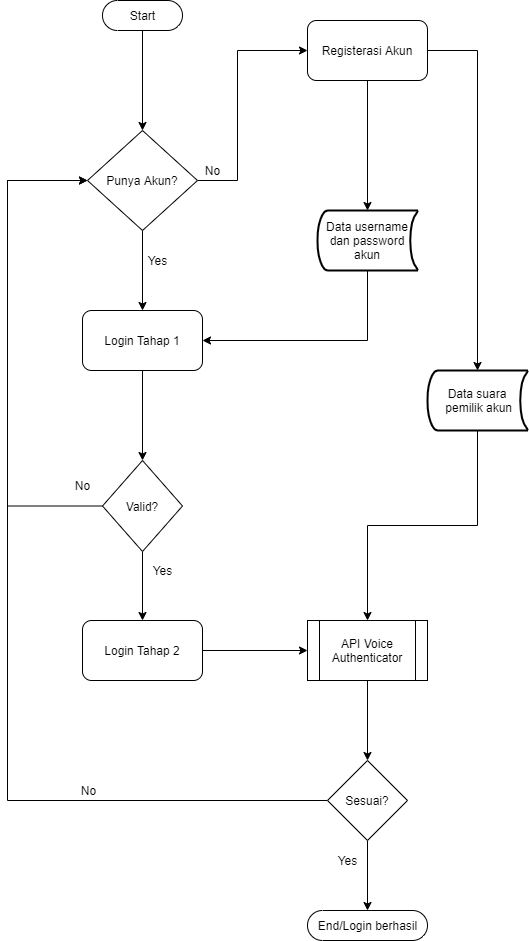
|  |  |  |  |
| --- | --- | --- | --- |
| No | Skenario | Data | Hasil Ekspektasi |
| 1 | User memasukan username dan passwordnya | Username, password | Rancangan sistem login berhasil melakukan verifikasi user ada dalam database atau tidak |
| 2 | User tidak dapat mengklik login sampai form username dan password terisi |  | Rancangan sistem login berhasil mencegah user masuk ke tahap selanjutnya |
| 3 | User asli menginput data suaranya ke website login. | Suara, UserID | Rancangan otentikasi suara berhasil mengirim Kembali hasil otentikasi. |
| 4 | Verifikasi waktu yang diperlukan untuk otentikasi suara memberi validasi |  | Rancangan otentikasi suara membutuhkan maksimal 10 detik untuk melakukan verifikasi |
| 3 | Melakukan panggilan API tanpa menggunakan website. | Suara, UserID | API dapat mendeteksi bahwa panggilan merupakan panggilan palsu. Sehingga API akan mengabaikan panggilan tersebut |
| 4 | Melakukan panggilan API menggunakan panggilan yang pernah dilakukan oleh user asli. | Suara, UserID | API dapat mendeteksi bahwa panggilan merupakan panggilan palsu. Sehingga API akan mengabaikan panggilan tersebut |
| 5 | Melakukan panggilan API menggunakan injeksi serangkai kode | kode | API dapat mendeteksi panggilan tersebut berupa kodingan dan merupakan panggilan invalid. |

# LAMPIRAN 5



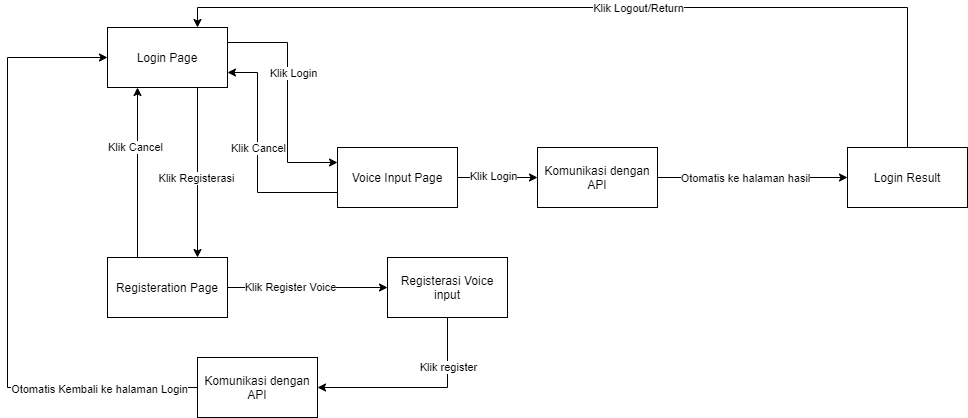
**Gambar 7.** Skema Rancangan Sistem *Login*

# LAMPIRAN 6



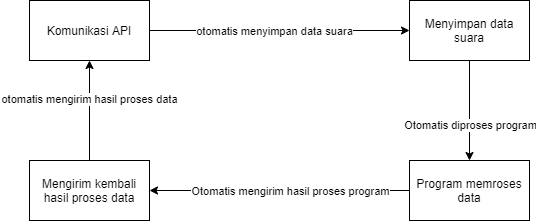
**Gambar 8.** Diagram Alur Sistem *Login*

# LAMPIRAN 7



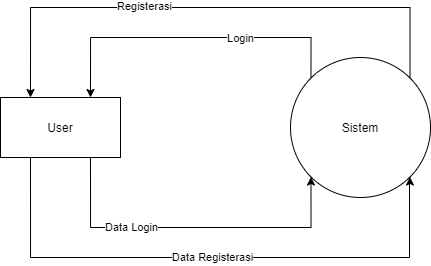
**Gambar 9.** Diagram STD Sistem *Login* dan Registrasi.

# LAMPIRAN 8



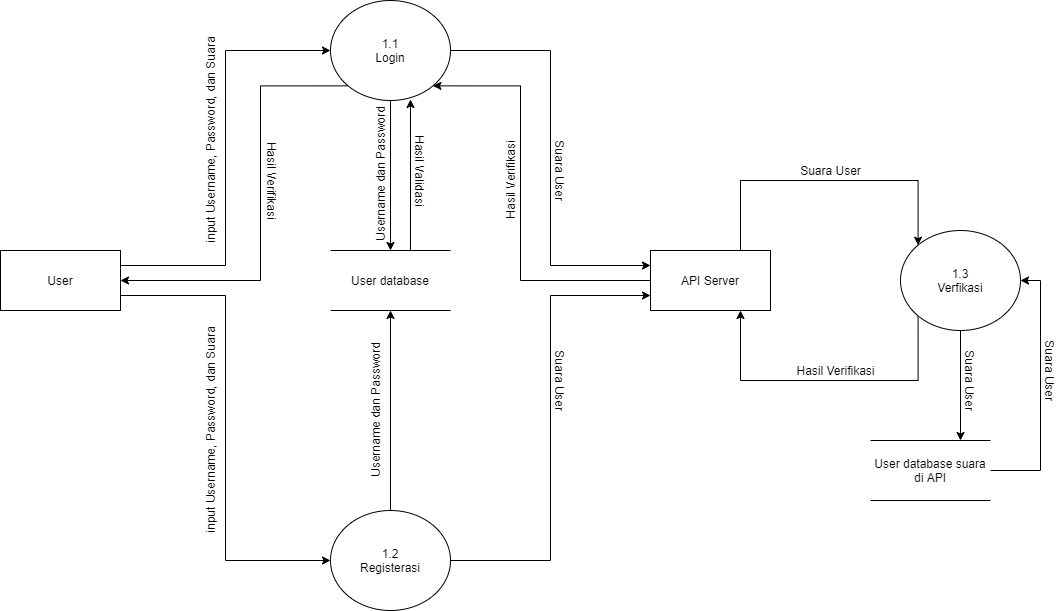
**Gambar 10.** Diagram STD API Sistem *Login*.

# LAMPIRAN 9



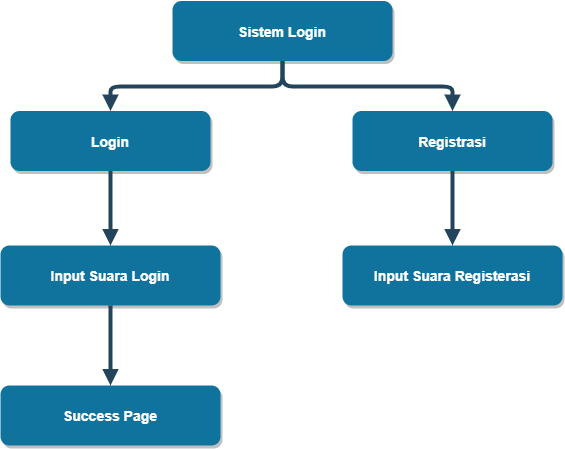
**Gambar 11.** Context Diagram Sistem *Login*.

# LAMPIRAN 10



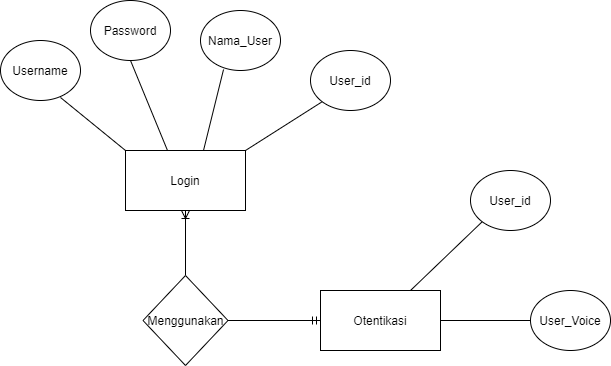
**Gambar 12.** DFD Sistem *Login.*

# LAMPIRAN 11



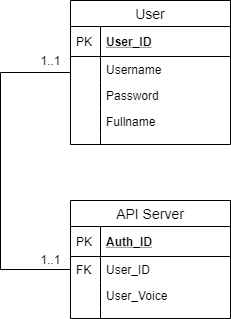
**Gambar 13.** Diagram Hierarki Sistem *Login*.

# LAMPIRAN 12



**Gambar 14**. Hubungan Antar Entitas Sistem *Login.*

# LAMPIRAN 13



**Gambar 15**. Hubungan Antar Tabel Sistem *Login*

# LAMPIRAN 14

**Tabel 4**. Tabel User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Kolom | Data type | Size | Keterangan |
| 1 | User\_ID | int | Max | Kode ID pengguna |
| 2 | Username | Varchar | 25 | Data Username Pengguna |
| 3 | Password | Varchar | 25 | Data Password Pengguna |
| 4 | FullName | Varchar | 50 | Nama Panjang pengguna |

*Primary Key*: User\_ID

*Foreign Key*: User\_ID

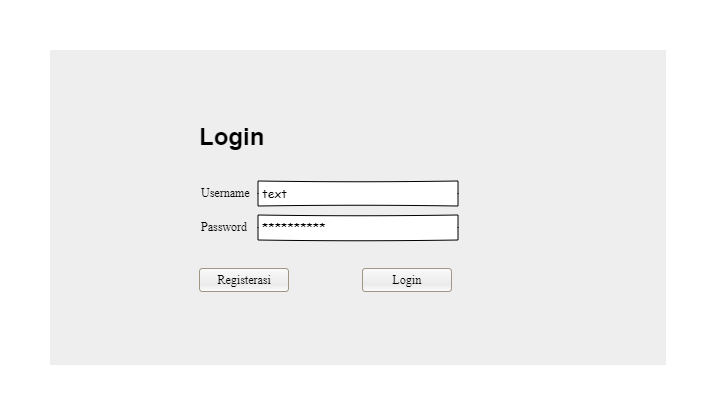
**Tabel 5.** Tabel API Server

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Kolom | Data type | Size | Keterangan |
| 1 | Auth\_ID | Varchar | 50 | Kode otentikasi ID pengguna |
| 2 | User\_ID | int | Max | Kode ID Pengguna |
| 3 | User\_Voice | Varchar | Max | Lokasi Data Suara Pengguna |

*Primary Key*: Auth\_ID

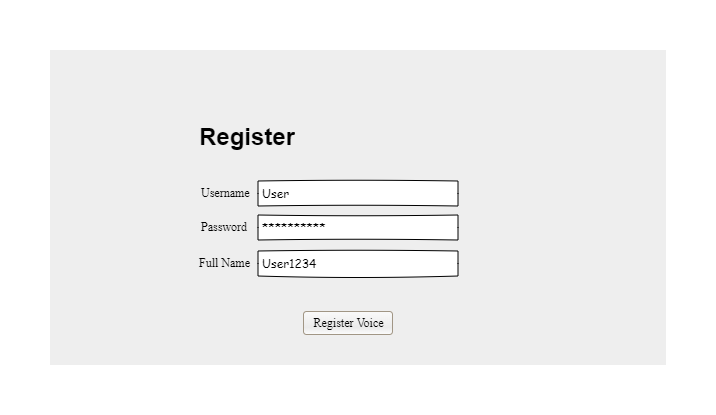
*Foreign Key*: User\_ID

# LAMPIRAN 15



**Gambar 16.** Rancangan Halaman *Login* untuk Sistem *Login*

# LAMPIRAN 16



**Gambar 17.** Rancangan Halaman Registrasi untuk Sistem *Login*

# LAMPIRAN 17



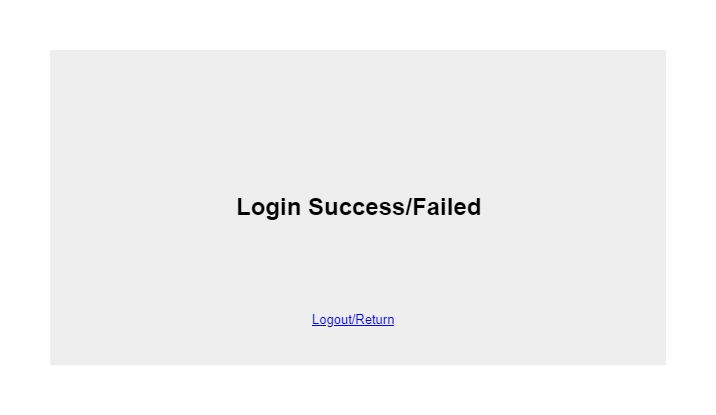
**Gambar 18.** Rancangan Halaman *Input* Suara *login* untuk Sistem *Login*

# LAMPIRAN 18



**Gambar 19.** Rancangan Halaman *input* Suara Registrasi untuk Sistem *Login*

# LAMPIRAN 19



**Gambar 20.** Rancangan Halaman Hasil *Login* untuk Sistem *Login*

1. Anil K. Jain, Karthik Nandakumar, Abhishek Nagar, Biometric template security, https://dl.acm.org/doi/abs/10.1155/2008/579416, Januari 2008. [↑](#footnote-ref-1)
2. Debnath Bhattacharyya et al., “Biometric Authentication: A Review”, International Journal of u- and e- Service, Science and Technology Vol. 2, No. 3, September 2009. [↑](#footnote-ref-2)
3. Jose Luis Alba-Castro et al., “Security issues of Internet-based biometric authentication systems: risks of Man-in-the-Middle and BioPhishing on the example of BioWebAuth”, Proceedings of SPIE - The International Society for Optical Engineering, Februari 2008. [↑](#footnote-ref-3)
4. Fisher, Sharon, “OS/2 EE to Get 3270 Interface Early”, San Francisco: Info World, Vol 10, Nomor 34, (Agustus 1988), h. 38 [↑](#footnote-ref-4)
5. Sarah Coble, API Attacks Increase During Lockdown, https://www.infosecurity-magazine.com/news/api-attacks-increase-during/, 15 Mei 2020 [↑](#footnote-ref-5)
6. Ding Wang dan Ping Wang, “Two Birds with One Stone: Two-Factor Authentication with Security Beyond Conventional Bound”, IEEE Transactions on Dependable and Secure Computing, Vol. 5, Nomor 4, (1 Juli 2018), h. 708. [↑](#footnote-ref-6)
7. Jose Luis Alba-Castro et al., “Security issues of Internet-based biometric authentication systems: risks of Man-in-the-Middle and BioPhishing on the example of BioWebAuth”, Proceedings of SPIE - The International Society for Optical Engineering, Februari 2008. [↑](#footnote-ref-7)
8. Battista Biggio et al., “Security Evaluation of Biometric Authentication Systems”, IET Biometrics, Vol 1, (Maret 2012), h.12-24. [↑](#footnote-ref-8)
9. Margaret Rouse, MD5 Definition, https://searchsecurity.techtarget.com/definition/MD5, April 2017. [↑](#footnote-ref-9)
10. Mary Cindy Ah Kioon, Zhao Shun Wang, Shubra Deb Das, “Security Analysis of MD5 Algorithm in Password Storage”, Applied Mechanics and Materials, Vol 347, (Agustus, 2013), h. 2706. [↑](#footnote-ref-10)
11. R. Rivest, “The MD5 Message-Digest Algorithm”, RFC1321, April 1992. [↑](#footnote-ref-11)
12. Anonymous, The MD5 cryptographic hash function, https://www.iusmentis.com/technology/hashfunctions/md5/, Oktober 2005 [↑](#footnote-ref-12)
13. Sri Waluyanti, Buku Direktorat PSMK Untuk Tehnik Audio Video, (Jakarta: Direktorat Pembinaan SMK, 2008), h.1. [↑](#footnote-ref-13)
14. Ibid., h.5. [↑](#footnote-ref-14)
15. Bill Poser, Audio Data, http://billposer.org/Linguistics/Computation/LectureNotes/AudioData.html, 9 September 2020. [↑](#footnote-ref-15)
16. Cliff Truesdell, Mastering Digital Audio Production: The Professional Music Workflow with Mac OS X, (Vancouver: Wiley Publishing Inc, 2007), h.516. [↑](#footnote-ref-16)
17. Abdel-rahman Mohamed, Deep Neural Network Model for ASR,Toronto: Department of Computer Science University Toronto (Thesis of Doctor Philosophy Graduate), 2014, h.44. [↑](#footnote-ref-17)
18. Ali Mustofa, Sistem Pengenalan Penutur dengan Metode Mel-Frequency Wrapping, Jurnal Teknik Elektro, Vol.7, No.2, (September, 2007), h.90. [↑](#footnote-ref-18)
19. Marwa A.Nasr, et.al., Speaker identification based on ormalized pitch frequency and Mel Frequency Cepstral Coefficient, International Journal of Speech Technology , Vol.21, Issue 4,(Desember, 2018), h.941-951 [↑](#footnote-ref-19)
20. Ali Mustofa, Op.Cit., h.90 [↑](#footnote-ref-20)
21. ibid [↑](#footnote-ref-21)
22. ibid [↑](#footnote-ref-22)
23. ibid [↑](#footnote-ref-23)
24. Agus Buono, Wisnu Jatmiko, dan Benyamin Kusumoputro, Perluasan Metode MFCC 1D ke 2D sebagai Ekstraksi Ciri pada Sistem Identifikasi Pembicara Menggn Hidden Markov Model (HMM), Makara, Sains, Vol. 13, Nomor 1, (April, 2009), h.87. [↑](#footnote-ref-24)
25. Bella, M. Prof. Dr. Ir. Dyah Erny Herwindiati and S. M. Janson Hendryli, Aplikasi Mobile Electronic Wallet Berbasis Android Dengan Fitur Voice Authentication, 2018. [↑](#footnote-ref-25)
26. Martin A Moskowitz, A Course in Complex Analysis in One Variable, (Singapore: World Scientific Publishing Co., 2002), h.7. [↑](#footnote-ref-26)
27. Agus Buono, Wisnu Jatmiko, dan Benyamin Kusumoputro, Op.Cit. [↑](#footnote-ref-27)
28. Ali Mustofa, Op.Cit. [↑](#footnote-ref-28)
29. ibid [↑](#footnote-ref-29)
30. Agus Buono, Wisnu Jatmiko, dan Benyamin Kusumoputro, Op.Cit.h.89. [↑](#footnote-ref-30)
31. Todor Dimitrov Ganchev, Speaker Recognition, Patras: Department of Electrical and Computer Engineering University of Patras (Unpublished Doctoral Dissertation), 2005, h.68. [↑](#footnote-ref-31)
32. Ibid. [↑](#footnote-ref-32)
33. Ibid., h.68. [↑](#footnote-ref-33)
34. Ibid. [↑](#footnote-ref-34)
35. Ibid., h.69. [↑](#footnote-ref-35)
36. Ibid., h.70. [↑](#footnote-ref-36)
37. Ibid., h.71 [↑](#footnote-ref-37)
38. Ibid., h.72 [↑](#footnote-ref-38)
39. Gu, Juan, Jun Chen, Qiming Zhou, Hongwei Zhang. *Gaussian Mixture Model of Texture for Extracting Residential Area from High-Resolution Remotely Sensed Imagery*, (ISPRS Workshop on Updating Geo-spatial Databases with Imagery & The 5th ISPRS Workshop on DMGISs), Agustus 2007, h.157-162. [↑](#footnote-ref-39)
40. Palaanen, et al. “Feature representation and discrimination based on Gaussian mixture model probability densities Practices and algorithms”, Pattern Recognition, Vol. 39, No. 7, (Juli 2006), h. 1346-1358 [↑](#footnote-ref-40)
41. Herru Darmadi, Merancang Web API yang Mudah Dieksplorasi, https://binus.ac.id/knowledge/2018/05/merancang-web-api-yang-mudah-dieksplorasi/, 31 Mei 2018. [↑](#footnote-ref-41)
42. Guru99, Black Box Testing, https://www.guru99.com/black-box-testing.html, 21 Oktober 2020. [↑](#footnote-ref-42)
43. Luqman Hakim, Bahasa Pemrograman C# dan EmguCV, (Yogyakarta: DEEPUBLISH, 2018), h. 11. [↑](#footnote-ref-43)
44. Hanry Ham, S.Kom., M.Eng., ASP.Net, https://socs.binus.ac.id/2018/12/20/asp-net/, 18 September 2020. [↑](#footnote-ref-44)
45. Ariata C., Apa Itu JavaScript, https://www.hostinger.co.id/tutorial/apa-itu-javascript/, Januari 2019. [↑](#footnote-ref-45)
46. Admin Nawadwipa, Pengertian Dan Fungsi HTML¸ https://www.nawadwipa.co.id/pengertian-dan-fungsi-html-hypertext-markup-language/, Agustus 2020. [↑](#footnote-ref-46)
47. Anonimous, Mengenal Apa Itu MySQL, Fungsi, dan Cara Kerjanya, https://idcloudhost.com/mengenal-apa-itu-mysql-fungsi-dan-cara-kerjanya-lengkap/, 18 September 2020 [↑](#footnote-ref-47)
48. Roger S.Pressman, Software Engineering, 7th Edition, (New York: McGraw-Hill, 2008), h. 13. [↑](#footnote-ref-48)
49. Putra, Pengertian SDLC adalah: Fungsi, Metode dan Tahapan SDLC, https://salamadian.com/sdlc-system-development-life-cycle/, 23 September 2020 [↑](#footnote-ref-49)
50. Ibid., h.28 [↑](#footnote-ref-50)
51. Anonimus, Gambar Skematik Diagram LRT Bandung Raya, https://kampungdrafter.com/gambar-skematik-diagram-lrt-bandung-raya/#:~:text=Gambar%20digaram%20skematik%20adalah%20gambar,bukan%20dengan%20bentuk%20yang%20sesungguhnya., 25 September 2020 [↑](#footnote-ref-51)
52. Ali Ridho Barakbah, S.Kom., PhD., Tita Karlita, S.Kom., M.Kom., dan Ahmad Syauqi Ahsan, S.Kom., Logika dan Algoritma, (Surabaya: Departemen Teknik Informatika dan Komputer Politeknik Elektronika Negeri Surabaya, 2013), h. 19. [↑](#footnote-ref-52)
53. Nova V. Siringoringo, Pengertian, State Diagram, https://medium.com/state-diagram/pengertian-state-diagram-67d3a4069da3, 3 Februari 2020. [↑](#footnote-ref-53)
54. Ansori, Pengertian DFD (Data Flow Diagram): Fungsi, Simbol, dan Contohnya, https://www.ansoriweb.com/2020/03/pengertian-dfd.html, 25 September 2020. [↑](#footnote-ref-54)
55. Dosen Pendidikan 3, Basis Data Adalah, https://www.dosenpendidikan.co.id/basis-data/, 25 September 2020. [↑](#footnote-ref-55)
56. Nabil Tediamaja, Pengertian, Fungsi, Contoh dan Cara Membuat ERD (Entity Pelationship Diagram), https://bilabil.com/contoh-erd/#Pengertian\_ERD\_Entity\_Pelationship\_Diagram, 25 September 2020. [↑](#footnote-ref-56)
57. Fatkhan Amirul Huda, Relasi Antar Tabel dalam Database, http://fatkhan.web.id/relasi-antar-tabel-dalam-database/#:~:text=Mempunyai%20pengertian%20%E2%80%9CSetiap%20baris%20data,orang%20tua%20begitu%20juga%20sebaliknya., 25 September 2020 [↑](#footnote-ref-57)