DESAIN KOMUNIKASI DAN KEAMANAN DATA ARSITEKTUR APLIKASI MULTIMASJID

Disusun Oleh:

N a m a : Unggul Budi Astowo
NIM : 20523026

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2024

HALAMAN PENGESAHAN DOSEN PEMBIMBING

DESAIN KOMUNIKASI DAN KEAMANAN DATA ARSITEKTUR APLIKASI MULTIMASJID

TUGAS AKHIR

Disusun Oleh:

N a m a : Unggul Budi Astowo

NIM : 20523026

Yogyakarta, 26 Juni 2024

Pembimbing,

(Ari Sujarwo, S.Kom., M.I.T.)

HALAMAN PENGESAHAN DOSEN PENGUJI

DESAIN KOMUNIKASI DAN KEAMANAN DATA ARSITEKTUR APLIKASI MULTIMASJID

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 26 Juli 2024

Tim Penguji

Ari Sujarwo, S.Kom., M.I.T. _____

Anggota 1

Dr. Ahmad Luthfi, S.Kom., M.Kom. _____

Anggota 2

Nur Wijayaning Rahayu, S.Kom., M.Sc. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Unggul Budi Astowo

NIM : 20523026

Tugas akhir dengan judul:

DESAIN KOMUNIKASI DAN KEAMANAN DATA ARSITEKTUR APLIKASI MULTIMASJID

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 26 Juni 2024

Unggul Budi Astowo

HALAMAN PERSEMBAHAN

Alhamdulillahirobbil'alamin, puji syukur saya ucapkan kepada Allah SWT atas berkat dan rahmat-Nya telah memberikan kemudahan dan kelancaran dalam proses pembuatan Laporan Tugas Akhir sehingga dapat terselesaikan dengan baik. Saya persembahkan karya tulis ini kepada kedua orang tua serta keluarga besar yang selalu mendukung serta mendoakan saya selama penjalanan kuliah hingga penyelesaian tugas akhir. Terima kasih kepada pembimbing saya, Bapak Ari Sujarwo, S.Kom., M.I.T. yang telah meluangkan waktu untuk membimbing, mengarahkan, memberi saran, pengetahuan baru, motivasi, serta dukungan. Yang terakhir saya persembahkan pula kepada teman-teman dan segala pihak yang turut membantu, mendukung, dan menemani proses yang telah saya jalani selama berkuliah sebagai mahasiswa Teknik Informatika, Universitas Islam Indonesia. Semoga Allah selalu senantiasa menjunjung kita ke dalam kebaikan, serta seluruh ilmu yang diperoleh menjadi berkah kelak bagi diri sendiri maupun orang lain, dan semoga kita diberikan kelancaran, kemudahan, serta tidak mudah menyerah terhadap kesulitan dalam ujian hidup. Aamiin ya robbal alamin

HALAMAN MOTO

“Selesaikan apa yang dimulai”

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan laporan tugas akhir dengan judul "Desain Komunikasi dan Keamanan Data Arsitektur Aplikasi MultiMasjid" ini dengan baik.

Laporan ini disusun untuk mendokumentasikan proses penelitian dan pengembangan sistem keamanan data yang mengimplementasikan teknologi JSON Web Token untuk otentikasi dan otorisasi, Cross-Origin Resource Sharing (CORS) untuk mengatur kontrol akses API, serta enkripsi database menggunakan algoritma AES untuk melindungi data sensitif. Tujuan dari penelitian ini adalah untuk membangun sebuah sistem keamanan yang kuat dan komprehensif dalam melindungi data pengguna dan jamaah dari akses tidak sah, serta menjaga kerahasiaan dan integritas data.

Dalam proses penyelesaian laporan ini, penulis menghadapi berbagai kendala dan tantangan, terutama dalam memahami konsep-konsep keamanan data serta mengimplementasikannya dalam sistem yang dibangun. Namun, berkat bimbingan dan dukungan dari berbagai pihak, kendala-kendala tersebut dapat diatasi dengan baik.

Penulis mengucapkan terima kasih kepada dosen pembimbing, Bapak Ari Sujarwo, S.Kom., M.I.T., yang telah memberikan bimbingan, arahan, dan masukan yang sangat berharga selama proses penelitian dan penyusunan laporan ini. Ucapan terima kasih juga penulis sampaikan kepada seluruh dosen dan staf program studi Informatika UII yang telah memberikan ilmu dan dukungan selama masa perkuliahan.

Penulis berharap laporan ini dapat memberikan kontribusi positif dalam pengembangan sistem keamanan data dan menjadi referensi yang bermanfaat bagi pembaca maupun penelitian-penelitian selanjutnya. Penulis menyadari bahwa laporan ini masih memiliki kekurangan dan masih terdapat ruang untuk perbaikan.

Yogyakarta, 26 Juni 2024

(Unggul Budi Astowo)

SARI

Keamanan data merupakan aspek penting dalam sebuah sistem untuk melindungi informasi sensitif dari akses tidak sah dan penyalahgunaan. Penelitian ini bertujuan untuk mengimplementasikan mekanisme keamanan data yang kuat dengan memanfaatkan teknologi JSON Web Token (JWT) untuk otentikasi dan otorisasi, Cross-Origin Resource Sharing (CORS) untuk mengatur akses akses API, serta enkripsi database menggunakan algoritma Advanced Encryption Standard (AES) untuk melindungi data sensitif.

Dalam penelitian ini, dibangun sebuah sistem keamanan data yang mengimplementasikan JWT untuk proses autentikasi pengguna dan otorisasi akses ke sumber daya. Implementasi CORS memungkinkan sistem untuk mengontrol akses lintas sumber dengan membatasi permintaan hanya dari domain yang diizinkan. Sementara itu, enkripsi database dengan AES digunakan untuk mengenkripsi data sensitif seperti informasi pengguna dan jamaah sebelum disimpan di database, sehingga menjaga kerahasiaan dan integritas data.

Metodologi yang digunakan dalam penelitian ini meliputi perancangan program sistem keamanan, pengembangan kode dengan menggunakan bahasa pemrograman JavaScript dan framework Node.js, serta pengujian sistem dengan menggunakan metode black box testing. Pengujian dilakukan dengan memanfaatkan aplikasi Postman sebagai klien untuk menguji berbagai skenario penggunaan sistem, seperti otentikasi pengguna, permintaan sumber daya, dan validasi token JWT.

Hasil pengujian menunjukkan bahwa implementasi keamanan data dengan JWT, CORS, dan enkripsi database berhasil dilakukan dengan baik. Setiap aspek keamanan yang diterapkan berfungsi sesuai harapan dan sistem mampu menangani berbagai skenario pengujian dengan hasil yang sesuai. Enkripsi database AES berhasil melindungi data sensitif dari kebocoran dan memastikan kerahasiaan data saat disimpan atau ditransmisikan.

Kata kunci: Keamanan data, JSON Web Token, CORS, Enkripsi database, Advanced Encryption Standard, Otentikasi, Otorisasi.

GLOSARIUM

Autentikasi	Proses verifikasi identitas pengguna atau entitas dalam sistem untuk memastikan bahwa mereka adalah pihak yang berwenang dan sah.
Otorisasi	Proses pemberian hak akses dan izin kepada pengguna atau entitas yang telah diautentikasi untuk mengakses sumber daya atau melakukan tindakan tertentu dalam sistem.
CORS	Mekanisme keamanan yang digunakan untuk membatasi bagaimana sumber daya pada satu asal dapat diakses oleh sumber daya dari asal yang berbeda.
Dekripsi	Proses mengonversi data yang terenkripsi menjadi data yang dapat dibaca atau teks dalam bentuk yang semula.
Enkripsi	Proses mengonversi data atau informasi menjadi bentuk yang tidak dapat dibaca atau teks tersamar untuk melindungi keamanan dan privasi data.
JavaScript	Bahasa pemrograman yang digunakan untuk membuat program menjadi interaktif dan dinamis.
JWT	Standar yang digunakan untuk mengamankan transmisi informasi antara pihak-pihak dengan cara yang aman.
Node.js	Lingkungan *runtime open-source* yang dibangun di atas JavaScript digunakan untuk menjalankan kode JavaScript di sisi server.
NPM	Manajer paket yang digunakan untuk instalasi, berbagi, dan mendistribusikan kode sumber dalam proyek JavaScript.
Triad CIA	Tiga aspek utama dalam keamanan data yaitu *Confidentiality* (kerahasiaan), *Integrity* (integritas), dan *Availability* (ketersediaan) yang menjadi panduan dalam pembuatan kebijakan terkait keamanan data dan informasi.
Waterfall	metode pengembangan perangkat lunak.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Masjid merupakan tempat yang disyariatkan oleh Nabi Muhammad SAW untuk melaksanakan sholat wajib secara berjamaah yang mencerminkan kebersamaan, solidaritas, dan silaturahmi antar sesama umat Islam. Selain itu, masjid juga berfungsi sebagai pusat kegiatan sosial, pendidikan, dan pengembangan wawasan keislaman. Oleh karena itu, masjid perlu dikelola dengan baik agar dapat memberikan manfaat bagi masyarakat di sekitarnya. Hal ini mesti dilaksanakan dengan manajemen masjid yang baik (Qadaruddin et al., 2016).

Saat ini, banyak masjid masih mengandalkan sistem pengelolaan manual, terutama di daerah pedesaan (Elsera et al., 2021). Salah satu permasalahan yang dihadapi adalah sistem pendataan jamaah yang dilakukan secara manual, seperti pencatatan menggunakan buku dan arsip. Metode ini tidak hanya memerlukan waktu yang signifikan, tetapi juga rentan terhadap kesalahan dan ketidakefisienan dalam pengelolaan data.

Dengan kemajuan teknologi informasi, pengelolaan masjid dapat ditingkatkan melalui digitalisasi dan penggunaan sistem informasi yang terkomputerisasi. Aplikasi MultiMasjid hadir sebagai solusi untuk mengatasi permasalahan tersebut. Aplikasi ini dirancang untuk memudahkan pengelolaan data jamaah dan kegiatan masjid, serta meningkatkan efisiensi dan akurasi pengelolaan masjid secara keseluruhan.

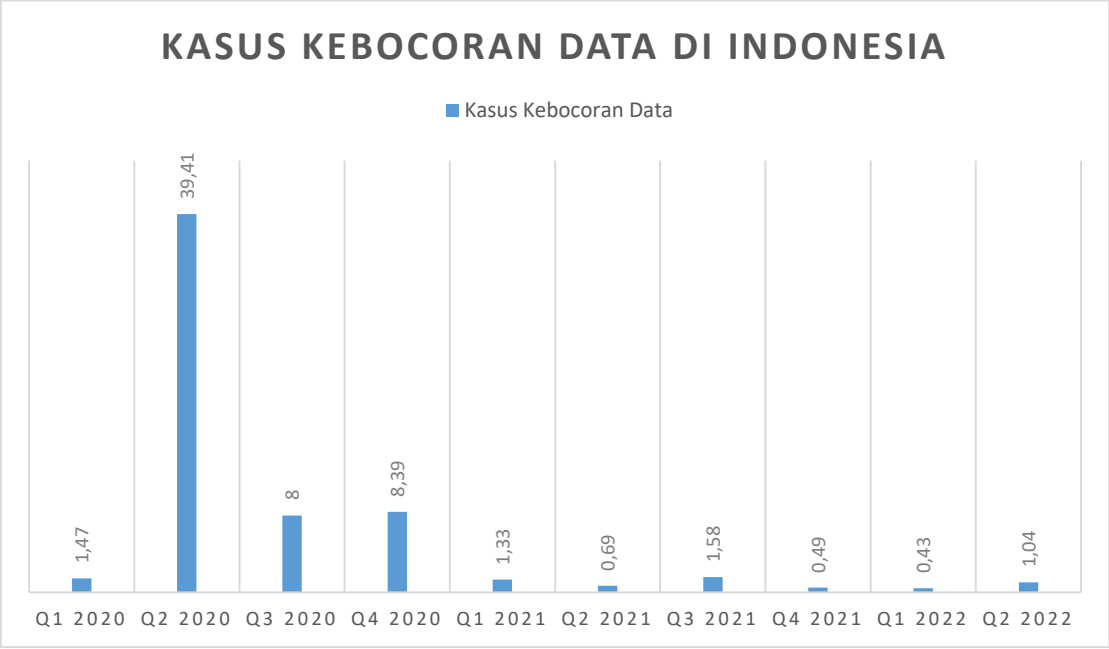
Aplikasi MultiMasjid menawarkan fitur yang digunakan untuk pengelolaan data jamaah dan kegiatan masjid. Fitur pengelolaan data jamaah memungkinkan pengurus masjid untuk mencatat dan mengelompokkan data jamaah berdasarkan nama, alamat, nomor telepon, nomor Kartu Keluarga (KK), dan Nomor Induk Kependudukan (NIK). Setiap elemen data ini memiliki kegunaan dalam mendukung berbagai aktivitas dan pelayanan masjid. Nama dan alamat digunakan pengurus masjid untuk mengelola distribusi zakat dan daging qurban secara efisien dan akurat, memastikan bantuan sampai kepada yang benar-benar membutuhkan. Informasi nomor telepon memfasilitasi komunikasi langsung dengan jamaah, baik untuk mengirimkan pengumuman kegiatan, jadwal sholat, atau undangan acara khusus melalui SMS atau WhatsApp.

Nomor KK dan NIK berperan penting dalam memvalidasi identitas jamaah, memastikan keakuratan data, dan menghindari duplikasi. Data ini juga mendukung pengelolaan administratif yang lebih baik, seperti pencatatan keanggotaan dan pelacakan partisipasi dalam kegiatan masjid. Selain itu, dengan memiliki data yang lengkap dan terstruktur, pengurus

masjid dapat membuat keputusan yang lebih tepat berdasarkan analisis data, seperti menentukan prioritas bantuan sosial dan merencanakan kegiatan yang lebih relevan dengan kebutuhan jamaah. Fitur kegiatan masjid memudahkan pengurus dalam merencanakan dan mengkoordinasikan berbagai kegiatan seperti kajian, pengajian, tadarus, dan bakti sosial, serta menampilkan agenda kegiatan terbaru dan terdekat untuk jamaah.

Namun, di balik manfaat tersebut, ada tantangan besar terkait dengan keamanan data pribadi yang disimpan. Data seperti nama, alamat, nomor telepon, KK, dan NIK adalah informasi sensitif yang jika tidak diamankan dengan baik, dapat disalahgunakan untuk kejahatan atau menyebabkan kerugian bagi jamaah. Kondisi ini diperburuk dengan meningkatnya kasus kebocoran data di Indonesia, yang menyebabkan keresahan di masyarakat mengenai privasi dan keamanan data pribadi mereka.

Meskipun survei khusus tentang kebocoran data di lingkungan masjid atau sektor sosial lainnya mungkin belum banyak dilakukan, laporan mengenai kebocoran data secara umum di Indonesia menunjukkan peningkatan signifikan dalam beberapa tahun terakhir.



Gambar 1.1 Jumlah Akun yang Mengalami Kebocoran Data di Indonesia

Sumber: Katadata.co.id

Berdasarkan gambar grafik di atas, menurut data dari perusahaan Surfshark kasus kebocoran data di Indonesia menunjukan peningkatan dari tahun ke tahun. Lonjakan drastis terjadi pada kuartal II 2020 dengan 39.61 juta dibanding hanya 1.47 juta pada kuartal I 2020

dan 8 juta pada kuartal III 2020. Jumlah insiden kebocoran data juga mengalami lonjakan sebesar 143% pada kuartal II 2022 dengan total kasus mencapai 1,04 juta akun dibandingkan kuartal I 2022 yang hanya berjumlah 430,1 ribu akun.

Lonjakan yang ditunjukkan pada grafik tersebut sejalan dengan beberapa insiden kebocoran data yang terjadi pada Agustus dan September 2022, yang dilakukan oleh seorang peretas dengan nama bjorka. Beberapa di antaranya adalah kebocoran 91 juta data pelanggan Tokopedia, 26 juta data pelanggan IndiHome yang mencakup NIK, email, nomor handphone, kata kunci pencarian, domain, platform, dan URL, serta 1,3 miliar data registrasi kartu SIM dari Kementerian Komunikasi dan Informasi, Terdiri dari NIK, KK, nama lengkap, tempat dan tanggal lahir, kelamin, usia, domisil dan TPS (Sukmawan et al., 2023).

Kasus-kasus tersebut menunjukkan bahwa ancaman kebocoran data semakin nyata dan dapat menimpa berbagai sektor, termasuk layanan sosial seperti masjid. Oleh karena itu, keamanan data harus menjadi fokus utama dalam pengembangan aplikasi khususnya aplikasi MultiMajid. Untuk merancang keamanan data pada sebuah sistem, tiga aspek utama yang menjadi panduan dalam pembuatan kebijakan terkait keamanan data dan informasi adalah *Confidentiality*, *Integrity*, dan *Availability* atau disebut dengan Triad CIA (Fauzi et al., 2023). *National Institute of Standards and Technology* (NIST) telah menerapkan triad CIA sebagai salah satu metode konvensional untuk sistem penilaian ICT yang digunakan pemerintah federal di Amerika Serikat. Kelompok-kelompok profesional dan organisasi juga telah menyarankan penggunaan CIA dalam sistem penilaian keamanan (Shoufan & Damiani, 2017).

Penerapan pinsip Triad CIA menjadi peran penting untuk memastikan bahwa data pada aplikasi MultiMasjid tetap aman dan terlindungi. Aspek *Confidentiality* (kerahasiaan) merupakan aspek yang berkaitan dengan segala macam upaya untuk mencegah akses dan penyalahgunaan informasi data yang bersifat sensitif oleh individu atau sistem yang tidak memiliki otoritas. Aspek *Integrity* (integritas) memastikan bahwa informasi data hanya dapat diubah secara sah oleh pihak-pihak yang memiliki otoritas. Aspek *Availability* (ketersediaan) merupakan aspek yang menjamin bahwa data dan sistem selalu tersedia untuk digunakan saat dibutuhkan oleh pihak yang berwenang.

Penelitian ini bertujuan untuk menganalisis berbagai teknik dan praktik terbaik dalam penerapan keamanan data pada aplikasi MultiMasjid, serta memberikan rekomendasi untuk implementasi yang optimal guna melindungi data jamaah dan meningkatkan kepercayaan masyarakat.

1.2 Rumusan Masalah

Rumusan masalah yang dapat diambil berdasarkan latar belakang adalah:

a. Bagaimana mengatasi tantangan keamanan data dalam aplikasi MultiMasjid untuk melindungi data pribadi jamaah?
b. Teknik dan teknologi keamanan data apa saja yang dapat diimplementasikan dalam aplikasi MultiMasjid untuk memastikan kerahasiaan, integritas, dan ketersediaan data?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk merancang dan mengimplementasikan sistem keamanan data yang efektif pada arsitektur aplikasi MultiMasjid. Fokus utama penelitian ini adalah untuk memastikan kerahasiaan, integritas, dan ketersediaan data melalui penggunaan teknik-teknik seperti enkripsi menggunakan *Advanced Encryption Standard* (AES), autentikasi menggunakan *JSON Web Token* (JWT), dan pengendalian akses melalui *Cross-Origin Resource Sharing* (CORS).

1.4 Manfaat Penelitian

Manfaat yang diperoleh dari penelitian ini diantaranya:

a. Penelitian ini memberikan rekomendasi teknik dan praktik yang tepat untuk implementasi keamanan data pada arsitektur aplikasi MultiMasjid, yang dapat diterapkan oleh pengelola masjid untuk meningkatkan keamanan data jamaah.
b. Penelitian ini diharapkan dapat memberikan kontribusi pengetahuan baru dalam bidang keamanan data, khususnya terkait dengan pengelolaan aplikasi pada lingkungan keagamaan.
c. Membantu meningkatkan literasi digital dan kesadaran akan pentingnya melindungi data pribadi dalam era digital, terutama di kalangan pengelola dan jamaah masjid.

1.5 Batasan Penelitian

Agar penelitian ini tetap terarah dan dilakukan dengan semestinya, ditentukan beberapa batasan masalah yang terdiri dari:

a. Penelitian ini akan memfokuskan pada analisis dan implementasi keamanan pada arsitektur Aplikasi MultiMasjid.
b. Penelitian ini tidak akan membahas pengembangan aplikasi dari awal atau fitur yang tidak terkait dengan keamanan, seperti desain antarmuka pengguna atau fitur non-keamanan.

c. Analisis keamanan data meliputi kerahasiaan data, integritas data, dan ketersediaan data pada aplikasi.
d. Teknik keamanan data yang dianalisis dan diimplementasikan mencakup enkripsi, autentikasi, otorisasi, akses kontrol, dan teknik lainnya yang relevan.

1.6 Sistematika Penulisan

Penelitian ini disusun dalam lima bab yang saling melengkapi. Dimulai dengan pemaparan latar belakang, masalah, tujuan, manfaat, batasan, dan sistematika penulisan pada Bab I, penelitian ini secara bertahap mengungkapkan inti permasalahan dan solusinya. Bab II mendalami kajian literatur terkait variabel, konsep, serta teori utama yang relevan dengan topik penelitian, memperkuat landasan teoritis penelitian ini. Bab III memaparkan langkah-langkah metodis dalam pengembangan sistem keamanan data pada aplikasi MultiMasjid, mulai dari penentuan objek penelitian hingga rencana pengujian. Bab IV menyajikan hasil implementasi dan pengujian sistem, mengukur efektivitas solusi yang diajukan. Terakhir, Bab V merangkum kesimpulan dari penelitian ini serta memberikan saran konstruktif untuk penelitian selanjutnya.

BAB II
KAJIAN LITERATUR

2.1 Transformasi Digital Masjid

Kemajuan dalam teknologi informasi telah memberikan banyak manfaat, termasuk dalam mendukung berbagai aktivitas di masjid. Dengan bantuan teknologi, aktivitas ibadah dan non-ibadah di masjid dapat berjalan dengan lebih baik. Sebagai pusat peradaban Islam, masjid harus mampu menyesuaikan diri dengan kemajuan teknologi masa kini. Penyesuaian ini penting untuk menyeimbangkan gaya hidup generasi muda dan membangkitkan minat mereka untuk memakmurkan masjid. Selain itu, penyelarasan dengan teknologi juga penting untuk meningkatkan kualitas pengelolaan masjid agar lebih baik.

Terdapat permasalahan yang kerap ditemui di masjid, seperti pencatatan jamaah masjid yang masih manual sehingga dapat mengakibatkan kesalahan dalam pencatatan, kehilangan data, maupun kesulitan dalam pencarian data. Selain itu terdapat beberapa permasalahan yang terjadi seperti kesalahan dalam perhitungan uang yang tercatat dalam buku kas bendahara masjid, informasi jadwal kegiatan yang tidak tepat untuk jamaah, dan kurangnya media untuk memberikan informasi mengenai kegiatan dan perkembangan masjid kepada jamaah (Elsera et al., 2021). Dengan melibatkan teknologi sebagai sistem informasi masjid diharapkan dapat membantu mengatasi permasalahan yang kerap terjadi dalam pengelolaan masjid secara konvensional.

2.1.1 Tawaran Sistem

Penerapan sistem pada masjid dengan melibatkan teknologi sangat diperlukan untuk meningkatkan kualitas pengelolaan dan pelayanan kepada jamaah. Purnasari et al. (2022) membuat sebuah perancangan sistem informasi pengelolaan dana Masjid Al-Istiqomah Muara Jambi berbasis web dengan menggunakan UML *(Unified Modeling Language)*. Tujuan dibuat sistem tersebut yaitu merancang sistem informasi untuk mempermudah pengurus masjid dalam pengelolaan dana serta memberikan informasi keuangan masjid yang lebih cepat, tepat, dan akurat kepada jamaah. Fitur utama sistem tersebut mencakup pengelolaan data informasi, laporan keuangan, serta fitur melihat informasi dan mengunduh laporan bagi pengguna umum/jamaah.

Penelitian yang dilakukan oleh Herfandi dan Hamdani (2022) yaitu mengimplementasikan sistem informasi masjid berbasis *website* yang dibangun menggunakan

bahasa pemrograman PHP dan menggunakan *Database* MYSQL. Tujuan dari penelitian ini adalah untuk mengembangkan Sistem Informasi Manajemen (SIM) untuk masjid, yang bertujuan untuk membantu dan memudahkan pengelola masjid dalam mengelola aktivitas dan keuangan masjid. Fitur yang ada di dalam sistem tersebut dibagi menjadi 3 bagian yaitu, fitur untuk pengunjung: informasi jadwal sholat, informasi arah kiblat, informasi keuangan masjid, informasi donasi, dan informasi kontak, fitur untuk admin: mengelola data petugas, data kategori donasi, backup dan restore, cetak data petugas, dan melihat histori aktivitas, serta yang terakhir fitur untuk pengurus: mengelola agenda, mengelola keuangan, dan mengelola laporan.

Novryaldy dan Setiadi (2018) juga membuat sebuah perencanaan sistem informasi profil masjid dengan basis *website*. Tujuannya adalah membangun sistem yang dapat membantu masjid dan Kemenag (Kementerian Agama) dalam mengelola administrasi dan menyajikan informasi profil masjid. Sistem yang dibangun memiliki fitur pendataan jamaah, pengelolaan keuangan, pengelolaan kegiatan, dan menghasilkan informasi profil masjid, jamaah dan keuangan.

Berdasarkan beberapa penelitian sebelumnya, diketahui bahwa sudah banyak dilakukan pengembangan sistem informasi masjid untuk membantu pengelolaan dan pelayanan masjid agar menjadi lebih baik. Namun, dari penelitian-penelitian yang ada belum ditemukan yang secara khusus mengangkat isu keamanan data dan keamanan informasi pada sistem informasi masjid.

Beberapa penelitian seperti yang dilakukan oleh Purnasari et al. (2022), Herfandi dan Hamdani (2022), serta Novryaldy dan Setiadi (2018) lebih berfokus pada pengembangan fitur-fitur fungsional sistem informasi masjid seperti pengelolaan keuangan, pendataan jamaah, jadwal sholat, dan lain sebagainya. Namun belum menyentuh aspek keamanan data jamaah, keamanan database, maupun keamanan aplikasi secara menyeluruh.

Oleh karena itu, terdapat celah atau *gap* penelitian untuk melakukan pengembangan sistem informasi masjid dengan tidak hanya berfokus pada fungsionalitas tapi juga memperhatikan aspek keamanan. Penelitian di bidang ini penting untuk dilakukan mengingat data dan informasi masjid bersifat sensitif sehingga perlu dilindungi dari akses yang tidak berwenang atau disalahgunakan. Pengembangan sistem informasi masjid yang aman dari sisi data dan aplikasi diharapkan dapat menjadi kontribusi berarti dalam riset di bidang teknologi informasi khususnya untuk institusi masjid.

2.1.2 Strategi Pengamanan Data Pribadi Jamaah

Pengelolaan masjid tidak lepas dari kebutuhan pengelolaan data pribadi. Data pribadi adalah sesuatu yang harus dilindungi karena sejatinya merupakan hak privasi setiap orang. Data pribadi jamaah yang dikelola oleh masjid umumnya berupa data identitas seperti nama, alamat, no telepon, jenis kelamin, dan lain sebagainya. Data ini rentan disalahgunakan jika tidak dilindungi dengan baik. Penyimpanan dan pengelolaan data pribadi oleh masjid merupakan aspek penting dalam menjaga privasi individu dan mematuhi ketentuan hukum seperti Undang-Undang Nomor 27 Tahun 2022 tentang Perlindungan Data Pribadi. Undang-undang tersebut merupakan landasan hukum yang komprehensif di Indonesia terkait perlindungan data pribadi. UU ini mengatur berbagai aspek mulai dari prinsip-prinsip perlindungan data pribadi, hak dan kewajiban individu maupun pengelola data, sanksi pelanggaran, hingga pembentukan lembaga pengawas.

Undang-Undang Perlindungan Data Pribadi adalah amanat dari Pasal 28G ayat (1) Undang-Undang Dasar Negara Republik Indonesia Tahun 1945 yang menyatakan bahwa "setiap orang memiliki hak untuk melindungi privasi pribadi, keluarga, martabat, dan harta benda mereka, serta memiliki hak untuk merasa aman dan terlindungi dari ancaman ketakutan untuk melakukan atau tidak melakukan tindakan yang merupakan hak asasi". Kekhawatiran akan pelanggaran data pribadi yang dapat dialami oleh individu atau badan hukum menjadi alasan munculnya isu perlindungan data pribadi. Pelanggaran tersebut dapat berakibat kerugian material dan nonmaterial.

Dengan berlakunya Undang-Undang Perlindungan Data Pribadi ini, pengelolaan data pribadi harus mematuhi ketentuan yang ada untuk menghormati hak privasi individu sesuai dengan Pasal 16 E ayat (2) yang menyatakan "Pemrosesan Data Pribadi dilakukan dengan melindungi keamanan Data Pribadi dari akses yang tidak sah, pengungkapan yang tidak sah, perubahan yang tidak sah, penyalahgunaan, perusakan, dan/atau penghilangan Data Pribadi".

Menurut penelitian yang dilakukan oleh Yel dan Nasution (2022) beberapa strategi yang dapat dilakukan untuk menjaga keamanan data pribadi antara lain:

a. Identifikasi

   Salah satu ciri dari sistem informasi adalah kemampuannya untuk mengetahui siapa yang menggunakannya. Identifikasi adalah proses awal untuk mendapatkan izin untuk mengakses informasi yang dilindungi. Identifikasi yang berkaitan dengan keamanan biasanya menggunakan nama pengguna dan nomor identitas pengguna.

b. Autentikasi

Autentikasi adalah langkah yang diperlukan untuk memastikan keaslian suatu entitas, baik itu data atau pengguna sebelum mendapatkan izin untuk mengakses informasi. Suatu sistem atau seorang pengguna yang lolos proses autentikasi berarti telah dibuktikan kebenaran identitasnya sesuai dengan yang diakuinya sejak awal.

c. Otorisasi

Otorisasi adalah proses penentuan hak untuk mengakses, memanfaatkan, atau mengubah suatu data atau sistem. Dengan demikian, otorisasi adalah proses pengecekan atas kewenangan untuk mengakses suatu data atau sumber daya pada sistem. Secara urutan pelaksanaan, otorisasi adalah proses yang dilakukan setelah proses autentikasi berhasil diselesaikan. Dengan proses otorisasi, pengguna mendapatkan akses lengkap kepada berbagai jenis sumber daya seperti informasi, file, database, dan sebagainya sesuai dengan level dan hak akses yang dimilikinya.

d. Akuntabilitas

Akuntabilitas adalah proses pendokumentasian dan penghitungan semua tindakan yang berkaitan dengan akses dan pemakaian data atau sumber daya pada suatu sistem. Akuntansi digunakan untuk menguji proses otorisasi yang telah dilakukan di masa sebelumnya untuk menjamin keamanan di masa sekarang dan di masa yang akan datang.

e. Enkripsi data

Teknik ini dilakukan dengan mengacak data asli menjadi data yang tidak dapat terbaca tanpa kunci khusus. Enkripsi dapat diterapkan pada database maupun saat transmisi data melalui jaringan.

2.2 Keamanan Data

Keamanan data merupakan aspek penting dalam pengembangan suatu sistem informasi. Menurut buku Harun (2018:7-9) yang berjudul Kriptografi untuk Keamanan Data, Keamanan data meliputi beberapa aspek, yaitu:

a. Privasi (Kerahasiaan)

Privasi biasanya berkaitan dengan data yang bersifat rahasia. Aspek privasi merupakan upaya yang dilakukan untuk melindungi informasi yang bersifat rahasia dari penyalahgunaan oleh orang yang tidak memiliki hak akses. Terdapat tiga aspek privasi, yaitu privasi mengenai pribadi seseorang, privasi mengenai data seseorang, dan privasi mengenai komunikasi seseorang. Penggunaan data pribadi seseorang oleh lembaga pemerintah, perusahaan swasta,

ataupun perorangan tanpa izin pemiliknya merupakan pelanggaran terhadap privasi. Lembaga atau individu tidak berhak menggunakan informasi pribadi seseorang tanpa sepengetahuan dan persetujuan dari yang bersangkutan. Hal ini melanggar hak privasi seseorang atas data dirinya.

b. *Integrity* (Konsisten)

Integritas data adalah aspek penting yang memastikan bahwa data yang digunakan adalah data asli yang dikirimkan oleh pihak yang tepat. Integritas juga menjamin bahwa data tersebut tidak mengalami perubahan selama proses pengiriman. Salah satu metode yang sering digunakan untuk memastikan hal ini adalah enkripsi. Enkripsi adalah teknik yang digunakan untuk mengubah data atau kata menjadi kode-kode yang tidak dapat dimengerti oleh orang lain termasuk komputer. Untuk mengetahui data asli, diperlukan metode lain yaitu deskripsi. Deskripsi adalah proses mengubah kode-kode tersebut menjadi data asli. Ilmu yang mempelajari teknik ini disebut kriptografi.

c. *Authenticity* (Keaslian)

Aspek ini berkaitan dengan metode atau cara untuk memastikan bahwa informasi adalah asli, orang yang mengakses atau memberikan informasi adalah orang yang benar-benar orang yang memiliki hak.

d. *Availability* (Ketersediaan)

Aspek ini berhubungan dengan data yang dapat diakses dan digunakan oleh pengguna yang berwenang pada saat dibutuhkan. Ketersediaan menjamin para pengguna dapat mengakses informasi yang mereka butuhkan tanpa gangguan atau pencegahan dari pihak lain yang tidak memiliki hak akses. Dengan kata lain, aspek ketersediaan memastikan bahwa data dan sistem selalu tersedia dan dapat diakses pengguna yang berwenang setiap saat diperlukan.

e. Access Control

Access Control berkaitan dengan cara mengatur akses terhadap informasi. Hal ini umumnya terkait dengan klasifikasi data, mekanisme otentikasi, dan juga privasi. *Access Control* seringkali dilakukan dengan menggunakan kombinasi user id dan password atau dengan mekanisme lainnya. *Access Control* bertujuan untuk membatasi akses informasi hanya kepada pengguna yang berwenang. Dengan *Access Control*, informasi rahasia dan sensitif hanya dapat diakses oleh pengguna tertentu sesuai dengan hak akses yang dimiliki. Mekanisme ini melindungi kerahasiaan dan integritas data dari akses ilegal.

2.2.1 Peran Triad CIA Terhadap Keamanan Data dan Informasi

Dari kelima aspek keamanan data yang ada pada buku Harun (2018:7-9), terdapat 3 aspek utama yang menjadi komponen inti dalam aspek keamanan data. Kerahasiaan, Integritas, dan Ketersediaan adalah tiga atribut penting dari keamanan data, yang sering disebut sebagai Triad CIA (Kumar et al., 2018). Triad CIA (Kerahasiaan, Integritas, Ketersediaan) adalah model yang terkenal untuk pengembangan kebijakan keamanan. Model ini digunakan untuk mengidentifikasi masalah dan menemukan solusi yang diperlukan untuk keamanan sistem dan sistem informasi.

Berdasarkan penelitian yang dilakukan oleh Dani (2008) selama lebih dari dua dekade, sistem informasi telah dibangun berdasarkan tiga prinsip utama, yaitu Kerahasiaan, Integritas, dan Ketersediaan, yang sering dikenal dengan Triad CIA, yaitu:

a. *Confidentiality* (Kerahasiaan)

Kerahasiaan adalah upaya untuk mencegah penyingkapan informasi yang bersifat rahasia dan sensitif. Beberapa mekanisme yang digunakan untuk menjaga konsep kerahasiaan ini meliputi;

1. Klasifikasi Data

   Adalah metode untuk memberi label pada informasi sehingga setiap individu dapat mengetahui siapa yang memiliki otoritas untuk melihat informasi tersebut.

2. Enkripsi

   Adalah metode yang diterapkan untuk menjaga kerahasiaan (*Confidentiality)*

3. Pemusnahan Peralatan (*Equipment Disposal*)

   Merupakan berbagai upaya atau aktivitas yang ditujukan untuk melindungi kerahasiaan suatu informasi ketika tidak lagi digunakan dalam media penyimpanan.

b. *Integrity* (Integritas)

Integritas adalah kondisi di mana data tidak dapat diubah, dibuat, atau dihapus tanpa proses otorisasi. Integritas adalah prinsip yang ditujukan untuk menjaga kebenaran suatu informasi. Adapun menurut Dani (2008) tujuan dari *Integrity* adalah:

1. Mencegah perubahan informasi oleh pengguna yang tidak berwenang.
2. Mencegah akses atau modifikasi informasi yang tidak disengaja oleh pengguna yang tidak memiliki hak.
3. Menjaga konsistensi internal dan eksternal.

Contoh upaya yang dapat dilakukan untuk menjaga integritas data dan informasi adalah sebagai berikut:

1. *Checksums*

Adalah serangkaian angka yang dihasilkan oleh fungsi matematika untuk memastikan bahwa blok data yang diberikan tidak berubah.

2. Kontrol Akses

Adalah mekanisme yang memastikan bahwa hanya pihak tertentu yang dapat melakukan sejumlah aksi tertentu.

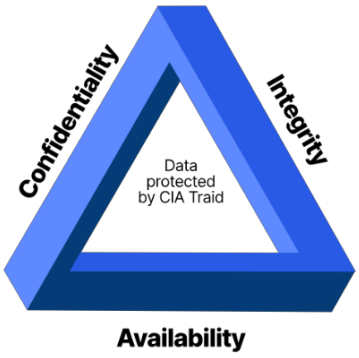
c. *Availability* (Ketersediaan)

Availability adalah konsep yang menjamin bahwa pengguna yang berhak mendapatkan akses tanpa gangguan sistem dan jaringan, dengan memastikan bahwa informasi atau sumber daya akan selalu tersedia saat dibutuhkan (Dani, 2008).

Untuk menjaga ketersediaan data dan informasi, beberapa langkah yang dapat diambil antara lain:

1. *Redundant system* atau implementasi ganda dalam infrastruktur. Hal ini merujuk pada proses penambahan komponen atau sistem tambahan ke dalam infrastruktur yang ada untuk meningkatkan reliabilitas dan ketersediaan
2. Menggunakan perangkat IPS yang bertujuan untuk mencegah ancaman serangan tertentu seperti DDoS yang dapat mengganggu layanan.

Konsep Triad CIA, yang mencakup Confidentiality (Kerahasiaan), Integrity (Integritas), dan Availability (Ketersediaan), diterapkan sebagai strategi utama dalam melindungi sistem informasi dan data. Hasil penelitian Qadir & Quadri (2016) telah menunjukkan bahwa penerapan konsep ini sangat penting dalam menjaga keamanan data dan informasi.



Gambar 2. 1 Triad CIA

Sumber: www.imperva.com

Gambar 2.1 menunjukkan konsep analogi hubungan dan keterkaitan *Triad CIA* dalam melindungi keamanan informasi dan data dalam sistem yang diterapkan. Ketiga faktor ini saling terkait dan saling mendukung ikatan satu sama lain. Oleh karena itu, keterkaitan *Triad CIA* sangat penting dalam menjaga keamanan sistem informasi dan data.

2.2.2 Keamanan Siber

Keamanan Siber merupakan serangkaian kegiatan dan tindakan yang dirancang untuk melindungi terhadap serangan, gangguan, atau ancaman lainnya yang ditujukan pada elemen-elemen *cyberspace* seperti perangkat keras, perangkat lunak, dan jaringan komputer (Fischer, 2005).

Menurut pendapat Jayantina (2018), keamanan siber terdiri dari dua komponen utama yaitu infrastruktur "lunak" dan infrastruktur "keras". Infrastruktur "lunak" mencakup Sumber Daya Manusia yang bertanggung jawab untuk mengelola dan membuat kebijakan keamanan (people), serta kebijakan proses, protokol, dan pedoman yang membentuk lingkungan yang aman untuk melindungi sistem dan data (*process*). Infrastruktur "keras" mencakup teknologi yang digunakan untuk mencegah dan mengatasi serangan siber, seperti perangkat keras, perangkat lunak, protokol komunikasi, dan alat enkripsi yang diperlukan untuk menjaga sistem dan data dari ancaman siber eksternal dan internal.

2.3 Ancaman Kebocoran Data

Kebocoran data merupakan suatu kondisi di mana informasi yang seharusnya tetap rahasia dan terlindungi dalam suatu sistem keamanan mengalami kebocoran dan tersebar ke luar sistem termasuk ke internet. Ada berbagai penyebab kebocoran data, mulai dari serangan yang direncanakan oleh penjahat siber dan peretas dengan niat jahat, serangan yang dilakukan oleh aktivis dengan motif ideologi tertentu (*hacktivist*), hingga kelalaian atau kesalahan dari pemilik atau pengelola sistem itu sendiri.

Ketika terjadi kebocoran data, peretas biasanya akan mengambil data-data sensitif seperti:

a. Informasi Pengguna

   Informasi Identitas: Termasuk nama, alamat, nomor telepon, nomor identitas, alamat email, username, kata sandi, dan lainnya.

   Aktivitas Pengguna: Meliputi riwayat pemesanan dan pembayaran, kebiasaan browsing, dan lainnya.

b. Informasi Kartu Kredit: Seperti nomor kartu, tanggal kadaluarsa, kode pos penagihan, dan lainnya.

Menurut laporan dari Risk Based Security (2019), selama sembilan bulan pertama tahun 2019, lebih dari 5000 insiden kebocoran data telah dilaporkan, naik 33,3% dibandingkan dengan periode yang sama pada tahun 2018. Dari total tersebut, 87% kebocoran data dilakukan oleh penyerang eksternal, 7% disebabkan oleh kelalaian dan kecerobohan dari pihak internal, dan 1,5% merupakan aksi balas dendam oleh pihak internal. Daftar kebocoran data besar-besaran sepanjang tahun 2019 telah didokumentasikan dalam beberapa artikel yang terus diperbarui, seperti yang dilakukan oleh Selfkey Foundation pada tahun 2019 dan IdentityForce, Inc (Rahma & Raf'ie, 2020).

Kasus kebocoran data seringkali terjadi karena dua fenomena yang saling berhubungan. Di satu sisi, era digitalisasi telah menyebabkan peningkatan jumlah data yang disimpan dalam format digital. Di sisi lain, nilai dari data tersebut semakin meningkat, sehingga menciptakan insentif finansial bagi pelaku kejahatan digital. Lebih jauh lagi, fenomena ekonomi cybercrime telah muncul, di mana insiden kebocoran data diikuti dengan transaksi finansial. Hal ini biasanya terjadi pada perusahaan yang menyediakan layanan seperti platform digital atau e-commerce.

2.3.1 Penyebab Kebocoran Data

Kasus kebocoran data sering terjadi karena penyebab, antara lain:

a. *Malware (Malicious Software)*

Perangkat lunak jahat, atau yang lebih umum dikenal sebagai *malware*, adalah salah satu ancaman keamanan tertua yang masih relevan hingga hari ini (Rahma & Raf'ie, 2020). *Malware* adalah perangkat lunak yang dirancang untuk merusak dengan menginfeksi sistem. Infeksi ini bisa terjadi melalui berbagai media, seperti email, unduhan internet, atau program yang terkontaminasi sehingga memungkinkan terjadinya pencurian informasi data. Seiring dengan perkembangan teknologi informasi, variasi dari *malware* juga terus berkembang dan menyebar di seluruh dunia.

b. Serangan *Hacker*

Serangan peretasan (hacking) merupakan salah satu ancaman utama yang dapat menyebabkan kebocoran data. Para hacker memiliki beragam teknik dan motif dalam melancarkan serangannya untuk mendapatkan akses tanpa otorisasi ke dalam sistem target. Beberapa cara umum yang dilakukan hacker antara lain melakukan serangan brute force

login dengan mencoba semua kemungkinan kombinasi kredensial, memanfaatkan celah keamanan sistem untuk remote exploitation, melakukan phishing untuk mendapatkan kredensial pengguna, melakukan serangan MITM (Man in the Middle) untuk mencuri data yang sedang ditransmisikan, dan lain sebagainya. Ancaman peretasan dari hacker ini sangat serius karena berpotensi membuat data sensitif bocor ke pihak yang salah.

c. *Human Error* atau ketidak sengajaan pengguna

Sebagian besar insiden kebocoran data terjadi karena kesalahan manusia. Misalnya, kebocoran terjadi ketika seorang programmer secara tidak sengaja membuat *database* menjadi terbuka untuk umum, sehingga data yang tersimpan di dalam database bisa berakibat kebocoran dan dapat diakses oleh siapa saja hingga akhirnya dikunci kembali. Ketika kesalahan semacam ini terjadi, mereka yang berniat meretas sistem dapat mengambil informasi rahasia dan memanfaatkannya. Meski kebocoran data tersebut tidak disengaja, dampaknya tetap sama, yaitu hukuman dan kerusakan reputasi.

2.4 Teknik Keamanan Data

Teknik keamanan data merupakan cara-cara yang digunakan untuk melindungi dari ancaman kebocoran data. Teknik-teknik ini bertujuan untuk menjaga kerahasiaan, integritas, dan ketersediaan data, baik daya yang disimpan di perangkat, disimpan dalam *database*, dikirim melalui jaringan, ataupun diproses oleh aplikasi.

Terdapat berbagai jenis teknik keamanan yang dapat diterapkan dalam mengamankan sebuah sistem, tergantung pada jenis dan sifat data yang ingin dilindungi. Beberapa contoh teknik keamanan adalah:

a. Enkripsi Data

Enkripsi adalah teknik yang digunakan untuk mengubah data menjadi format yang sulit dibaca, sehingga informasi yang terkandung di dalamnya tetap aman dan tidak dapat diakses tanpa proses dekripsi (proses pengembalian data terenkripsi menjadi format aslinya) terlebih dahulu. Kata enkripsi sendiri berasal dari Bahasa Yunani yaitu *kryptos* yang berarti tersembunyi atau rahasia (Rahmadi & Yunita, 2020). Jadi enkripsi merupakan proses yang digunakan untuk mengubah data asli atau *plaintext* menjadi format yang sulit dibaca atau *ciphertext*. Dengan kata lain, *ciphertext* adalah hasil dari enkripsi data, yang telah diubah sedemikian rupa sehingga tidak dapat dibaca dengan mudah.

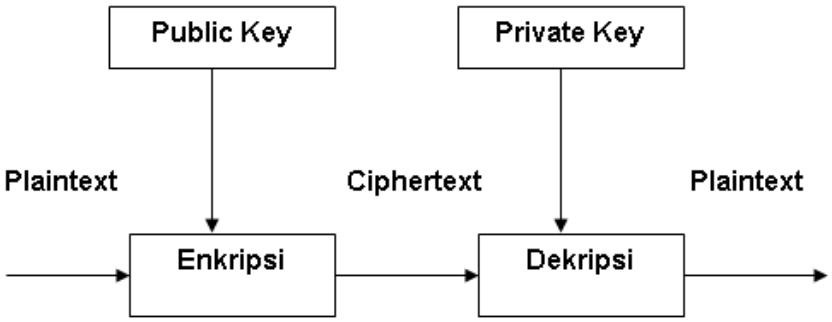
Pengetahuan yang mempelajari tentang enkripsi dinamakan kriptografi. Kriptografi adalah bidang studi yang berfokus pada teknik-teknik matematika yang terkait dengan

keamanan informasi. hal ini mencakup aspek-aspek seperti menjaga kerahasiaan data, memastikan keabsahan dan integritas data, serta melakukan autentikasi data. Menurut Rahmadi dan Yunita (2020) Kriptografi adalah studi dan juga seni dalam melindungi keamanan pesan. Sedangkan algoritma kriptografi adalah seperangkat aturan yang digunakan untuk proses enkripsi (mengubah pesan menjadi format tersembunyi) dan dekripsi (mengembalikan format tersembunyi menjadi pesan asli).

Menurut buku Harun (2018) Kriptografi untuk Keamanan Data, algoritma kriptografi dibagi menjadi dua bagian berdasarkan tipe kunci yang digunakan, dua bagian tersebut yaitu:

1. Algoritma Simetris

Algoritma simetris atau juga dikenal sebagai algoritma kriptografi konvensional, yaitu jenis algoritma dengan menggunakan kunci yang sama untuk melakukan proses enkripsi dan deskripsi. Ketika mengirim pesan menggunakan algoritma ini, penerima pesan harus diberitahu tentang kunci yang digunakan agar mereka dapat mendekripsi pesan tersebut. Keamanan pesan yang dienkripsi dengan algoritma ini sangat bergantung pada kerahasiaan kunci, jika kunci diketahui oleh orang lain, mereka akan dapat mengenkripsi dan mendekripsi pesan tersebut (Toyib & Wijaya, 2018). Beberapa contoh dari algoritma kunci simetris antara lain DES *(Data Encryption Standard), Blowfish, Twofish, MARS, IDEA, 3DES, RC2/4/5/6, dan AES (Advanced Encryption Standard).*



Gambar 2. 2 Proses Enkripsi/Dekripsi Algoritma Simetris

Algoritma AES (Advanced Encryption Standard) adalah salah satu algoritma yang populer dalam kriptografi. Algoritma ini memiliki blok berukuran 128 bit dan mendukung panjang kunci sebesar 128, 192, dan 256 bit. AES digunakan untuk proses enkripsi dan dekripsi informasi melalui serangkaian proses yang disebut ronde. Keunggulan AES terletak pada efisiensi biaya dan kemudahan implementasinya pada memori berukuran kecil (Mustika, 2020). AES dikenal sebagai algoritma kriptografi simetris yang banyak digunakan karena

kecepatan dan keamanannya yang cenderung lebih unggul dibandingkan dengan algoritma enkripsi simetris lainnya (Rahma & Raf'ie, 2020).

2. Algoritma Asimetris

Algoritma asimetri sering juga disebut dengan algoritma kunci public dengan kata lain algoritma asimetris merupakan jenis algoritma yang memanfaatkan kunci yang tidak sama untuk melakukan enkripsi dan dekripsi. Menurut Mukhtar Harun (2018) dalam algoritma ini ada dua jenis kunci, yaitu kunci umum dan kunci privat. Kunci umum dapat diakses oleh siapa saja dan digunakan untuk mengenkripsi pesan sebelum dikirim sedangkan kunci privat (*private key)* hanya diketahui oleh penerima pesan dan digunakan untuk mendekripsi pesan yang telah dienkripsi dengan kunci publiknya. Beberapa contoh dari algoritma enkripsi asimetris meliputi *Rivest–Shamir–Adleman (RSA), Diffie-Hellman, Digital Secure Algorithm (DSA), XTR, Elliptic Curve Cryptography (ECC),* dan *Elgamal Encryption System (ESS)*. Manfaat utama dari enkripsi asimetris adalah memiliki perlindungan yang kuat yang menjadikan pemulihan teks asli menjadi sulit dan tidak dapat ditebak oleh hacker.

Tabel 2. 1 Perbandingan Kriptografi Simetris dan Asimetris

Aspek Perbandingan	Kriptografi Simetris	Kriptografi Asimetris
Konsep dasar	Kunci yang sama untuk enkripsi dan dekripsi	Kunci dekripsi beda dengan kunci enkripsi
Julukan lain	Shared-secret key algorithm	Public key algorithm
Ukuran kunci	Pendek (40-256 byte)	Panjang (512-4096 byte)
Kecepatan komputasi	Lebih cepat, karena berbasis operasi matematika sederhana	Lebih lambat, karena berbasis algoritme komputasi yang rumi
Penggunaan	Digunakan untuk enkripsi data dalam ukuran besar	Digunakan untuk enkripsi data kecil

Contoh algoritma	AES, Blowfish, DES, 3DES, IDEA, RC2/4/5/6	ElGamal, Diffie-Hellman, elliptic curves, RSA

Sebuah penelitian yang pernah dilakukan Siregar et al. (2019) yaitu membuat penerapan algoritma kriptografi untuk mengamankan database mahasiswa pada Kampus Poltekkes Kemenkes Medan. Metode yang digunakan adalah algoritma RC4 untuk enkripsi simetris per byte dengan operasi XOR dan algoritma Rail Fence untuk transposisi karakter dengan pola zig-zag. Kombinasi kedua algoritma tersebut menghasilkan ciphertext yang lebih kuat. Hasilnya penerapan algoritma tersebut berupa aplikasi enkripsi dan dekripsi database mahasiswa berbasis web dengan PHP dan MySQL. Data mahasiswa yang sudah dienkripsi berubah menjadi simbol-simbol acak yang tidak bermakna.

Dalam keamanan data pada sistem yang dikembangkan, menggunakan teknik enkripsi di dalam database yang menyimpan data sensitif jamaah seperti nama, nomor HP, alamat, NIK, dan no KK sangat penting untuk melindungi kerahasiaan dan mencegah penyalahgunaan data tersebut. Jika data jamaah yang bersifat privat ini disimpan tanpa enkripsi, maka apabila database berhasil diakses oleh pihak yang tidak bertanggung jawab, data jamaah dapat dicuri dan disalahgunakan untuk tindak kejahatan seperti pencurian identitas atau penipuan. Penggunaan enkripsi kuat pada database jamaah mutlak diperlukan agar data pribadi tersimpan aman dan terjaga kerahasiaannya meskipun database berhasil disusupi. Enkripsi database adalah cara efektif melindungi privasi jamaah serta mencegah dampak negatif dari pencurian atau kebocoran data.

b. Autentikasi dan Otorisasi

Autentikasi adalah proses verifikasi identitas pengguna. Proses ini biasanya melibatkan penggunaan *ID login* dan kata sandi sebagai bukti bahwa pengguna tersebut otentik (Syofian et al., 2018). Menurut Noor dan Azam (2016) autentikasi adalah proses identifikasi atau pengenalan, baik secara sistem maupun informasi. Dalam komunikasi antara dua pihak, mereka saling memperkenalkan diri sebagai validasi pengguna untuk mengakses sistem. Maka autentikasi dapat diartikan sebagai proses identifikasi yang dilakukan satu pihak terhadap pihak lain, atau sebaliknya. Tujuannya adalah untuk memastikan keaslian dari informasi yang diperoleh. Dengan kata lain, autentikasi merupakan mekanisme yang dilakukan oleh suatu pihak untuk mengidentifikasi pihak lainnya, atau sebaliknya, guna memastikan validitas dan keaslian informasi yang dipertukarkan di antara mereka.

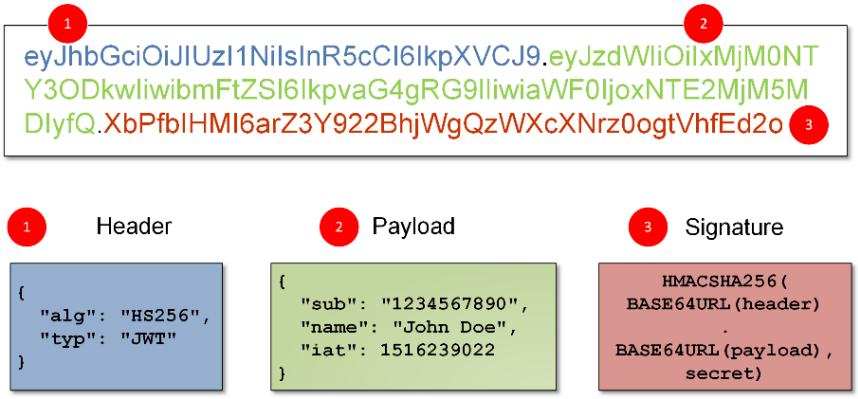
Sedangkan otorisasi adalah kumpulan aturan yang menentukan batas akses pengguna yang telah ter-autentikasi. Hal tersebut menentukan peran pengguna dalam sistem, apa yang dapat mereka akses dan apa yang tidak. Proses ini melibatkan beberapa tahap, antara lain:

a. Manajemen Kebijakan
   Menentukan apa yang perlu dilindungi dan siapa yang dapat mengakses area tertentu.

b. *Access Control*
   Mekanisme pembatasan akses yang paling dekat dengan sumber daya. Kontrol akses sangat spesifik terhadap sumber daya atau perangkat yang digunakan.

Dalam urutan pelaksanaan, proses otorisasi berlangsung setelah autentikasi berhasil. Melalui proses otorisasi ini, pengguna mendapatkan hak akses sepenuhnya ke berbagai jenis sumber daya seperti informasi, file, database, dan lainnya sesuai dengan level dan hak akses yang telah ditetapkan.

Salah satu teknik pengamanan dalam proses autentikasi dan autorisasi adalah penggunaan JSON Web Token. JSON Web Token (JWT) adalah format standar yang digunakan untuk mengamankan informasi pribadi menjadi klaim yang di-*encode* dalam format JSON dan menjadi bagian dari JSON Web *Signature*. Klaim ini dapat dilindungi dengan tanda tangan digital seperti Message Authentication Code (MAC) atau dapat dienkripsi. JWT adalah token berbentuk *string* yang terdiri dari tiga bagian: *header*, *payload*, dan *signature*, yang digunakan dalam proses otentikasi dan pertukaran informasi. Menurut Rahmatulloh et al. (2019), JWT paling sering digunakan pada server API untuk multiplatform seperti Android, web, IOS, desktop, dan lainnya. JWT tidak tergantung pada bahasa pemrograman tertentu. Struktur JWT terdiri dari tiga bagian yang dipisahkan oleh titik (“.”), yaitu header, payload, dan signature, seperti pada gambar 2.2 di bawah (Mahindrakar & Pujeri, 2020).



Gambar 2. 3 Konsep JWT

Bagian Header berisi informasi seperti jenis token yang digunakan, dalam hal ini adalah JSON Web Token (JWT), dan algoritma hashing yang digunakan, seperti SHA-256, SHA-512, RSA, dan lainnya. Bagian kedua merupakan Payload, yaitu berisi sebuah klaim yang menyatakan tentang entitas seperti hak akses ke token, ID pengguna, pemberi token, dan tanggal kedaluarsa token. Bagian ketiga dari JWT adalah signature, yang merupakan hasil hash dari header, payload, dan secret key. Apabila isi Header atau Payload diubah secara tidak sah, maka Signature akan menjadi tidak valid. Signature, yang dibuat dengan menggunakan header dan payload, memungkinkan JWT untuk memberikan tingkat keamanan bagi pengguna aplikasi.

c. Penerapan Kontrol Akses pada API

Dalam proses perancangan aplikasi, seringkali diperlukan penggunaan API (*Application Programming Interface*) untuk memungkinkan aplikasi terhubung dengan server yang bertujuan bertujuan untuk mengambil data atau menjalankan operasi-operasi tertentu. Keamanan komunikasi antar aplikasi dengan API sangatlah penting untuk menjaga integritas dan kerahasiaan data yang ditransmisikan. Keamanan tersebut melibatkan penerapan protokol keamanan yang kuat untuk mencegah akses tidak sah yang dapat mengancam sistem.

Kontrol akses merupakan protokol keamanan yang kuat untuk mencegah akses tidak sah yang dapat mengancam sistem adalah penerapan kontrol akses. Kontrol akses adalah mekanisme keamanan yang dirancang untuk menjaga kerahasiaan data dengan memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses objek data tertentu. Mekanisme kontrol akses bertugas untuk memverifikasi hak akses, sesuai dengan otorisasi yang sudah diberikan sebelumnya (Patil & Meshram, 2012). Dalam penerapan keamanan suatu aplikasi, kontrol akses adalah salah satu aspek penting, terutama pada pembuatan aplikasi yang terhubung dengan API. Kontrol akses bertujuan untuk membatasi dan mengatur siapa saja yang dapat mengakses sumber daya tertentu, seperti API endpoint.

Salah satu metode keamanan yang dapat digunakan untuk melakukan kontrol akses API pada sistem yang sedang dikembangkan adalah dengan menerapkan *Cross-Origin Resource Sharing* (CORS). CORS merupakan standar keamanan yang memungkinkan API server untuk menentukan domain mana saja yang diperbolehkan untuk mengakses sumber dayanya. Hal tersebut sangat penting untuk keamanan API server karena memungkinkan server untuk mengontrol akses ke sumber daya dari aplikasi yang berjalan di domain yang berbeda.

Salah satu *header* CORS yang paling penting adalah *Access-Control-Allow-Origin*. *Header* ini digunakan oleh server untuk menentukan domain mana saja yang diizinkan untuk

mengakses sumber daya API. Selain itu, server juga dapat mengatur header CORS lainnya seperti *Access-Control-Allow-Methods* untuk menentukan metode (GET, POST, PUT, DELETE) yang diizinkan, *Access-Control-Allow-Headers* untuk menentukan header permintaan yang diizinkan, serta *Access-Control-Max-Age* untuk menyetel waktu maksimum preflight request dapat disimpan dalam cache.

2.5 Kebutuhan Pada Masjid Modern

MultiMasjid merupakan sebuah aplikasi yang dirancang khusus untuk memudahkan para takmir dalam mengelola masjid. Dalam pengembangannya, aplikasi MultiMasjid menerapkan standar keamanan data dan informasi yang tinggi.

Salah satu keamanan yang akan diterapkan pada aplikasi tersebut adalah enkripsi data pada database menggunakan algoritma AES. Data yang seharusnya dirahasiakan berpotensi untuk disalahgunakan atau diakses oleh pihak yang tidak bertanggung jawab. Pelaku kejahatan selalu mencari metode untuk mengakses data tersebut, salah satunya dengan cara mengakses langsung tabel database. Oleh karena itu, perlindungan yang lebih baik terhadap database sangat diperlukan untuk mencegah akses ilegal. Data sensitif seperti data jamaah akan disimpan dalam bentuk terenkripsi sehingga terjaga kerahasiaan dan keamanannya. Meskipun database berhasil disusupi, data yang diperoleh berada dalam bentuk yang tidak dapat dipahami tanpa kunci dekripsi.

MultiMasjid juga menerapkan sebuah keamanan JSON Web Token (JWT) untuk proses authentikasi dan otorisasi pengguna. Setelah login, pengguna akan mendapatkan JWT yang berisi informasi identitas. JWT ini disertakan pada setiap permintaan untuk melakukan otentikasi dan otorisasi tanpa perlu menyimpan sesi. Penggunaan JWT membuat aplikasi lebih aman karena JWT bersifat *self-contained* dan memiliki masa berlaku tertentu.

Lebih lanjut, MultiMasjid akan menerapkan keamanan *Cross-Origin Resource Sharing* (CORS) sebagai keamanan API server, yaitu mekanisme keamanan yang mengatur server untuk secara spesifik menyatakan domain mana saja yang diperbolehkan untuk mengakses sumber daya. Dengan mengonfigurasi CORS, sistem dapat membatasi akses ke sumber daya hanya pada domain yang telah ditentukan, sehingga mencegah penyalahgunaan data oleh pihak yang tidak berwenang.

Dengan menerapkan standar keamanan berupa enkripsi pada database, JWT dan CORS, MultiMasjid diharap dapat menyediakan platform pengelolaan masjid terintegrasi yang handal dan terjaga privasinya untuk melayani para takmir dan jamaah masjid.

2.6 Penggunaan MongoDB pada Aplikasi MultiMasjid

Database yang akan digunakan dalam proses penyimpanan data pada aplikasi MultiMasjid adalah *MongoDB*. *MongoDB* adalah *database NoSQL* yang mendukung skalabilitas horizontal, memungkinkan aplikasi untuk menangani volume data yang besar dengan efisien dan mendukung pertumbuhan aplikasi seiring waktu. Pemilihan *MongoDB* sebagai database untuk aplikasi MultiMasjid didasarkan pada pertimbangan ilmiah yang mendukung efektivitasnya dalam memenuhi kebutuhan aplikasi.

Penelitian yang dilakukan oleh (Aan et al., 2016) menjelaskan bahwa *MongoDB* memiliki performa yang lebih baik dibandingkan MySQL dalam hal kecepatan eksekusi perintah seperti insert data, update data, dan delete data. Selain itu, MongoDB juga lebih efisien dalam ukuran penyimpanan data dibandingkan MySQL dan mampu menangani transaksi per-detik yang lebih banyak dibandingkan MySQL. Fleksibilitas skema *MongoDB* memungkinkan penyimpanan data dalam format *JSON-like*, mempermudah penyesuaian skema data tanpa perubahan struktural besar (Chodorow & Dirolf, 2019). Kinerja tinggi dalam pengolahan *query* dan transaksi serta dukungan untuk data terstruktur dan tidak terstruktur menjadikannya pilihan yang tepat untuk aplikasi ini.

MongoDB, sebagai salah satu basis data NoSQL terkemuka, menawarkan fleksibilitas yang luar biasa dalam pengelolaan data yang beragam dan dinamis, menjadikannya pilihan yang ideal untuk digunakan aplikasi MultiMasjid. Dalam pengelolaan masjid, MongoDB memungkinkan penyimpanan dan pengelolaan data jamaah yang kompleks, mencakup informasi penting seperti nama, alamat, nomor telepon, serta data identitas lainnya yang sering kali memerlukan pembaruan atau modifikasi secara berkala. Struktur dokumen JSON yang digunakan oleh MongoDB memungkinkan perubahan dan penambahan data ini dilakukan secara efisien tanpa perlu melakukan penyesuaian skema yang rumit, yang biasanya diperlukan dalam sistem basis data relasional tradisional. Selain itu, MongoDB sangat efektif dalam mengelola data terkait berbagai kegiatan sosial dan program-program yang diinisiasi oleh aplikasi Multi Masjid, seperti pembagian zakat, pengajian, dan bakti sosial. Setiap elemen dari program ini, termasuk data peserta, penerima manfaat, dan catatan donasi, dapat disimpan dalam dokumen yang terstruktur namun fleksibel, memungkinkan pengelola masjid untuk mengakses, memantau, dan menganalisis data secara mendalam guna memastikan pelaksanaan program berjalan sesuai rencana.

2.6.1 Dukungan MongoDB dalam Keamanan Sistem

Keamanan data merupakan aspek yang sangat krusial dalam pengelolaan informasi sensitif, terutama dalam aplikasi MultiMasjid yang menangani data pribadi jamaah. Pengelolaan data seperti ini membutuhkan sistem yang tidak hanya mampu menangani volume data yang besar, tetapi juga menjamin keamanan dan integritas informasi dari potensi ancaman dan kebocoran. MongoDB, sebagai sistem manajemen basis data *NoSQL*, menawarkan berbagai fitur keamanan yang mendukung perlindungan data secara komprehensif dan dapat diandalkan dalam pada pengembangan aplikasi MultiMasjid.

Salah satu fitur utama yang ditawarkan MongoDB adalah enkripsi data, baik saat disimpan (*encryption at rest*) maupun selama proses transmisi (*encryption in transit*). MongoDB menggunakan algoritma Advanced Encryption Standard (AES) dengan kunci enkripsi sepanjang 256 bit (AES-256), yang merupakan standar untuk keamanan data. Algoritma ini telah terbukti kuat dalam melindungi data sensitif dari akses yang tidak sah. Implementasi enkripsi *at rest* melindungi informasi yang disimpan, seperti data pribadi jamaah, dari akses fisik yang tidak sah terhadap perangkat keras penyimpanan. Sementara itu, enkripsi *in transit* memastikan bahwa data yang dikirimkan antara aplikasi dan server MongoDB tidak dapat disadap oleh pihak ketiga yang berpotensi melakukan tindakan yang merugikan.

Selain enkripsi, MongoDB juga menyediakan mekanisme backup dan restore yang efisien dan cepat. Fitur ini memungkinkan pemulihan data secara menyeluruh jika terjadi kegagalan sistem atau insiden keamanan yang mengganggu operasional aplikasi. Proses backup dapat diotomatisasi dan dilakukan secara berkala, sehingga data selalu dalam kondisi siap untuk dipulihkan tanpa kehilangan informasi penting. Ini sangat penting dalam menjaga kontinuitas layanan aplikasi MultiMasjid, memastikan bahwa informasi jamaah dan aktivitas masjid tetap aman dan tersedia meskipun terjadi situasi darurat.

Keamanan MongoDB tidak hanya terbatas pada enkripsi dan backup. Sistem ini juga terintegrasi dengan baik dengan mekanisme keamanan lain yang digunakan dalam aplikasi MultiMasjid, seperti JSON Web Token (JWT) untuk autentikasi dan otorisasi pengguna. JWT memungkinkan aplikasi untuk mengelola sesi pengguna dengan aman, memastikan bahwa setiap permintaan yang dilakukan oleh pengguna telah diverifikasi dan otorisasinya sesuai dengan hak akses yang diberikan. Selain itu, MongoDB mendukung Cross-Origin Resource Sharing (CORS), yang memberikan kontrol granular terhadap akses API dari berbagai domain, sehingga melindungi aplikasi dari serangan cross-site request forgery (CSRF) dan meminimalkan risiko eksposur data ke sumber yang tidak dipercaya.

Dengan kombinasi fitur-fitur keamanan yang canggih, MongoDB tidak hanya menjamin perlindungan data yang optimal tetapi juga memperkuat kepercayaan pengguna terhadap integritas dan reliabilitas sistem yang digunakan. Pengguna dapat merasa aman bahwa data pribadi dan transaksi yang mereka lakukan melalui aplikasi MultiMasjid dikelola dengan standar keamanan tertinggi, menjadikan MongoDB pilihan yang sangat andal dalam pengelolaan informasi yang sensitif dan bernilai tinggi.

BAB III
METODOLOGI PENELITIAN

3.1 Objek Penelitian

Objek penelitian adalah perilaku atau fenomena yang terjadi pada subjek penelitian dan diobservasi selama penelitian. Objek penelitian sangat menentukan desain penelitian, karena mempengaruhi perumusan masalah, batasan penelitian, tujuan, manfaat, dan metodologi yang digunakan. Pemilihan objek yang tepat merupakan dasar penting dalam merancang dan melaksanakan sebuah penelitian. Objek pada penelitian ini adalah implementasi JSON Web Token, penerapan kebijakan akses kontrol API menggunakan CORS (*Cross Origin Resource Sharing*) dan penerapan enkripsi database menggunakan kriptografi asimetris AES sebagai pilihan keamanan data pada aplikasi MultiMasjid.

3.2 Pengumpulan Data

Pentingnya sumber data dalam proses penelitian tidak bisa diabaikan, dan kekurangan akan akses sumber data bisa menghambat kemajuan dalam proses penelitian. Menerapkan metode pengumpulan data yang sesuai akan memudahkan peneliti dalam mencapai tujuan penelitian yang dijalankan. Dalam penelitian ini, data sekunder akan dijadikan sebagai sumber data utama. Data sekunder yang didapat dari penelitian ini berasal dari studi pustaka (*library research*) yaitu dengan melakukan pencarian terhadap jurnal, buku, dan artikel yang berkaitan tentang jenis-jenis sistem keamanan, metode pengembangan sistem keamanan, dan pengujian terhadap sistem keamanan yang sesuai dengan topik permasalahan yang diangkat. Tujuan dari pendekatan ini adalah untuk mendapatkan pemahaman yang lebih dalam dan memperkuat dasar penelitian yang akan dilakukan.

3.3 Indikator Keberhasilan

Indikator keberhasilan merupakan hal yang dapat menunjukkan tercapainya tujuan dalam sebuah penelitian. Keberhasilan suatu penelitian ditentukan oleh terpenuhi atau tidaknya tujuan yang sudah ditetapkan sebelumnya. Adapun indikator keberhasilan pada penelitian ini sebagai berikut:

a. Setiap permintaan mengakses data pada sistem harus melalui proses otentikasi dan otorisasi dengan memverifikasi JWT (JSON Web Token) yang dikirimkan pengguna. JWT tersebut harus valid dan ditandatangani dengan benar untuk membuktikan identitas pengguna yang mengakses. Ketika pengguna berhasil login, mereka akan mendapatkan token JWT. Token JWT ini selanjutnya digunakan untuk otentikasi pada setiap permintaan akses data,

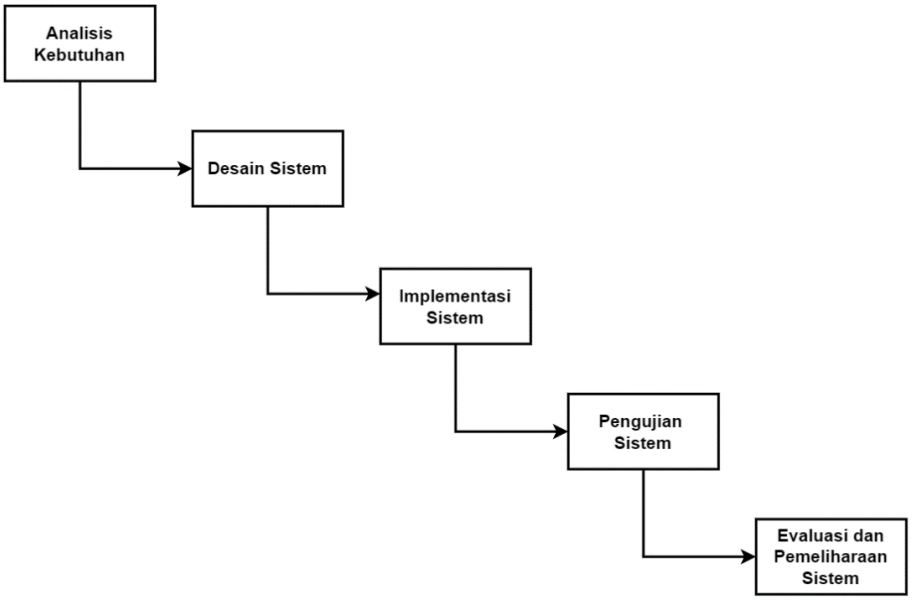
sehingga pengguna hanya dapat melihat dan menginput data jika permintaan tersebut menyertakan token JWT yang valid. Sebaliknya, permintaan tanpa token JWT yang valid akan ditolak dan pengguna tidak akan dapat mengakses data

b. Terenkripsinya data penting di database dengan metode kriptografi kunci asimetris Advanced Encryption Standard (AES) sehingga data menjadi aman dan tidak dapat diakses secara ilegal. AES adalah advance enkripsi yang menggunakan kunci simetris untuk enkripsi dan dekripsi data. AES lebih unggul dibandingkan enkripsi lain karena menggunakan ukuran blok dan panjang kunci yang lebih besar sehingga lebih sulit untuk dipecahkan. Algoritma AES juga lebih cepat dan efisien dalam proses enkripsi dan dekripsi data dalam jumlah besar.
c. Sistem berhasil memblokir akses ke sumber daya API dari domain yang tidak terdaftar dalam konfigurasi CORS. Permintaan dari domain yang tidak diizinkan akan mendapatkan respon error terkait kebijakan CORS saat mencoba mengakses API. server akan memeriksa asal domain dari setiap permintaan dan akan menolak permintaan dari domain yang tidak terdaftar dalam konfigurasi CORS.

3.4 Metode Penelitian

Agar tujuan penelitian dapat tercapai, diperlukan serangkaian proses dalam menjalankan penelitian tersebut. Proses-proses sistematis ini disebut dengan metode penelitian. Metode penelitian menjadi penting karena menunjukkan tahapan-tahapan yang harus dilakukan selama penelitian berlangsung. Dengan mengikuti metode penelitian yang tepat, peneliti dapat melakukan penelitian secara terstruktur sehingga tujuan akhir penelitian dapat tercapai.

Dalam penelitian ini, metodologi yang digunakan adalah model *waterfall* yang terdiri dari lima tahap, yaitu (1) Analisis Kebutuhan (*Requirement Definition*), (2) Desain Sistem (*System and Software Design*), (3) Implementasi Sistem (*Implementation and Unit Testing*), (4) Pengujian Sistem (*Integration and System Testing*), (5) Evaluasi dan Pemeliharaan Sistem. Penulis memilih model ini karena model ini lebih mudah dipahami serta dalam pengerjaannya secara berurutan, apabila tahapan sebelumnya belum selesai maka tahapan selanjutnya belum bisa dikerjakan. Tahapan dari model waterfall dapat dilihat pada gambar di bawah ini:



Gambar 3. 1 Metode Waterfall

Pada tahap analisis kebutuhan dilakukan pengumpulan kebutuhan secara lengkap kemudian dianalisis dan didefinisikan kebutuhan yang harus dipenuhi oleh sistem yang akan dibangun pada penelitian ini. Setelah menyelesaikan pengumpulan data kebutuhan sistem, langkah berikutnya adalah merancang desain sistem. Tahap ini akan menghasilkan berbagai rancangan yang diperlukan dalam proses pengembangan sistem. Beberapa elemen yang akan dirancang dalam sistem ini meliputi *use case* diagram, *activity* diagram, rancangan antarmuka, dan rancangan basis data. Selanjutnya penelitian akan dilanjutkan dengan tahap implementasi sistem. Pada tahap ini akan mengeksekusi segala rancangan-rancangan berdasarkan hasil dari tahap desain sistem. Tahap selanjutnya yaitu pengujian sistem. Pada tahap pengujian sistem, sistem yang telah selesai dibuat akan diuji coba. Tujuan pengujian adalah memverifikasi dan memvalidasi apakah sistem berjalan dengan benar sesuai harapan atau tidak. Kemudian diakhiri dengan tahap evaluasi dan pemeliharaan. Sistem yang sudah jadi dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya.

3.5 Analisis Kebutuhan

Analisis kebutuhan merupakan proses yang dilakukan agar mendapatkan informasi terkait kebutuhan sistem keamanan data pada aplikasi MultiMasjid. Pada tahap analisis kebutuhan ini, semua kebutuhan yang diperlukan dalam membangun sistem keamanan akan dianalisis dalam proses input data, data proses, hasil input data. Sebelum melakukan proses analisis kebutuhan terdapat tambahan mengenai gambaran umum aplikasi dan sistem

keamanan yang akan diimplementasikan di dalam aplikasi yang akan dijelaskan lebih lanjut sebagai berikut.

3.5.1 Gambaran Umum

Untuk memudahkan pemahaman pembaca tentang sistem keamanan Aplikasi MultiMasjid ini, berikut adalah penjelasan mengenai gambaran umum sistem.

a. Tentang Aplikasi MultiMasjid

Aplikasi MultiMasjid merupakan aplikasi yang dirancang untuk mengelola data jamaah dan kegiatan masjid secara digital. Aplikasi ini memiliki dua fitur utama, yaitu:

1. Data Jamaah: Fitur ini memungkinkan pengelola masjid untuk mencatat dan mengelompokkan data jamaah masjid berdasarkan nama, alamat, nomor telepon, Nomor KTP dan KK. Fitur ini dapat digunakan untuk melakukan klasifikasi data jamaah dalam pembagian zakat, pembagian daging qurban serta dapat mengirimkan informasi dan undangan kegiatan masjid kepada jamaah melalui SMS, WhatsApp, atau aplikasi lainnya.
2. Kegiatan Masjid: Fitur ini memfasilitasi pengelola masjid untuk merencanakan dan mengkoordinasikan kegiatan masjid, seperti kajian, pengajian, tadarus, bakti sosial, dan lain-lain. Fitur ini juga dapat menampilkan agenda kegiatan masjid yang terbaru dan terdekat.

b. Tentang Sistem Keamanan Data pada Aplikasi MultiMasjid

Keamanan dalam aplikasi ini sangat penting untuk melindungi informasi sensitif yang dimiliki oleh masjid dan penggunanya. Untuk itu, Aplikasi MultiMasjid mengimplementasikan mekanisme keamanan melalui otentikasi dan otorisasi dengan JSON Web Token (JWT), penerapan kebijakan akses kontrol API menggunakan CORS (*Cross Origin Resource Sharing*), dan enkripsi database menggunakan kriptografi Advanced Encryption Standard (AES).

Otentikasi dan otorisasi dilakukan menggunakan JWT di mana setiap pengelola masjid harus melakukan proses *login* terlebih dahulu untuk mendapatkan Token sebelum dapat mengakses fitur aplikasi termasuk fitur pengelolaan data jamaah masjid. Saat pengelola masjid mencoba melakukan proses *input* data jamaah, sistem akan memeriksa token JWT mereka. Jika token tidak valid, pengelola tidak akan dapat memasukkan data. Hal ini membantu memastikan bahwa hanya pengguna yang berwenang yang dapat memodifikasi data dalam aplikasi.

Sistem akan memblokir akses ke sumber daya API dari domain yang tidak terdaftar dalam konfigurasi CORS. Permintaan dari domain yang tidak diizinkan akan mendapatkan respon error terkait kebijakan CORS saat mencoba mengakses API. server akan memeriksa asal

domain dari setiap permintaan HTTP dan akan menolak permintaan dari domain yang tidak terdaftar dalam konfigurasi CORS.

Data jamaah seperti nama, alamat, nomor KTP, dan KK akan dienkripsi dan disimpan di dalam database. Enkripsi dilakukan menggunakan algoritma dan kunci enkripsi AES dengan panjang 256-bit yang cukup aman untuk melindungi kerahasiaan data. Proses enkripsi dan dekripsi dilakukan secara otomatis di server sehingga data tersimpan dalam bentuk terenkripsi di dalam database.

c. Fungsi Sistem Keamanan

Dengan mengimplementasikan JWT, CORS dan AES, Aplikasi MultiMasjid menghadirkan tiga lapisan keamanan: pertama, melakukan otorisasi dan otentikasi pengguna secara efektif melalui JWT yang memberikan kontrol yang aman terhadap akses sistem. Kedua, membatasi akses API hanya dari domain yang diizinkan dengan CORS untuk mencegah permintaan yang tidak sah. Ketiga, memastikan bahwa data yang disimpan dalam database terlindungi dari akses tidak sah melalui enkripsi AES yang kuat. Kombinasi ketiga metode ini memberikan jaminan keamanan yang maksimal untuk aplikasi dan data pengguna.

3.5.2 Analisis Kebutuhan Input

Input merujuk pada data dan informasi yang dimasukkan ke dalam sistem untuk diolah. Input merupakan titik awal dari setiap proses sistem. Dalam proses input kebutuhan yang diperlukan antara lain:

a. Identitas pengguna untuk autentikasi yaitu berupa *username* dan *password*.
b. Data jamaah meliputi, Nama, Nomor telepon, Nomor KTP, dan KK.

3.5.3 Analisis Kebutuhan Proses

Input merujuk pada data dan informasi yang dimasukkan ke dalam sistem untuk diolah. Input merupakan titik awal dari setiap proses sistem. Dalam proses input kebutuhan yang diperlukan antara lain:

a. Identitas pengguna untuk autentikasi yaitu berupa *username* dan *password*.
b. Data jamaah meliputi, Nama, Nomor telepon, Nomor KTP, dan KK.

3.5.4 Analisis Kebutuhan Output

Output merupakan hasil dari proses yang dilakukan oleh sistem. Output dapat berupa data, informasi, atau tindakan yang dihasilkan oleh sistem sebagai respon terhadap input yang diterima. Dalam proses output kebutuhan yang diperlukan antara lain:

a. Token JWT yang dikirim ke pengguna setelah berhasil masuk.

b. Informasi tentang keberhasilan atau kegagalan akses data berdasarkan validitas token
c. Data yang tersimpan di database dalam bentuk terenkripsi.
d. Data yang telah dienkripsi ditampilkan dalam bentuk aslinya setelah proses dekripsi.

3.5.5 Analisis Kebutuhan Sistem

Untuk membuat sistem keamanan data pada aplikasi MultiMasjid membutuhkan beberapa kebutuhan sistem yang akan mendukung penyelesaian dari penelitian ini. Kebutuhan sistem pada penelitian ini dapat dilihat pada Tabel 3.1 di bawah ini.

Tabel 3. 1 Kebutuhan Sistem

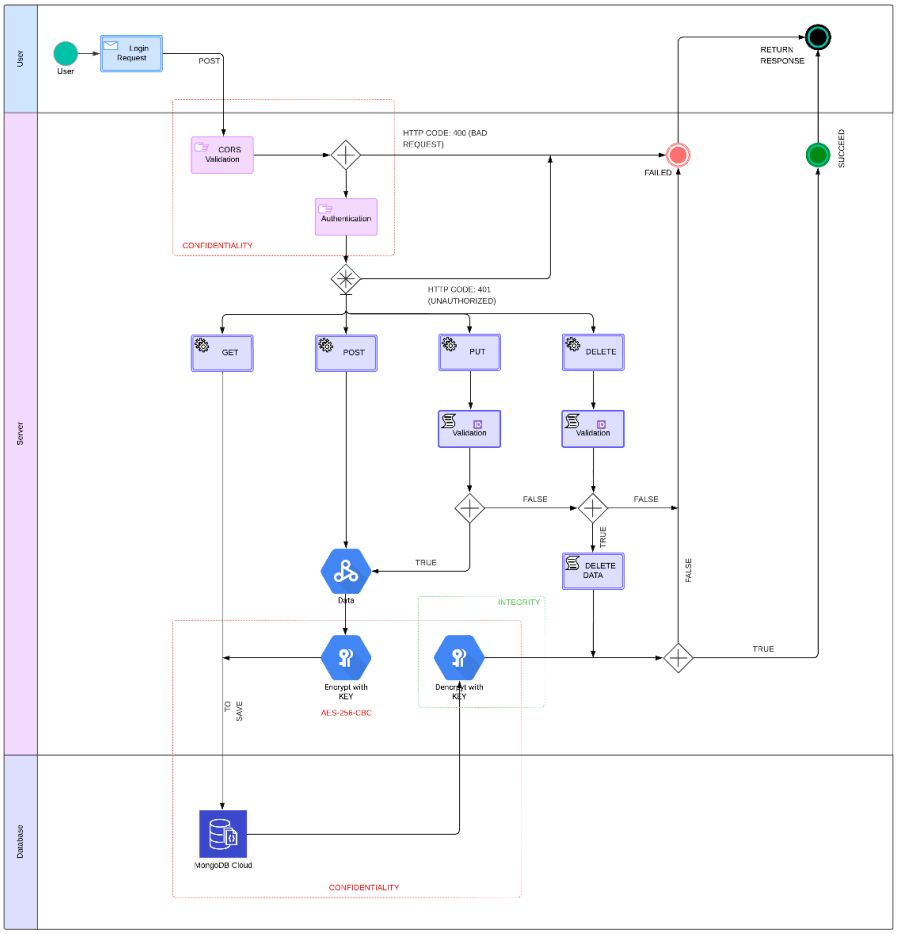
Teknologi	Keterangan	Spesifikasi
Visual Studio Code	Aplikasi teks editor yang digunakan untuk menulis kode program dimana mendukung banyak bahasa pemrograman.	v1.87.0
Draw.io	Aplikasi yang dipergunakan sebagai alat untuk merancang use case diagram dan flowchart	https://app.diagrams.net/
MongoDB	Sebagai basis data NoSQL dari sistem yang akan dibuat, MongoDB digunakan untuk menyimpan data pengguna dan data jamaah.	cloud.mongodb.com
Node JS Runtime	Runtime Environment untuk menjalankan node js service pada system	-
Express JS	Merupakan framework yang digunakan untuk pengembangan API	v4.16.1
Postman API Client	digunakan dalam pengujian API sistem keamanan aplikasi MultiMasjid yaitu mekanisme otentikasi dan otorisasi dengan JWT serta memastikan proses enkripsi dan dekripsi yang terjadi di database berfungsi dengan benar	v10.23.1

3.6 Desain Sistem Keamanan

Setelah mengumpulkan data yang diperlukan untuk sistem, langkah selanjutnya adalah perancangan desain sistem. Desain sistem bertujuan untuk memberikan gambaran umum tentang sistem yang akan dikembangkan serta memahami alur informasi dan proses yang ada di dalam sistem tersebut. Tahap ini akan menghasilkan desain sistem yang dibutuhkan untuk pengembangan, yang akan terdiri dari desain arsitektur, use case diagram, activity diagram, dan rancangan proses.

3.6.1 Desain Arsitektur Sistem Keamanan

Sistem keamanan yang akan dikembangkan pada aplikasi MutiMasjid ini akan dirancang dengan fokus utama pada keamanan dan integritas data, menggunakan teknologi seperti JSON Web Token (JWT) untuk autentikasi dan otorisasi, enkripsi data menggunakan Advanced Encryption Standard (AES), serta implementasi Cross-Origin Resource Sharing (CORS) untuk membatasi akses API hanya dari domain yang sudah ditentukan.



Gambar 3. 2 Arsitektur Keamanan Data

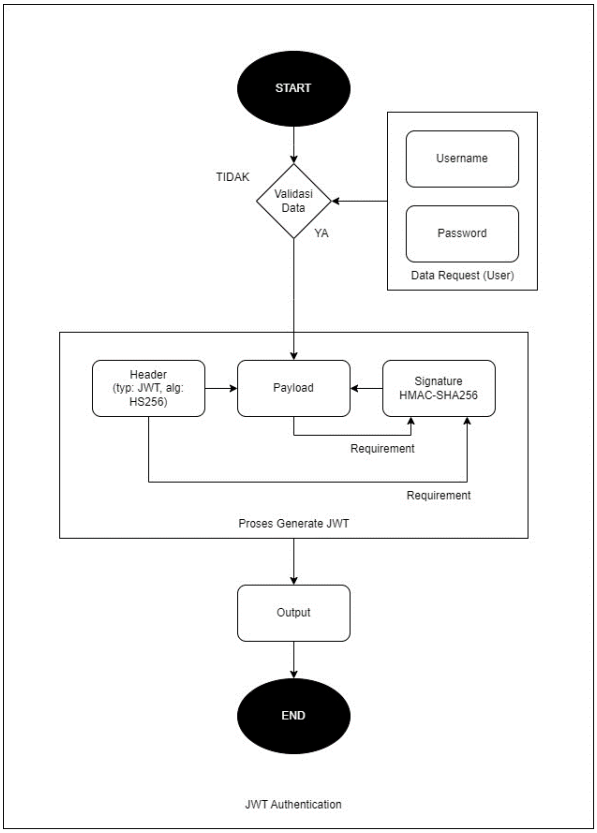
Pada Gambar 3.2 menggambarkan arsitektur keamanan untuk sistem MutiMasjid, yang mengintegrasikan berbagai teknologi untuk melindungi data jamaah. Desain arsitektur keamanan yang diterapkan dalam sistem ini dirancang dengan hati-hati untuk melindungi data pengguna melalui serangkaian mekanisme yang mengintegrasikan tiga aspek utama keamanan informasi, yaitu kerahasiaan (*Confidentiality*), integritas (*Integrity*), dan ketersediaan (*Availability*).

Pada tahap awal, sistem memproses permintaan *login* dari pengguna dengan melakukan validasi *CORS* (Cross-Origin Resource Sharing), yang bertujuan untuk memastikan bahwa permintaan hanya datang dari sumber yang sah dan mencegah akses dari domain yang tidak diotorisasi. Langkah ini merupakan bagian penting dari menjaga kerahasiaan (*Confidentiality*) sistem, memastikan bahwa hanya sumber yang diizinkan yang dapat berinteraksi dengan server. Setelah validasi *CORS*, sistem melakukan autentikasi untuk memverifikasi identitas pengguna, sehingga hanya pengguna dengan kredensial yang sah yang dapat mengakses data atau sumber daya sistem, yang juga mendukung aspek kerahasiaan (*Confidentiality*).

Setiap operasi yang dilakukan dalam sistem, seperti *GET*, *POST*, *PUT*, dan *DELETE*, dilengkapi dengan mekanisme validasi untuk memastikan bahwa data yang diakses, diubah, atau dihapus sesuai dengan kebijakan keamanan yang telah ditetapkan. Validasi ini penting untuk memastikan integritas (*Integrity*) data, menjaga agar data tetap akurat dan konsisten selama proses operasi.

Selain itu, data yang akan disimpan di basis data MongoDB Cloud dienkripsi menggunakan algoritma AES-256-CBC. Enkripsi ini tidak hanya memastikan kerahasiaan (*Confidentiality*) data yang disimpan, tetapi juga mendukung integritas (*Integrity*) data dengan melindunginya dari modifikasi yang tidak sah. Ketika data yang dienkripsi diperlukan kembali, sistem akan mendekripsi data tersebut dengan menggunakan kunci yang sesuai, memastikan bahwa data yang diambil kembali adalah data asli dan belum dirusak.

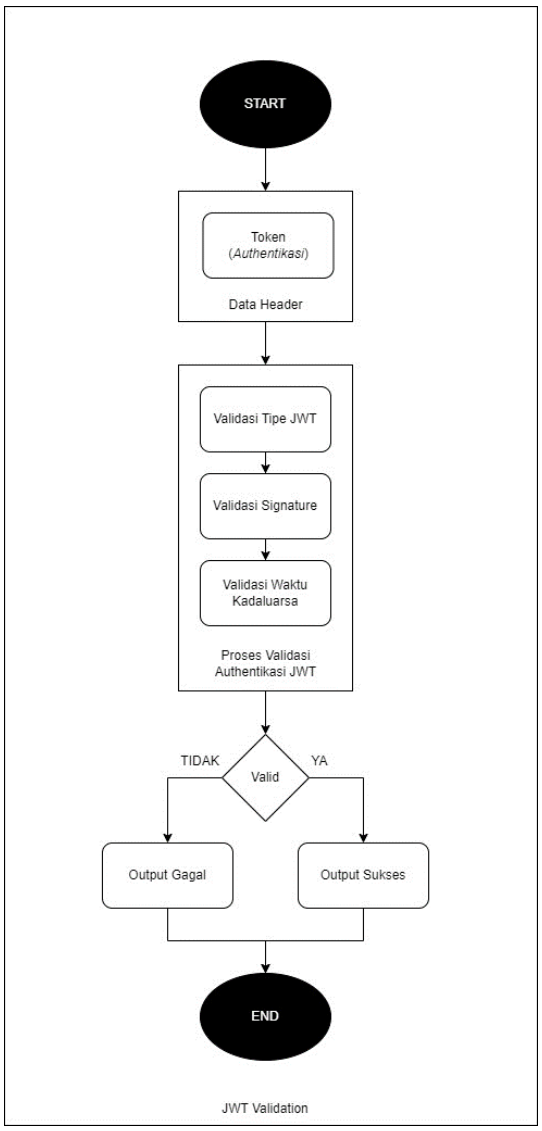
Seluruh proses ini dirancang untuk memastikan bahwa sistem tidak hanya menjaga kerahasiaan dan integritas data, tetapi juga memastikan ketersediaan (*Availability*) informasi yang valid dan responsif kepada pengguna yang sah. Dengan demikian, desain arsitektur keamanan ini memberikan perlindungan menyeluruh terhadap data pengguna, memastikan bahwa data dilindungi dari akses yang tidak sah, modifikasi yang tidak diinginkan, serta memastikan bahwa informasi selalu tersedia ketika dibutuhkan.



Gambar 3. 3 JWT Authentication

Pada Gambar 3.3 menunjukkan proses autentikasi token JWT pada sistem yang memerlukan dua sumber data: username dan password. Jika data memenuhi kriteria yang diminta, maka akan divalidasi. Setelah itu, akan dienkripsi menggunakan JWT dengan Algoritma Type HS256 dan tanda tangan SHA-256, dan token string yang berawalan "ey" akan dikeluarkan dari data output.

Setelah token JWT dienkripsi, token tersebut akan divalidasi sebelum akses atau memulai proses lainnya, seperti yang ditunjukkan pada Gambar 3.3.



Gambar 3. 4 JWT Validation

Dalam proses ini, validasi JWT dengan awalan token string "ey" akan divalidasi melalui beberapa proses, seperti

a. Tipe Token

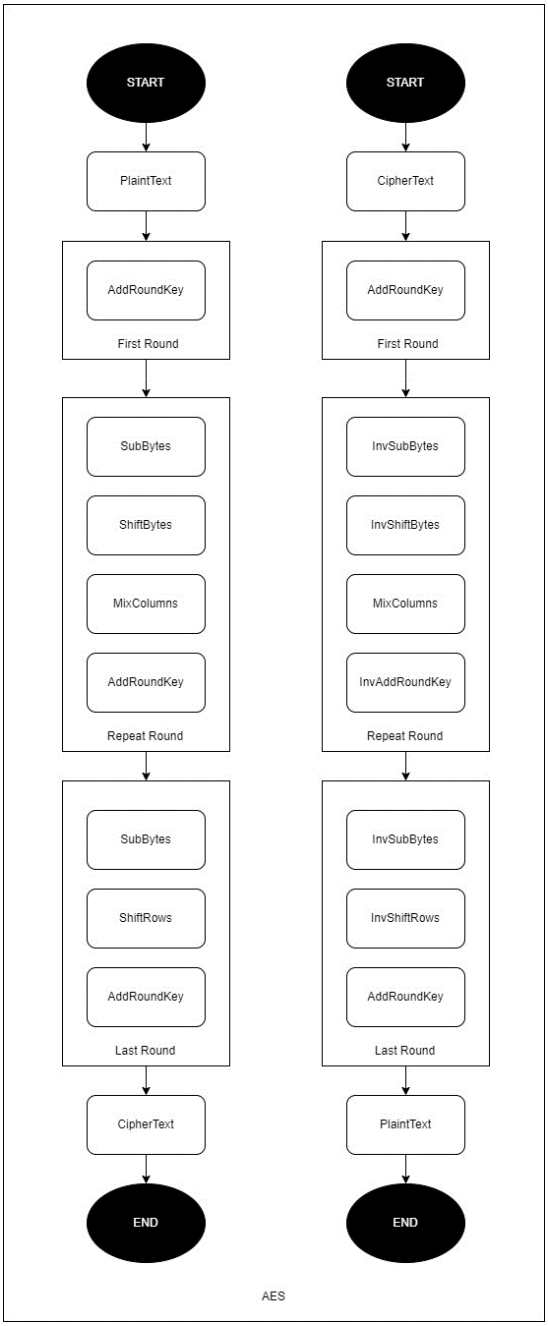
Memverifikasi format token JWT termasuk memastikan bahwa itu terdiri dari tiga bagian yang terpisah oleh titik (.) dan bahwa setiap bagian adalah string yang terenkripsi dengan Base64.

b. Tanda tangan (*signature*)

Untuk memastikan bahwa token tidak diubah dan berasal dari sumber yang sah, pastikan tanda tangan digital padanya sepadan dengan tanda tangan yang diharapkan dengan menggunakan kunci rahasia yang sama yang digunakan untuk membuat tanda tangan.

c. Validasi waktu kadaluarsa

Memeriksa masa berlaku (*expiration*), peran pengguna, atau klaim lainnya yang relevan sesuai dengan kebutuhan aplikasi dapat termasuk dalam kategori ini. Jika semuanya sudah tervalidasi dan lulus, proses berikutnya akan dimulai, terlepas dari apakah outputnya berhasil atau gagal.



Gambar 3. 5 Proses Enkripsi Dekripsi AES

Selanjutnya, pada Gambar 3.5 menunjukan proses peneliti ketika melakukan proses Enkripsi dan Dekripsi dengan AES.

a. Proses Enkripsi AES:

Proses enkripsi AES terdiri dari tiga tahap, biasanya disebut sebagai putaran pertama, putaran berulang, dan putaran terakhir. Setiap putaran terdiri dari sejumlah operasi yang bertujuan untuk mengubah data secara bertahap.

1. Putaran Pertama, AddRoundKey: Untuk menghasilkan output awal, kunci rahasia (kunci enkripsi) akan di-"XOR" dengan blok teks biasa (plaintext) pada putaran pertama.
2. Putaran Berulang (Putaran Berulang):

   SubBytes

   Nilai dari tabel substitusi non-linear, juga dikenal sebagai S-box, akan digunakan untuk mengganti setiap byte dalam blok data dengan nilai yang sesuai dengan byte tersebut.

   ShiftRow

   Menggeser baris ke kiri pada blok matriks akan mengubah data blok.

   MixColumns

   Kombinasi Kolom: Ini adalah operasi yang melibatkan kolom-kolom dalam blok yang mengalikan matriks dan menghasilkan transformasi linier.
3. Putaran Terakhir

   SubBytes

   Seperti pada putaran sebelumnya, byte dapat diganti dengan S-box.

   ShiftRows

   Kumpulan baris dalam blok matriks.

   AddRoundKey

   Adakan kunci putaran terakhir untuk memasukkan blok terakhir ke dalam "XOR" dengan kunci putaran terakhir.

b. Proses Dekripsi AES

Dekripsi AES biasanya menggunakan kebalikan enkripsi. Untuk mengembalikan teks terenkripsi ke bentuk aslinya, atau plaintext, proses dekripsi menggunakan prosedur yang berlawanan dengan prosedur enkripsi. Ini terdiri dari tiga tahap utama: putaran pertama (*first round*), putaran berulang (*repeat rounds*), dan putaran terakhir.

1. Putaran Pertama. AddRoundKey: untuk menghasilkan output awal pada putaran pertama dekripsi, kunci rahasia atau kunci dekripsi akan di-“XOR” dengan blok teks terenkripsi pada putaran pertama.

2. Putaran Berulang

   Langkah-langkah ini dilakukan secara terbalik dari proses enkripsi.

   a. InvShiftRows: Kembalikan pergeseran baris yang dilakukan pada putaran enkripsi.
   b. InvSubBytes: Lakukan invers substitusi byte menggunakan tabel invers S-box.
   c. InvMixColumns: Operasi invers pada kolom-kolom dari blok, menggunakan matriks invers untuk mengembalikan transformasi linier.

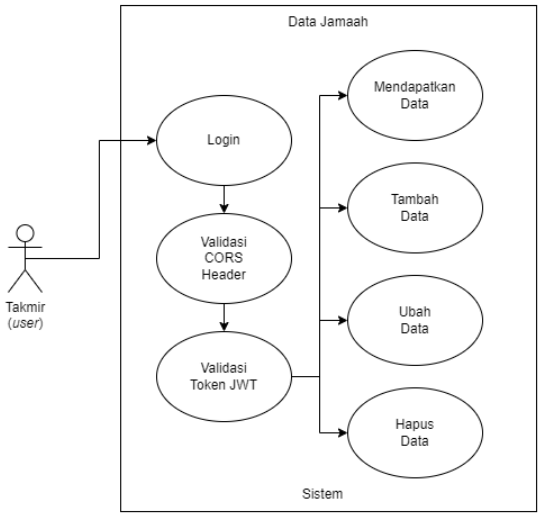
3. Putaran Terakhir

   AddRoundKey: Putaran terakhir di mana blok teks terenkripsi di-"XOR" kan dengan kunci putaran terakhir.

   Blok teks terenkripsi akan kembali ke blok teks asli (plaintext) setelah putaran terakhir selesai. Untuk menghasilkan hasil yang tepat, dekripsi AES memerlukan prosedur yang berlawanan dengan enkripsi untuk mengembalikan teks terenkripsi ke bentuk aslinya.

3.6.2 Use Case Diagram

Gambar 3.6 menunjukan *use case* diagram yang menjelaskan bagaimana pengguna sistem berinteraksi dengan sistem keamanan yang akan dibuat



Gambar 3. 6 Use Case Diagram

Proses penggunaan sistem ditunjukkan pada Gambar 3.6 Use Case Diagram, Pengguna (*User: Takmir*) melakukan login dan sistem melakukan validasi dari CORS Header dan Token JWT. Jika diterima oleh sistem, Pengguna dapat melakukan fungsi seperti mendapatkan data, menambah data, mengubah data, dan menghapus data. Fungsi ini telah dienkripsi dengan AES.

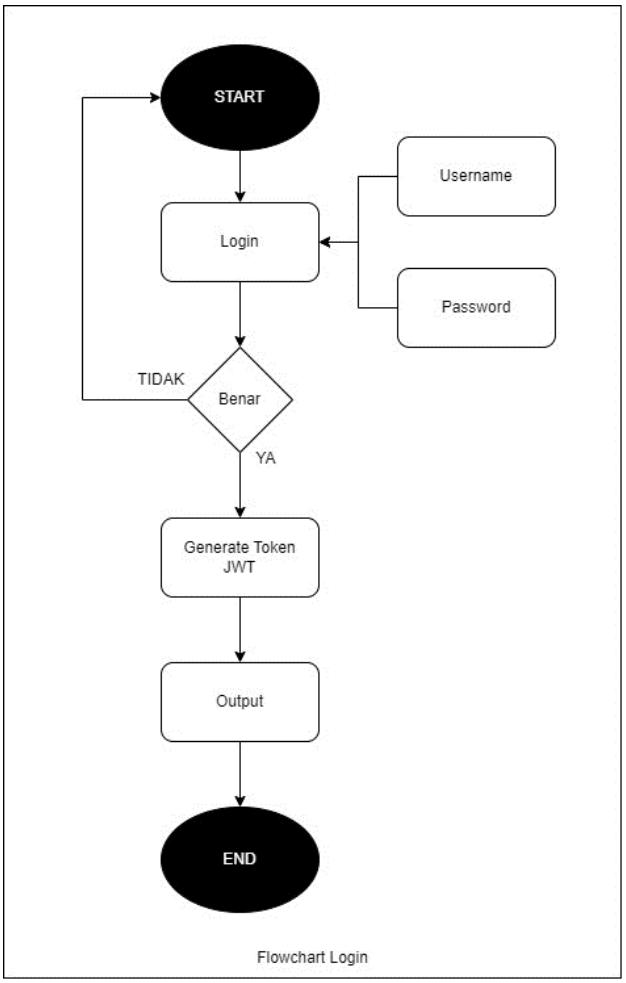
No	*Use Case*	Keterangan
1	Login	Proses otorisasi bagi User (Takmir) agar dapat memasuki halaman sistem.
2	Validasi CORS Header	Proses validasi pada bagian Server dengan data Authentication JWT, Origin, Content-Type, Method (GET, CREATE, PUT, DELETE)
3.	Validasi Token JWT	Proses validasi dengan otoriasi JWT Token pada bagian Header: Authentication
4.	Mendapatkan Data	User (*Takmir*) dapat melakukan tindakan mengambil semua data yang tersedia pada database.
5.	Tambah Data	User (*Takmir*) dapat melakukan tambah data pada database Jamaah.
6.	Ubah Data	User (*Takmir*) dapat melakukan tindakan perubahan data pada database.
7.	Hapus Data	User (*Takmir*) dapat melakukan hapus data pada database.

Table 3.1 Penjelasan Use Case Diagram

3.6.3 Flowchart Proses Sistem

Flowchart merupakan representasi grafis dari langkah-langkah dan urutan prosedur yang ada dalam suatu sistem. Flowchart menggambarkan alur logika dari proses bisnis yang sedang dirancang secara visual dan terstruktur. Berikut penjelasan mengenai flowchart yang telah berhasil dirancang.

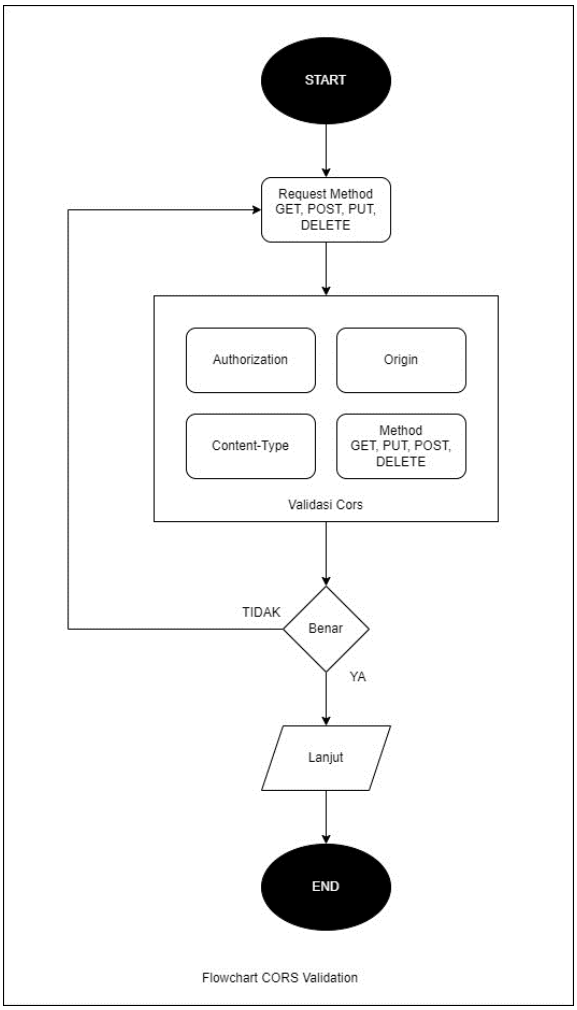
a. Flowchart Login



Gambar 3. 7 Flowchart Login

Berdasarkan Gambar 3.7 di atas, proses dimulai ketika pengguna ingin login ke dalam sistem. Pengguna diminta untuk memasukkan identitas mereka, yaitu username dan password yang telah mereka daftarkan sebelumnya. Sistem kemudian memeriksa apakah kombinasi username dan password yang dimasukkan sudah benar dengan mencocokkan data di dalam database. Jika kombinasi tersebut benar, sistem akan menghasilkan token JWT (JSON Web Token) sebagai tanda pengguna telah terautentikasi. Token ini dikirimkan kepada pengguna, memungkinkan mereka untuk melanjutkan ke proses atau layanan berikutnya dalam sistem.

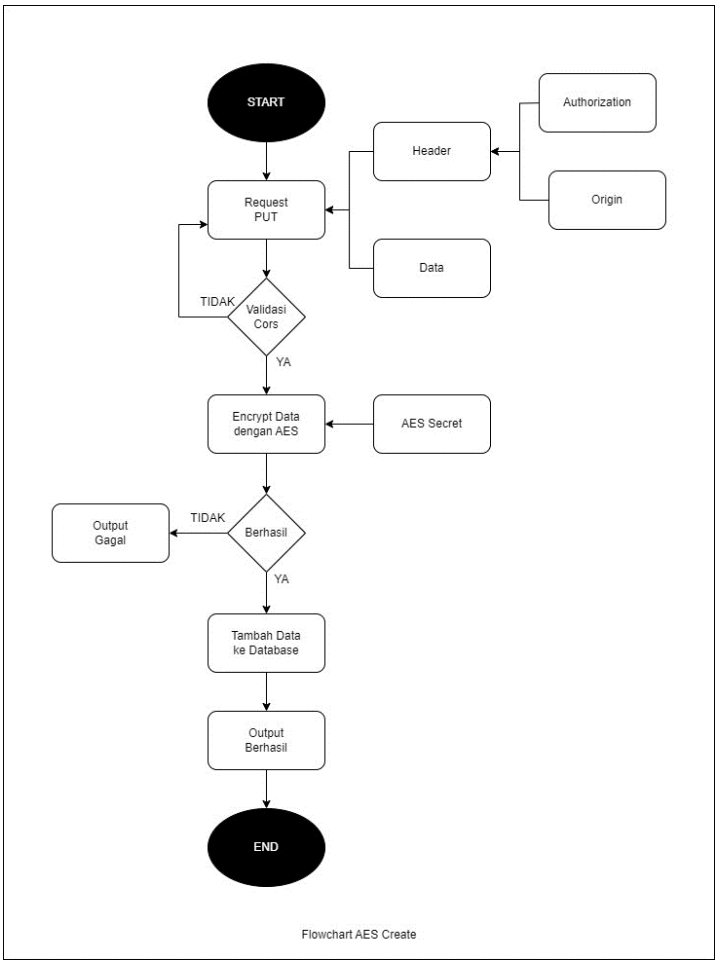
1. Flowchart Validasi CORS



Gambar 3. 8 Flowchart CORS Validasi

Gambar 3.8 menunjukkan proses validasi Cross-Origin Resource Sharing (CORS), sebuah mekanisme keamanan untuk mengontrol akses ke sumber daya pada server dari domain lain. Proses ini dimulai ketika sebuah request API diterima oleh server. Pertama, server memeriksa metode request yang digunakan, seperti GET, POST, PUT, atau DELETE. Kemudian, server memeriksa header request untuk autentikasi, yang memastikan bahwa request tersebut dikirim oleh pengguna yang terotentikasi. Selanjutnya, server memeriksa asal atau *origin* dari request untuk mengonfirmasi bahwa domain yang melakukan request telah diizinkan. Server juga memeriksa *Content-Type* untuk menentukan jenis konten yang diminta. Setelah semua berhasil divalidasi, pengguna dapat melanjutkan proses selanjutnya.

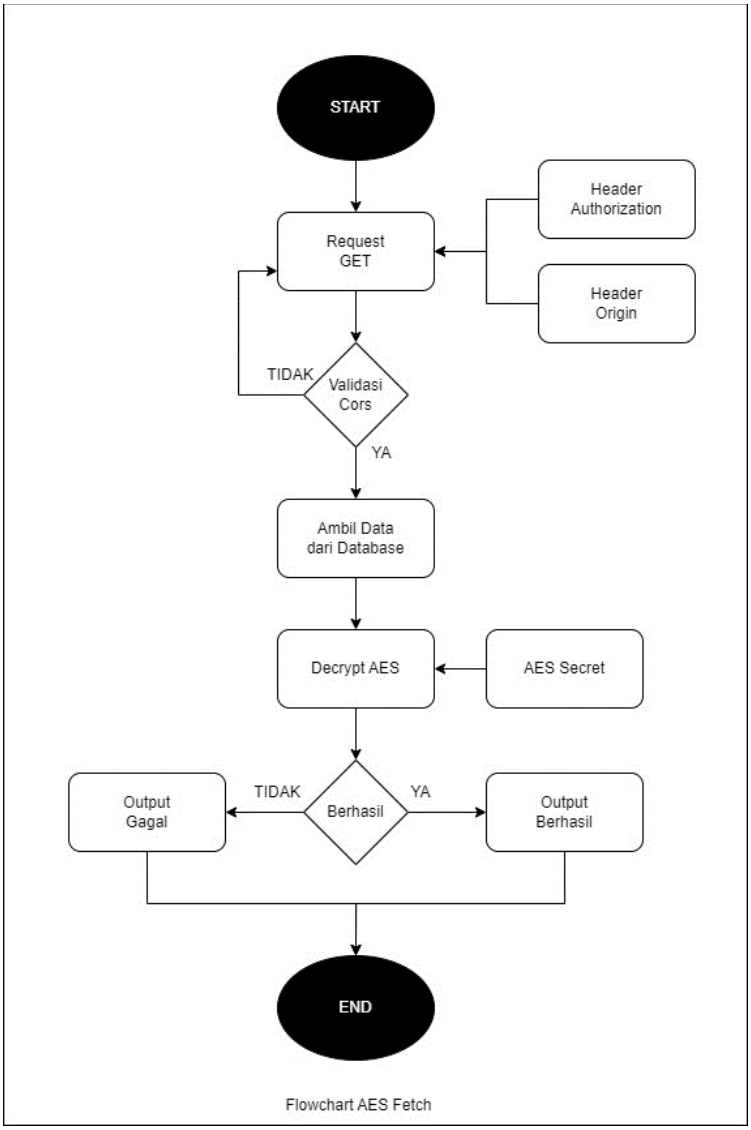
2. Flowchart Data Jamaah (*Create)*



Gambar 3. 9 Flowchart AES Create

Gambar 3.9 menunjukkan cara menambahkan data jamaah yang kemudian disimpan ke dalam database dengan enkripsi menggunakan kriptografi AES. Dijelaskan bahwa proses ini akan disimpan setelah menerima permintaan data atau permintaan pengguna yang melalui tahap validasi berupa *authorzation* token JWT yang didapat dari proses *login* pertama kali serta memeriksa *origin* dari request. Proses ini dianggap berhasil jika data tersedia dan proses enkripsi telah berhasil. Output dari proses ini dapat berupa string JSON yang siap diolah atau output dari data enkripsi yang telah disimpan.

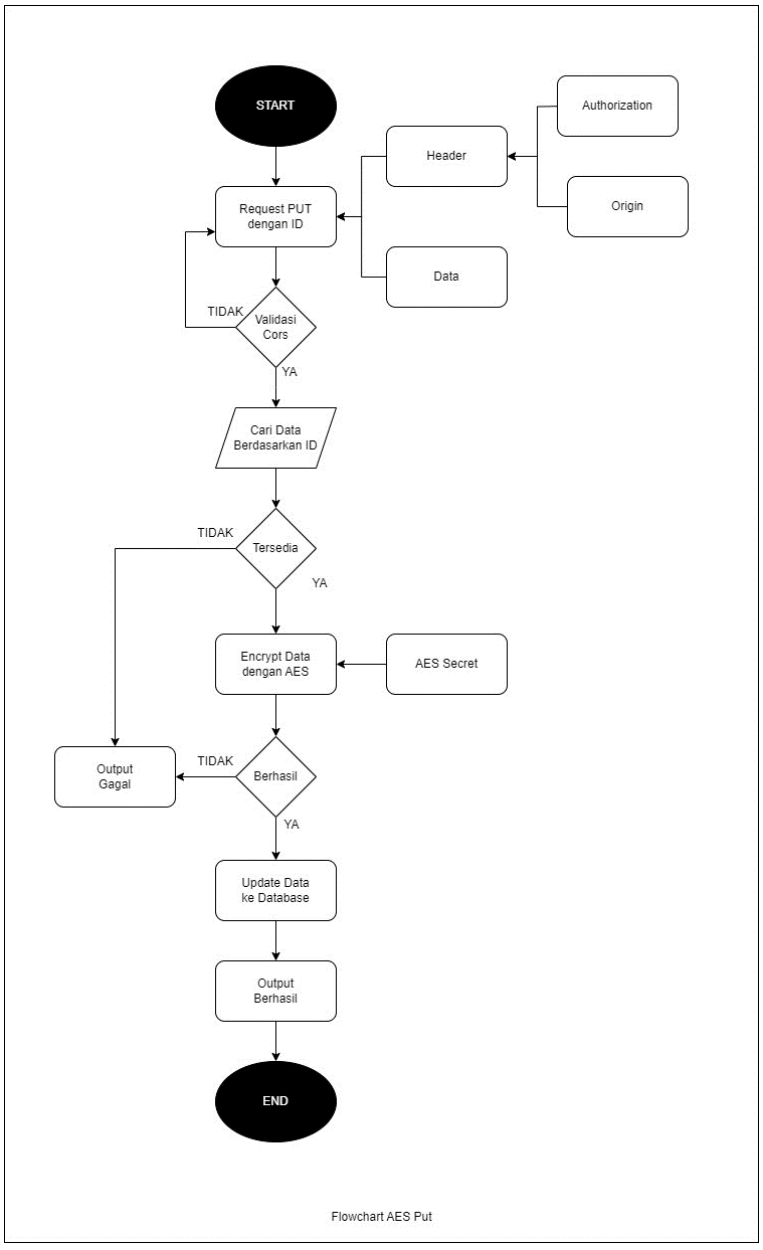
3. Flowchart Data Jamaah (*Fetch*)



Gambar 3. 10 Flowchart AES Fetch

Gambar 3.10 Flowchart AES Fetch adalah proses mengambil seluruh data dari database. Dijelaskan bahwa proses ini akan menampilkan data jamaah setelah menerima permintaan data atau permintaan pengguna. Sistem akan memeriksa apakah pengguna memiliki otorisasi melalui tahap validasi berupa *authorzation* token JWT yang didapat dari proses *login* pertama kali serta memeriksa *origin* dari request.

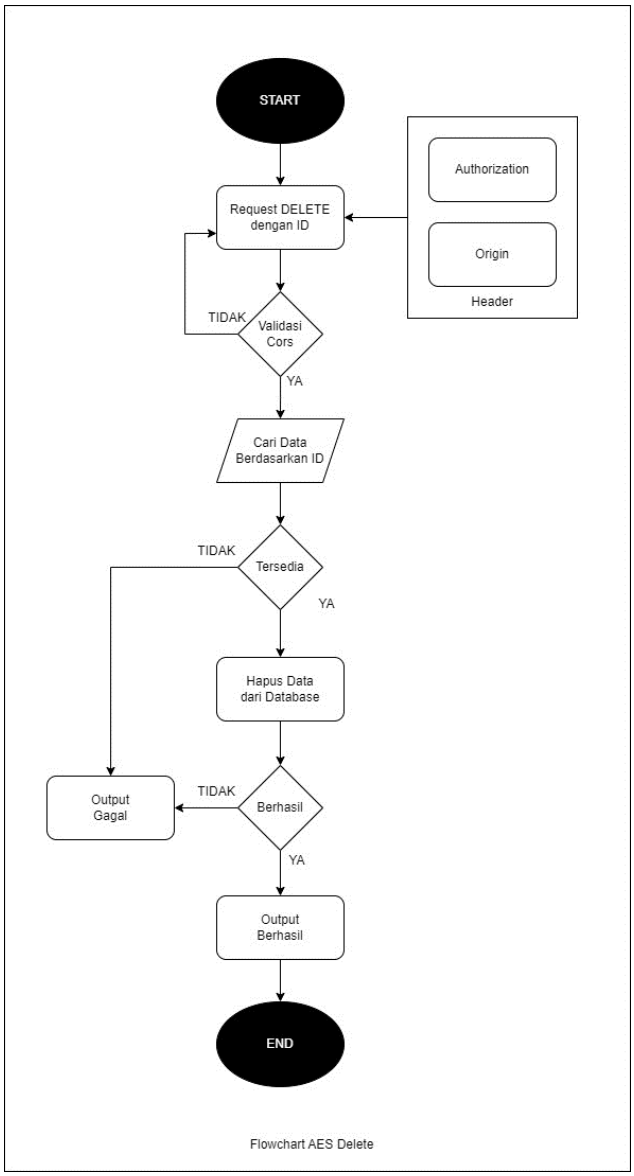
4. Flowchart Data Jamaah (*Put*)



Gambar 3. 11 Flowchart AES Put

Selain menambah dan menampilkan data jamaah, Gambar 3.11 Flowchart AES Put menunjukkan proses perubahan data jamaah. Proses ini membutuhkan sebuah ID untuk mengidentifikasi data yang akan diubah, setelah berhasil diubah data akan di enkripsi dengan kriprotografi AES, dan outputnya adalah JSON.

5. Flowchart Data Jamaah (*Delete)*

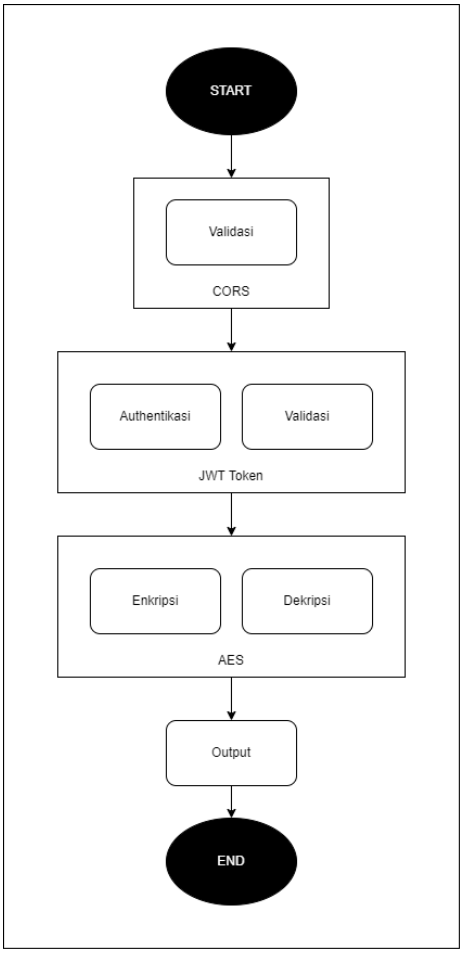


Gambar 3. 12 Flowchart AES Delete

Gambar 3.12 Flowchart AES Delete menunjukkan proses hapus tanpa AES karena fungsi pencarian data di database yang akan dihapus hanya menggunakan validasi dengan ID dan validasi JWT dan CORS.

3.7 Tahapan Pengujian Sistem

Rencana pengujian akan dilakukan dengan 3 tahap, yaitu tahap pengujian CORS, tahap pengujian JWT, dan yang terakhir tahap pengujian AES, yang mana ditunjukan pada Gambar 3.12



Gambar 3. 13 Tahapan pengujian

a. Tahap Pertama: Pengujian CORS

Pengujian CORS sangat penting untuk memastikan bahwa aplikasi dapat dengan aman berinteraksi dengan sumber daya dari berbagai domain. Dalam skenario ini, penguji akan menguji CORS dengan dua sistem tahapan.

1. Pengujian *Requrest Metode*: Memastikan bahwa permintaan GET, POST, PUT, dan DELETE, hanya dapat dilakukan oleh domain yang sudah diatur di dalam sistem.
   Langkah:
   1) Mengirim permintaan *GET, POST, PUT,* dan *DELETE* dari domain yang diizinkan dan yang tidak diizinkan.

2) Memastikan apakah server menerima permintaan dari domain yang diizinkan dan menolak permintaan dari domain yang tidak diizinkan.
3) Mengamati respon server untuk memastikan bahwa permintaan dari domain yang tidak diizinkan mendapatkan respon yang sesuai (HTTP 403 Forbidden atau HTTP 400 Bad Request).

2. Pengujian *Header* Request: Melakukan verifikasi apakah header yang diperlukan seperti Origin ada dalam permintaan sumber daya lintas domain.

   Langkah:

   1) Mengirim permintaan dari berbagai domain dengan header Origin yang berbeda.
   2) Memverifikasi apakah server merespons dengan benar hanya ketika header Origin ada dan valid.
   3) Pastikan bahwa permintaan tanpa header Origin yang sesuai atau dengan header Origin yang tidak valid mendapatkan respons error yang sesuai (seperti HTTP 400 Bad Request).

b. Tahap Kedua: Pengujian JWT Token

Pengujian ini bertujuan untuk memastikan bahwa sistem otentikasi yang menggunakan token JWT (JSON Web Token) berfungsi dengan baik dan aman. Pengujian ini mencakup dua aspek utama: otentikasi dan validasi token.

1. Pengujian Otentikasi Token JWT
   1) Uji Otentikasi Berhasil: Memastikan bahwa token JWT dibuat dengan benar dan dikirim ke klien setelah pengguna berhasil login.
   2) Uji Informasi dalam Token: Memastikan bahwa payload token JWT berisi informasi yang benar seperti klaim, tanggal kadaluarsa, dan identitas pengguna.
2. Pengujian Validasi Token JWT
   1) Uji Validasi Struktur: Memastikan bahwa struktur token JWT benar. Terdiri dari tiga bagian (header, payload, signature) yang dipisahkan oleh titik.
   2) Uji Validasi Klaim: Memeriksa apakah klaim dalam token benar dan sesuai dengan yang diharapkan.

c. Tahap Ketiga: Pengujian AES

Pengujian enkripsi dan dekripsi AES dilakukan dengan tujuan memastikan bahwa algoritma enkripsi AES beroperasi dengan benar saat mendekripsi dan mengamankan data. Berikut adalah prosedur yang digunakan penguji untuk menguji enkripsi dan dekripsi AES:

1. Uji Enkripsi Data: Memastikan bahwa data dapat dienkripsi dengan benar menggunakan kunci tertentu.
2. Uji Dekripsi Data: Mendekripsi data yang telah dienkripsi sebelumnya dengan menggunakan kunci yang benar, memastikan bahwa data yang ter-dekripsi identik dengan data asli dan memastikan bahwa tidak ada data yang ter-dekripsi dengan benar setelah mendekripsi data dengan menggunakan kunci yang salah.\

BAB IV
HASIL DAN PEMBAHASAN

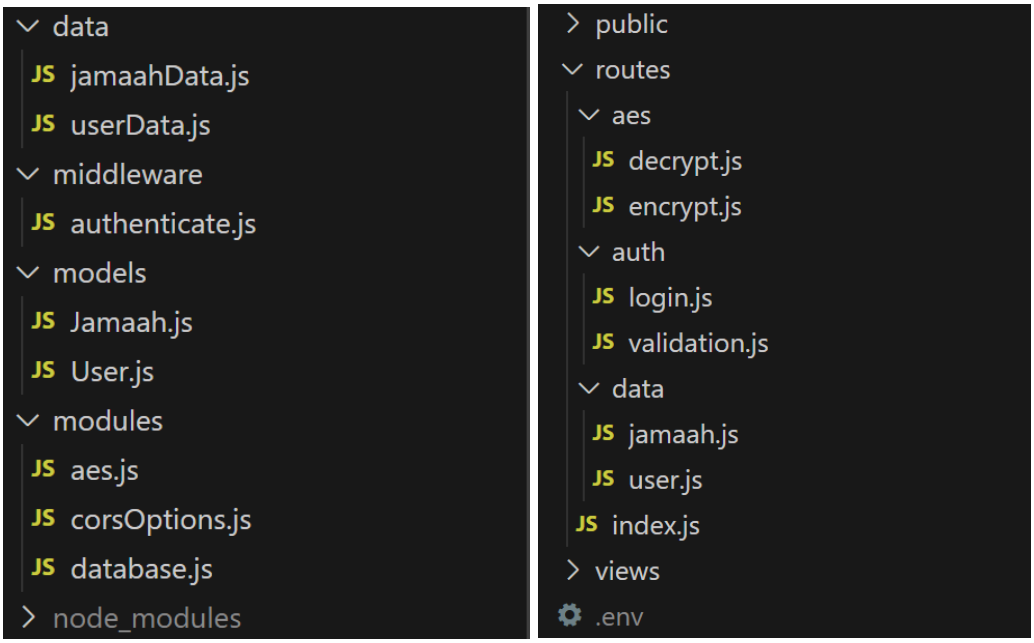
4.1 Pendahuluan

Bab ini akan membahas penerapan sistem keamanan data dengan mengimplementasikan JSON Web Token sebagai keamanan autentikasi dan otorisasi, CORS sebagai mekanisme pemberian izin akses ke sumber daya, serta enkripsi database untuk menjaga keamanan dan privasi pengguna. Pengembangan sistem keamanan ini mencakup pembuatan API endpoint untuk registrasi dan login pengguna yang menyimpan data pengguna secara aman di database. Kemudian dilakukan generasi dan validasi token JSON Web Token untuk autentikasi setiap permintaan ke server. Selain itu, akan dibangun sistem otorisasi berlapis untuk mengontrol akses pengguna ke fitur dan data aplikasi.

Implementasi sistem ini akan dijelaskan langkah demi langkah disertai penjelasan teknis dan contoh kode implementasi akan diberikan untuk mendukung setiap langkah. Dengan demikian, diharapkan dapat memberikan pemahaman mendalam mengenai cara membangun sistem keamanan yang baik. Hal ini juga akan memberikan pemahaman yang kuat tentang praktik terbaik dan masalah keamanan yang perlu diperhatikan selama proses ini.

4.2 Perancangan Program Sistem Keamanan

Pada tahap awal dalam penelitian ini, dilakukan perancangan program dengan menerapkan layanan API untuk berinteraksi antara klien dan server. API yang digunakan ini akan dijalankan dalam *local environment* di sistem operasi berbasis windows dengan menggunakan *Node Package Manager* (NPM) yang digunakan sebagai package manager atau dependencies management. NPM akan membaca daftar dependensi yang tercantum pada file package.json, mengunduh paket-paket yang dibutuhkan, dan memasangnya di dalam direktori node_modules. Keseluruhan struktur dari projek yang sedang dibangun dapat dilihat pada Gambar 4.1



Gambar 4. 1 Struktur Program Sistem Keamanan

Dapat dilihat pada Gambar 4.1 di atas, terdapat beberapa direktori yang di dalamnya berisi file kode program yang masing masing memiliki fungsi yang berbeda beda.

4.2.1 Direktori Data

Folder data berisikan dua file kode program yaitu jamaahData.js dan userData.js yang digunakan untuk melakukan migrasi dan menyimpan data ke dalam basis data MongoDB.

a. jamaahData.js

Kode program tersebut berfungsi untuk memasukkan data jamaah ke dalam tabel jamaah di dalam basis data mongoDB. Data jamaah tersebut berupa nama, alamat, nomor telepon, dan nomor identitas.

b. userData.js

Kode program user data digunakan untuk membuat dan mengimport data pengguna atau client ke dalam basis data mongoDB. Informasi pengguna tersebut berupa username dan password yang terkait dengan sistem yang sedang dikembangkan.

4.2.2 Middleware

Pada middleware terdapat file program authenticate.js yang berfungsi sebagai autentikasi untuk melindungi endpoint pada sistem yang menerapkan penggunaan JSON Web Token (JWT). Dengan middleware ini, setiap rute atau endpoint yang dilindungi harus menyertakan token JWT yang valid dalam header otorisasi permintaan HTTP. Middleware akan memverifikasi token tersebut sebelum mengizinkan akses ke rute atau endpoint yang dilindungi. Jika token valid, permintaan akan diizinkan untuk melanjutkan ke fungsi berikutnya

dalam siklus permintaan respon aplikasi. Namun, jika token tidak ada, kedaluwarsa, atau tidak valid, middleware akan mengembalikan respon dengan status error yang sesuai, seperti "Token expired", "Forbidden", atau "Unauthorized". Dengan begitu middleware membantu melindungi sumber daya aplikasi dari akses tidak sah dan memastikan hanya pengguna yang terotentikasi yang dapat mengaksesnya. Secara lebih rinci, middleware mengikuti langkah-langkah sebagai berikut:

4.2.3 Models

File Models berfungsi sebagai *blueprint* atau pola untuk membantu dalam mengorganisir dan mendefinisik.an struktur data untuk setiap koleksi dalam basis data MongoDB. Dalam direktori models terdapat 2 file kode program, yaitu Jamaah.js dan User.js. Model ini digunakan dalam aplikasi untuk berinteraksi dengan koleksi jamaah dan user dalam basis data, seperti melakukan operasi CRUD pada data jamaah dan pengguna.

4.2.4 Modules

Dalam file modules terdapat 3 file program yaitu aes.js, corsOptional.js, dan database.js. ketiga file tersebut memiliki fungsi masing masing diantaranya:

a. Modules aes.js

Modul aes merupakan komponen yang berfungsi untuk mengenkripsi dan mendekripsi data. Penulis skripsi memperkenalkan modul ini yang dibuat dalam bahasa JavaScript. Modul ini menggunakan algoritma AES, yang merupakan salah satu algoritma enkripsi simetris yang paling umum digunakan untuk melindungi data sensitif. File program tersebut berisi dua fungsi utama yaitu *encrypt* dan *decrypt*.

3. Fungsi encrypt berfungsi untuk melakukan enkripsi data atau pesan menggunakan algoritma AES. Fungsi ini menerima parameter pesan, yang merupakan teks yang akan dienkripsi, dan kemudian menggunakan vector initialization (IV) dan kunci enkripsi yang dibuat secara acak. Hasil enkripsi dan IV dikembalikan sebagai objek.
4. Fungsi decrypt yang bertanggung jawab untuk mendekripsi data yang telah dienkripsi sebelumnya. Fungsi ini menerima parameter pesan, yang merupakan data yang akan didekripsi, dan vektor awal, yang digunakan selama proses enkripsi. Kunci enkripsi dan IV yang diberikan digunakan untuk mendekripsi data dan mengembalikan teks asli yang telah di berfungsi dekripsi sebelumnya.

Kedua fungsi tersebut memanfaatkan modul *crypto* bawaan Node.js untuk melakukan operasi enkripsi dan dekripsi dengan algoritma AES. Kunci enkripsi dan algoritma yang digunakan diambil dari variabel yang diatur dengan menggunakan modul dotenv.

b. Modules Cors Optional

Modul ini merupakan bagian dari implementasi keamanan sistem yang diteliti dalam naskah skripsi. File corsOptions.js ini berisi konfigurasi untuk mengatur Cross-Origin Resource Sharing (CORS) pada sistem yang berfungsi membantu mengatur akses antara sumber daya di suatu domain ke sumber daya di domain lain. Dengan konfigurasi CORS, server akan mengizinkan permintaan CORS dari domain https://example.com dan https://google.com dengan metode HTTP GET, PUT, POST, dan DELETE, serta header Content-Type dan Authorization. Jika permintaan CORS berasal dari domain lain, maka akan ditolak dengan pesan "Failure: Access denied."

c. Modules Database

Dalam aplikasi yang sedang diteliti, modul ini mengelola koneksi dan konfigurasi ke basis data MongoDB. Basis data ini digunakan untuk menyimpan dan mengelola data yang diperlukan oleh aplikasi. Modul ini menggunakan modul mongoose untuk menghubungkan ke basis data MongoDB yang dihosting di MongoDB Atlas, yang juga dikenal sebagai cloud MongoDB. Dengan menggabungkan informasi dari variabel lingkungan, modul ini membuat URL koneksi MongoDB. URL ini kemudian digunakan dalam fungsi connect untuk melakukan koneksi ke basis data MongoDB. Koneksi ini diperlukan agar sistem dapat melakukan operasi CRUD (Create, Read, Update, Delete) pada basis data MongoDB.

4.2.5 Routes

Dalam pengembangan modul ini, Node.js menyediakan fungsi routing untuk untuk mengatur dan mengelola rute yang tersedia dalam sistem yang sedang dikembangkan. Folder routes memungkinkan pemisahan kode yang bertanggung jawab untuk menangani rute-rute dari logika utama sistem yang berfungsi agar lebih terstruktur dan memudahkan pemeliharaan kode. Pada sistem keamanan yang sedang dikembangan oleh penulis, terdapat 3 sub-folder yang berisi 8 file program yang mengatur rute yang ada pada sistem.

a. Routes AES – Encrypt

Kegunaan utama dari file encrypt.js adalah untuk menyediakan layanan enkripsi data menggunakan algoritma AES yang kuat. Pengguna dapat mengirimkan pesan teks melalui parameter URL, dan file ini akan mengenkripsi pesan tersebut dengan kunci dan vektor inisialisasi yang aman. Data terenkripsi dan vektor inisialisasi kemudian disimpan dalam basis data untuk digunakan dalam proses dekripsi jika dibutuhkan.

b. Routes AES - Decrypt

File decrypt.js berfungsi untuk memberikan layanan dekripsi bagi data yang telah dienkripsi dengan algoritma AES. Pengguna dapat mengirimkan data terenkripsi melalui parameter URL, dan file ini akan mencari data tersebut dalam basis data. Jika data ditemukan, file ini akan menggunakan kunci dekripsi dan vektor inisialisasi yang tersimpan untuk mendekripsi data tersebut dan mengembalikan data asli dalam bentuk teks.

c. Routes Auth - Login

Fungsi utama file login.js adalah bertindak sebagai gerbang otentikasi untuk sistem keamanan yang sedang dikembangkan, di mana pengguna harus memasukkan kredensial yang valid untuk mendapatkan token JWT yang nantinya akan digunakan untuk mengakses sumber daya atau melakukan operasi yang memerlukan otentikasi. Dengan mekanisme autentikasi, sistem dapat memastikan bahwa hanya pengguna yang berwenang dan memiliki kredensial yang valid yang dapat mengakses sumber daya yang dilindungi.

d. Routes Auth – Validation

File kode program validation.js berfungsi sebagai middleware untuk memverifikasi dan memvalidasi token autentikasi JSON Web Token (JWT) yang dikirimkan dalam permintaan oleh pengguna. Tujuan utamanya adalah untuk memastikan bahwa hanya pengguna yang telah diautentikasi dengan benar yang dapat mengakses sumber daya atau rute yang dilindungi dalam sistem keamanan yang sedang dikembangkan.

e. Routes Data – Jamaah

File program jamaah.js merupakan sebuah rute yang digunakan untuk mengelola data jamaah, termasuk mendapatkan daftar jamaah yang tersimpan dalam basis data (GET /), membuat data Jamaah baru dengan enkripsi AES (POST /), pembaruan data berdasarkan ID dengan enkripsi (PUT /:id), dan penghapusan data Jamaah berdasarkan ID (DELETE /:id). Enkripsi dilakukan pada data jamaah untuk menjaga keamanan dan kerahasiaan informasi yang disimpan dalam basis data. Dalam setiap rute, file menggunakan middleware *cors Options* untuk mengizinkan akses lintas sumber sesuai dengan konfigurasi. File juga menggunakan middleware authenticate JWT untuk memastikan bahwa hanya pengguna yang telah diautentikasi yang dapat mengakses rute-rute tersebut.

f. Routes Data – User

File program ini menyediakan fungsionalitas untuk mengelola data User, termasuk mengambil semua data User yang telah didekripsi dari database (GET /), membuat data User baru dengan enkripsi AES (POST /), memperbarui data berdasarkan ID setelah dilakukan enkripsi (PUT /:id), dan menghapus data User berdasarkan ID (DELETE /:id).

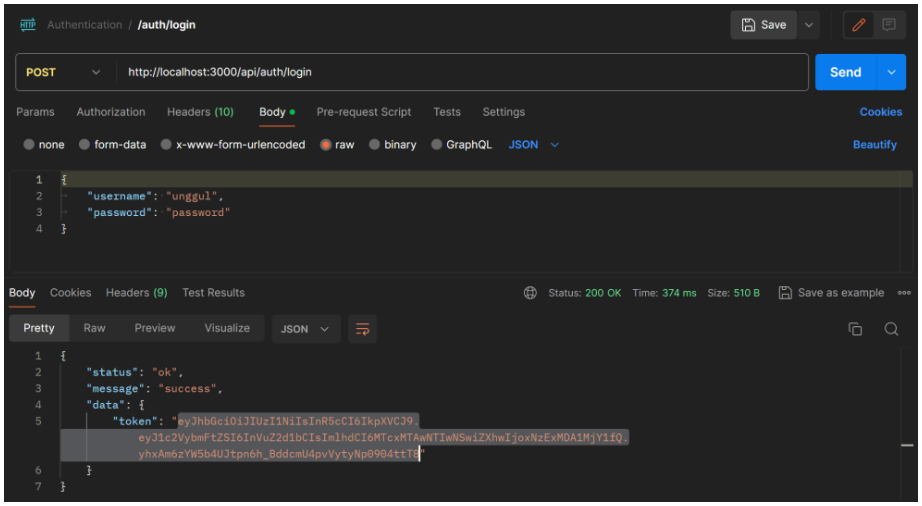
Program ini dilengkapi dengan middleware authenticate JWT yang memastikan otentikasi pengguna sebelum mengizinkan akses ke fungsi pengelolaan data user. Data User disimpan dalam bentuk terenkripsi di database menggunakan AES (Advanced Encryption Standard) dengan library bcrypt untuk mengenkripsi password. Selain itu, middleware cors digunakan untuk mengelola permintaan Cross-Origin Resource Sharing (CORS), memungkinkan interaksi yang aman antara domain yang berbeda.

4.3 Pengujian

Pada penelitian ini pengujian akan memanfaatkan aplikasi Postman sebagai client untuk menguji API sistem keamanan yang telah dibuat. Terdapat empat metode yang digunakan dalam proses pengujian, yaitu metode POST untuk mengirim data, GET untuk mengambil data yang ada, PUT untuk memperbarui data, dan DELETE untuk menghapus data. Dalam setiap kali penggunaan metode tersebut, pengguna harus menyertakan token yang diperoleh setelah berhasil melakukan login dengan metode POST dan memasukkan username dan password yang benar pada aplikasi Postman. Token tersebut berfungsi sebagai kunci akses untuk melakukan permintaan berikutnya. Selain itu, pengguna diwajibkan untuk menyertakan CORS-Origin dengan *value* "https://example.com" pada *header* saat melakukan seluruh request. Hal ini untuk memastikan bahwa permintaan sesuai dengan kebijakan CORS (Cross-Origin Resource Sharing) yang ditetapkan oleh server. Kebijakan CORS ini memungkinkan server untuk menentukan domain tertentu yang diizinkan untuk mengakses sumber dayanya, sehingga hanya permintaan dari https://example.com yang akan diterima. Apabila tidak disertakan pengaturan CORS sesuai yang dikehendaki, maka seluruh permintaan tidak dapat diproses. Langkah ini penting untuk memastikan keamanan dan untuk mengendalikan akses ke API.

4.3.1 Pengujian Authentication User

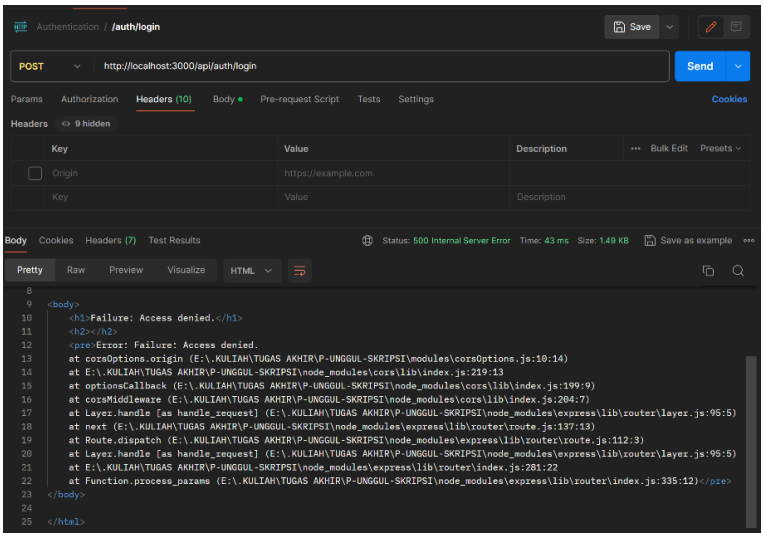
Pada tahap ini akan dilakukan proses pengujian autentikasi yang bertujuan untuk memastikan bahwa user yang mengakses sistem merupakan individu yang memiliki hak akses dan telah terdaftar dalam sistem. Pengujian dilakukan dengan mengirimkan parameter berupa *username* dan *password* yang valid serta menambahkan *header key* berupa "*origin"* dengan *value* "https://example.com" dan dilakukan dengan menggunakan metode *request* POST.



Gambar 4. 2 Pengujian *Login*

Pada Gambar 4.2 dapat dilihat telah terjadi proses *login user* yang mana akan menghasilkan sebuah token. Token tersebut terdiri dari *header* yang berisi algoritma dan tipe token yang digunakan, *payload* berisikan informasi tentang pengguna seperti ID pengguna, nama pengguna, email, dan lain-lain, serta *signature* yang digunakan untuk memverifikasi integritas token. Nilai token tersebut akan berbeda beda karena setiap token memiliki *payload* dan *signature* yang berbeda. Token tersebut berfungsi sebagai kunci untuk mendapatkan akses ketika melakukan *request* selanjutnya.

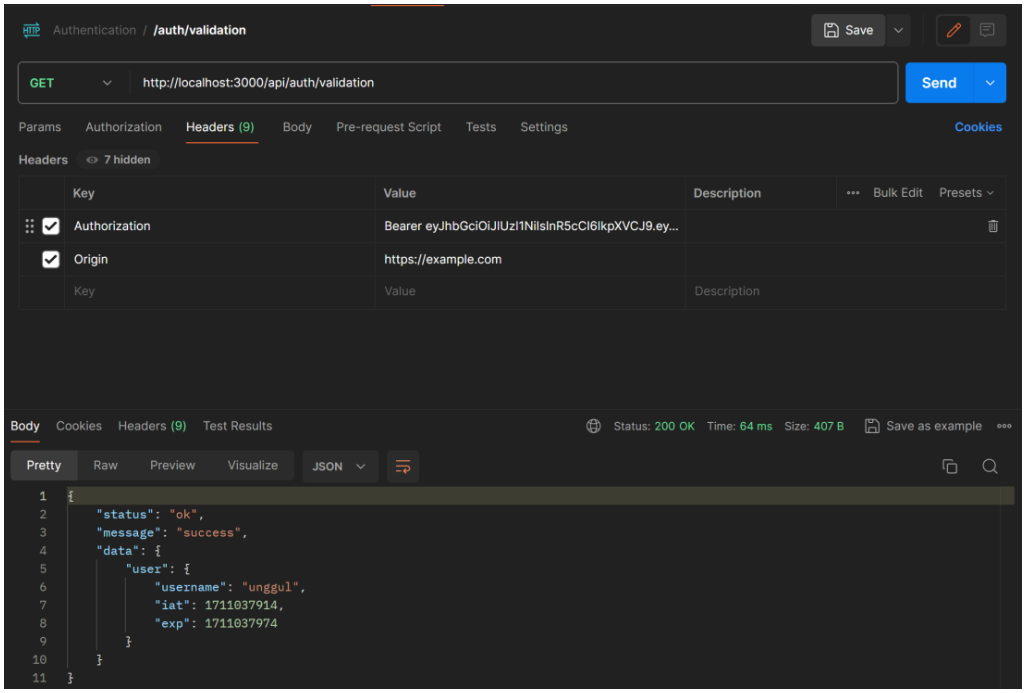
Dalam proses login dan permintaan lainnya apabila tidak menyertakan pengaturan CORS-Origin di dalam *header*, maka proses *login* dan permintaan lainnya akan gagal. Hal tersebut ditunjukkan pada Gambar 4.3 di bawah.



Gambar 4. 3 Penonaktian CORS

Gambar 4.3 menunjukkan bahwa pengujian CORS tidak berhasil karena *header* “Origin” dan nilainya telah dinonaktifkan. Hal tersebut mengindikasikan bahwa tidak adanya header “Origin” yang dikirim atau header tersebut tidak memiliki nilai yang valid dalam proses pengujian. Tanpa header tersebut, permintaan lintas asal domain tidak dapat diverifikasi keasliannya, sehingga mengakibatkan kegagalan saat pengujian. Nilai yang tidak valid dapat dianggap sebagai upaya serangan atau permintaan yang tidak sah.

Selanjutnya, dilakukan juga pengujian validasi token yang menggambarkan peran proses validasi JWT dalam sistem autentikasi dengan melakukan permintaan GET ke endpoint http://localhost:3000/api/auth/login. Dalam header request, terdapat sebuah header bernama “Auhtorization” yang berisikan token JWT. Token tersebut dikirim dengan menggunakan skema “Bearer” diikuti dengan nilai token JWT yang didapatkan dari proses login. Terdapat juga header "Origin" yang berisi nilai "https://example.com". Header ini digunakan untuk mengizinkan akses lintas sumber (Cross-Origin Resource Sharing/CORS) dari asal tertentu.



Gambar 4. 4 Proses Validasi Token

Gambar 4.4 menampilkan respon dari validasi token yang telah berhasil dieksekusi. Dengan mengirimkan token JWT yang valid dalam *header* “*Authorization*”, permintaan ini memungkinkan server untuk memverifikasi keabsahan token dan mengekstrak informasi pengguna yang terkandung di dalamnya. Jika token yang diproses valid, server akan mengembalikan respon dengan status “ok” dan mengirimkan informasi pengguna terkait dengan token tersebut. Informasi pengguna tersebut berupa username, iat (issued at time) yang

menunjukkan waktu penerbitan token, dan exp (expiration time) menunjukkan waktu kadaluarsa dari token. Proses validasi token JWT ini sangat penting dalam sistem otentikasi berbasis token, karena memungkinkan server untuk memastikan bahwa hanya pengguna yang memiliki token valid yang dapat mengakses sumber daya atau operasi yang dilindungi. Token JWT yang valid menunjukkan bahwa pengguna telah berhasil diautentikasi sebelumnya dan memiliki izin akses yang sesuai.

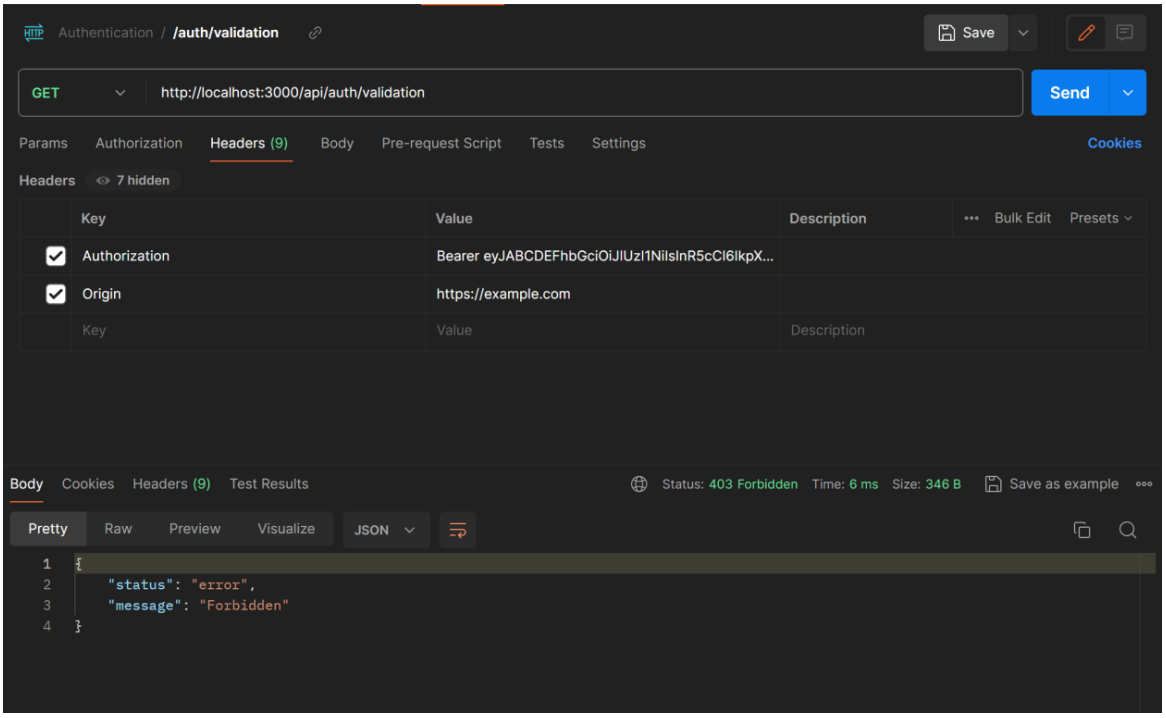
Selanjutnya akan dilakukan sebuah pengujian dimana pengguna memasukkan sebuah token yang tidak valid. Beberapa faktor yang dapat membuat sebuah token tidak valid antara lain:

1. Kadaluarsa waktu: Jika waktu yang ditentukan dalam klaim exp (expiration time) telah terlampaui, token akan menjadi tidak valid
2. Klaim tidak lengkap atau salah: Token JWT yang tidak memiliki klaim yang diperlukan atau memiliki klaim dengan nilai yang salah akan dianggap tidak valid
3. Manipulasi Token: Setiap perubahan yang dilakukan pada isi token tanpa menghasilkan ulang tanda tangan yang sesuai akan membuat token tidak valid

Pengujian dilakukan dengan memasukkan sebuah token yang dimodifikasi dengan menambahkan 2 karakter. Dengan menggunakan token asli yang didapat dari proses login, kemudian token JWT dirubah dengan menambahkan 5 karakter yaitu”ABCDEF”.

Table 4.1 Modifikasi Token

Token Sebelum Diubah	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InVuZ2d1bCIsImlhdCI6MTcxMTA2NDE0OCwiZXhwIjoxNzExMDY0MjA4fQ.pHafM6GvLpw-pVMAjAqG3kljqCf42xVl_9gZI2h6iKI
Token Setelah Dimodifikasi	eyJ**ABCDEF**hbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InVuZ2d1bCIsImlhdCI6MTcxMTA2NDE0OCwiZXhwIjoxNzExMDY0MjA4fQ.pHafM6GvLpw-pVMAjAqG3kljqCf42xVl_9gZI2h6iKI



Gambar 4. 5 Pengujian Token Tidak Valid

Gambar 4.5 menunjukkan bahwa sistem keamanan berhasil mendeteksi token yang tidak valid tersebut. Sistem akan menolak permintaan dan mengembalikan respon dengan status *error* dan message *forbidden*, bahwa permintaan ditolak karena token JWT yang diberikan tidak valid.

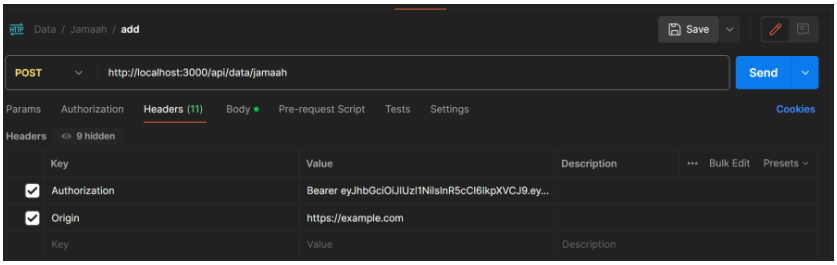
4.3.2 Pengujian Request Sumber Daya

Pengujian *request* sumber daya merupakan langkah penting dalam memastikan bahwa sistem berfungsi sebagaimana mestinya. Proses ini melibatkan pengiriman data ke endpoint yang ditentukan dan memeriksa respon yang diterima. Hal ini dilakukan untuk mengonfirmasi bahwa sumber daya yang diminta dapat diakses dengan benar dan data yang dikembalikan adalah akurat dan sesuai dengan harapan. Pengujian ini juga membantu mengidentifikasi masalah potensial dalam manajemen sumber daya, seperti kesalahan dalam penanganan permintaan atau dalam pengaturan keamanan yang dapat mempengaruhi integritas sistem.

Pada pengujian ini terdapat dua aspek utama yang akan diujikan. Aspek pertama, pengujian berfokus pada implementasi keamanan data untuk memastikan bahwa prosedur-prosedur JSON Web Tokens (JWT), Cross-Origin Resource Sharing (CORS), dan enkripsi database beroperasi dengan baik dalam mengelola data jamaah. Aspek kedua dari pengujian ini berfokus pada implementasi keamanan data pengguna. Pengujian ini bertujuan untuk

memastikan bahwa sistem dapat melindungi data pengguna dengan aman selama proses request sumber daya.

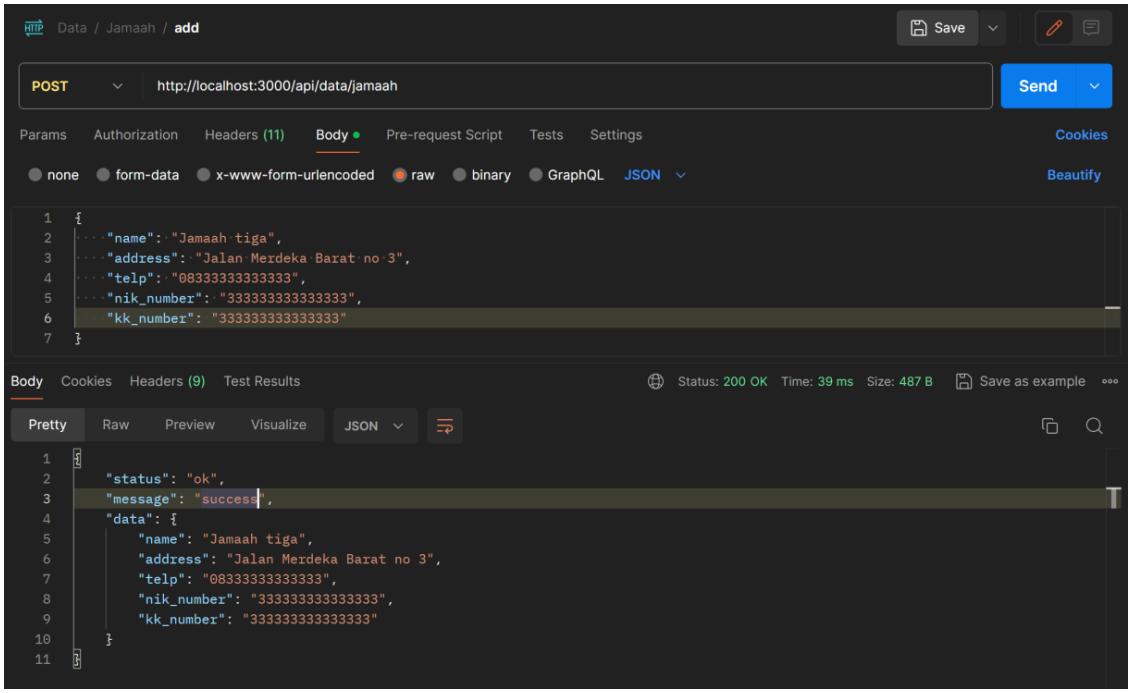
Dalam seluruh proses pengujian, terdapat dua header aktif yang digunakan, yaitu Authorization dan Origin. Header Authorization mengandung token Bearer, yang merupakan standar untuk menyediakan akses Token JWT ke sumber daya yang dilindungi. Token ini diperoleh melalui proses autentikasi dan dikirimkan sebagai bagian dari setiap request untuk menjamin bahwa hanya pengguna yang berwenang yang dapat melakukan request. Sementara itu, header Origin digunakan untuk menentukan asal request, yang membantu dalam mencegah serangan dengan membatasi request hanya dari asal yang diizinkan. Kedua header tersebut wajib diaktifkan dalam melakukan seluruh proses pengujian request sumber daya.



Gambar 4. 6 *Header Authorization* dan *Origin*

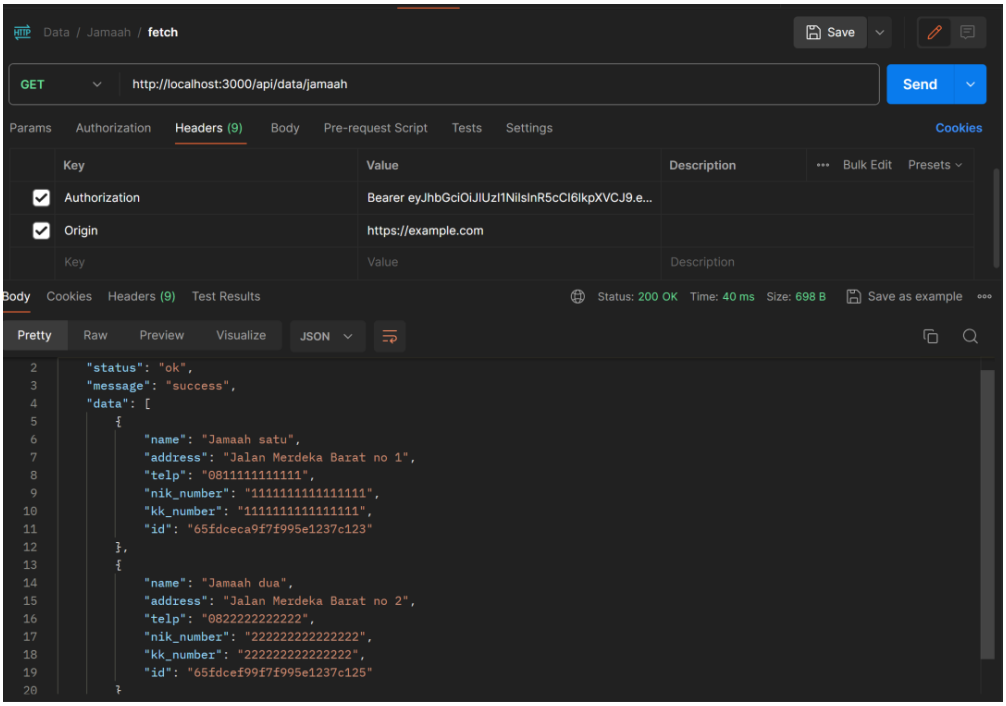
Pengujian Request Sumber Daya – Data Jamaah

Pada tahap ini dilakukan proses pengujian untuk memastikan implementasi keamanan dapat bekerja dalam proses pengelolaan masjid. Pengujian tersebut meliputi proses pembuatan data jamaah baru, melihat data jamaah, memperbarui data jamaah, serta menghapus data jamaah. Tujuan utama dari pengujian ini adalah untuk memastikan bahwa sistem keamanan yang telah diterapkan dapat beroperasi dengan baik dan melindungi data jamaah dari akses tidak sah atau penyalahgunaan, serta memenuhi standar keamanan data yang telah ditetapkan.



Gambar 4. 7 Penambahan Jamaah Baru

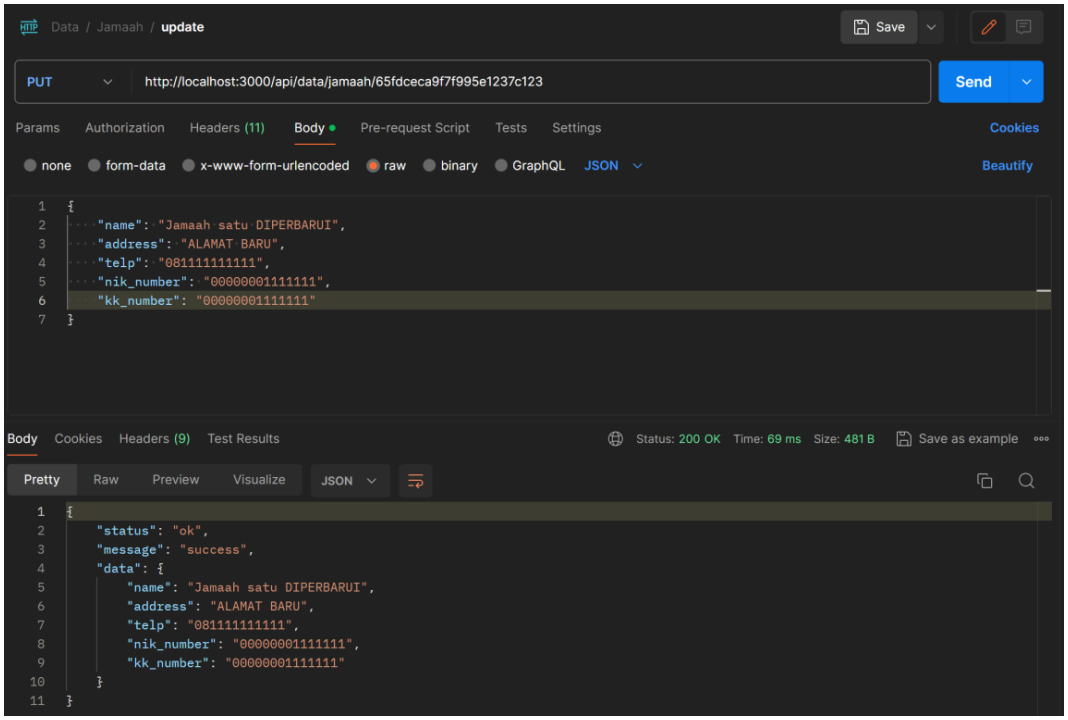
Pada Gambar 4.7, terlihat proses pengujian penambahan jamaah baru. Proses pengujian untuk menambahkan jamaah melalui API dilakukan dengan mengirimkan permintaan POST ke endpoint yang sesuai. Dalam pengujian ini, data jamaah yang diperlukan adalah nama, alamat, nomor telepon, nomor induk kependudukan, dan nomor kartu keluarga dikirimkan dalam bentuk JSON. Server kemudian akan memproses data tersebut dan jika berhasil akan mengembalikan respon status OK yang menandakan bahwa jamaah telah berhasil ditambahkan ke dalam sistem. Selanjutnya pengguna dapat melihat data jamaah yang telah dibuat pada gambar 4.8 di bawah.



Gambar 4. 8 Proses Fetch Jamaah

Pengujian fetch jamaah pada Gambar 4.8 melibatkan pengiriman permintaan GET ke endpoint API untuk melihat data jamaah yang tersimpan. Ketika permintaan GET diterima oleh server, server akan memverifikasi keabsahan token akses yang diberikan dalam header Authorization. Jika token akses valid, server akan mengizinkan akses ke sumber daya. Selain itu, server juga akan memeriksa header Origin untuk memastikan bahwa asal permintaan diizinkan sesuai dengan kebijakan CORS yang diterapkan. Setelah verifikasi berhasil, server akan merespon dengan kode status 200 OK dan mengembalikan data jamaah yang berisi informasi seperti nama, alamat, nomor telepon, nomor identitas (NIK dan KK), serta ID untuk setiap jamaah yang terdaftar dalam sistem.

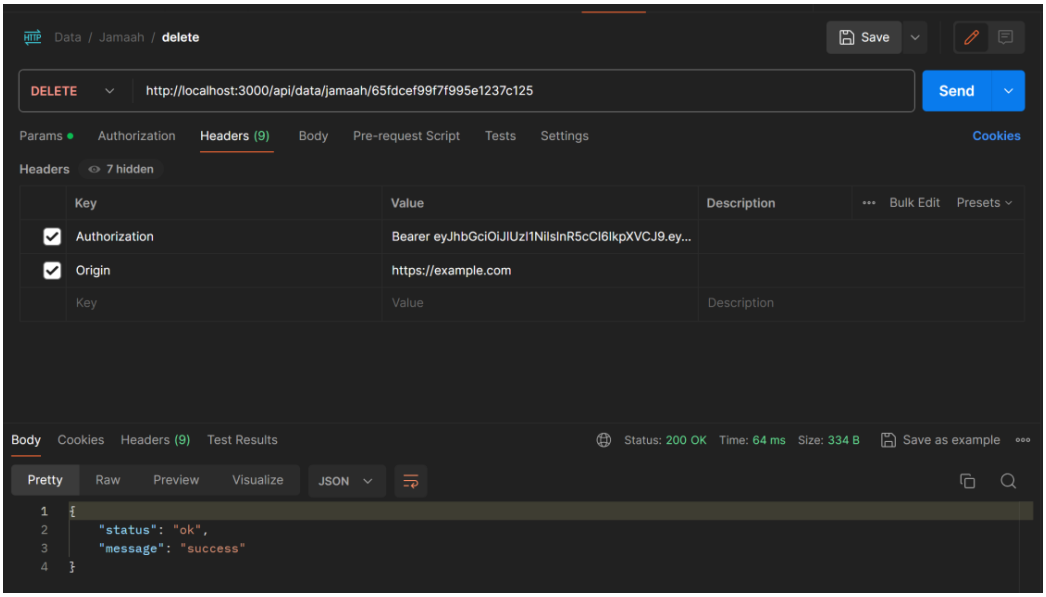
Pengguna juga dapat melakukan perubahan data dengan cara mengirimkan permintaan PUT ke endpoint http://localhost:3000/api/data/jamaah/(ID), disertai dengan header aktif seperti pengujian-pengujian sebelumnya yaitu authorization dan origin. ID adalah identifikasi unik untuk data jamaah yang diperoleh melalui proses pengambilan data (*fetch* data) sebelumnya.



Gambar 4. 9 Perubahan Data Jamaah

Terlihat pada Gambar 4.9 data jamaah dengan ID “65fdceca9f7f995e1237c123” telah berhasil diperbarui. Perubahan tersebut melibatkan penggunaan header authorization untuk menyertakan token akses JWT sebagai bukti otentikasi dan otorisasi pengguna yang melakukan permintaan perubahan data jamaah, serta header origin yang digunakan untuk mengidentifikasi asal permintaan. Tanpa header yang valid, sistem akan menolak permintaan pembaruan untuk mencegah akses tidak sah dan modifikasi data oleh pihak yang tidak berwenang.

Selain melakukan pembaruan data jamaah, pengguna juga dapat melakukan penghapusan data jamaah dengan menggunakan ID jamaah disertai aturan keamanan yang sama dengan proses perubahan data, di mana *header Authorization* dan *Origin* juga diperlukan. Hal ini memastikan bahwa setiap permintaan penghapusan diautentikasi dan berasal dari sumber yang terpercaya.

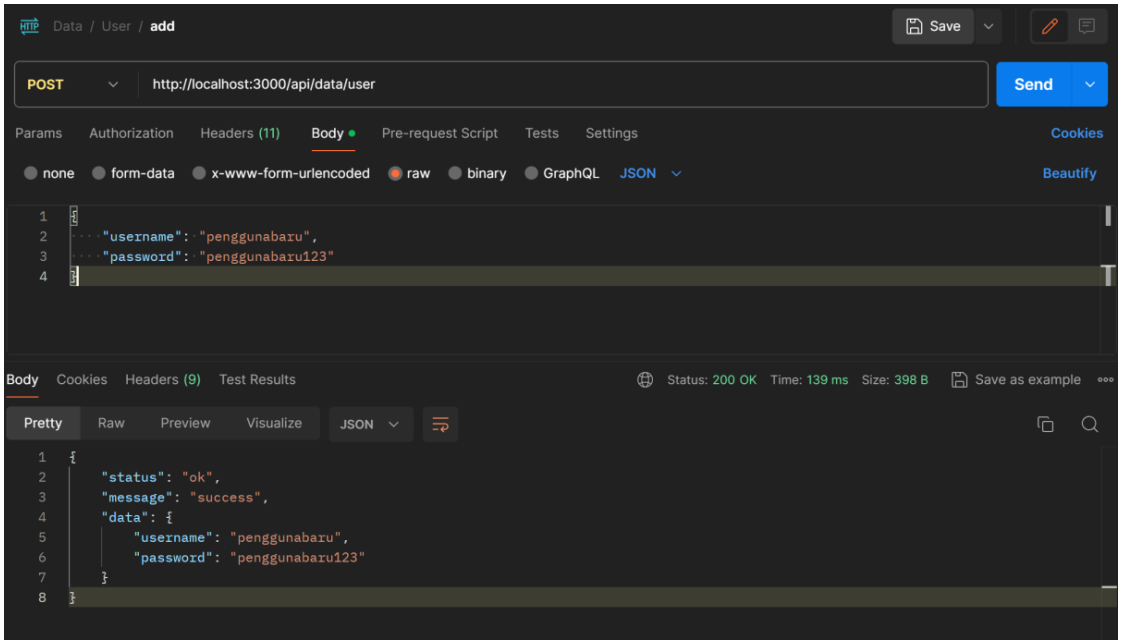


Gambar 4. 10 Penghapusan Data Jamaah

Pada Gambar 4.10 tersebut, terlihat sebuah pengujian yang dilakukan untuk menghapus data jamaah masjid yang sudah ada. Pengujian ini menggunakan metode permintaan *DELETE* ke endpoint http://localhost:3000/api/data/jamaah/(ID). Ketika server menerima permintaan *DELETE* tersebut, server akan memverifikasi token akses JWT yang diberikan dalam header Authorization serta memeriksa apakah asal permintaan yang tercantum dalam *header Origin* diizinkan sesuai dengan kebijakan CORS. Jika permintaan valid dan pengguna memiliki izin yang diperlukan, server akan menghapus data jamaah yang sesuai dengan ID yang diberikan dalam endpoint.

Pengujian Request Sumber Daya – Data User

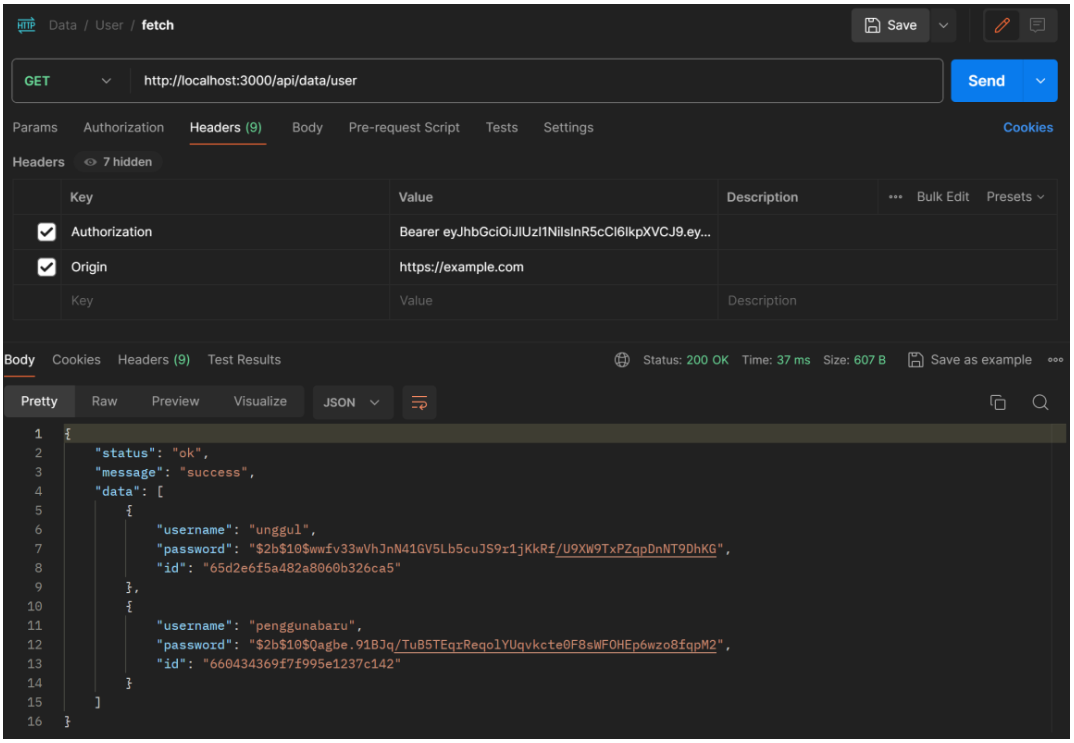
Pengujian keamanan tidak hanya terbatas pada data jamaah, tetapi juga mencakup proses request sumber daya akun pengguna sistem. Pengujian ini secara khusus menangani pembuatan akun pengguna baru, melihat data pengguna yang sudah ada, pembaruan informasi pengguna, serta prosedur penghapusan akun pengguna. Tujuan dari serangkaian pengujian ini adalah untuk memastikan bahwa mekanisme keamanan yang diimplementasikan dapat bekerja dengan baik dalam melindungi data pengguna dan jamaah dari intervensi yang tidak diinginkan.



Gambar 4. 11 Pembuatan Pengguna Baru

Pada Gambar 4.11 di atas, merupakan proses pengujian keamanan dalam pembuatan akun pengguna baru. Pengujian pembuatan akun pengguna baru melibatkan langkah-langkah penggunaan *header "Authorization"* yang berisi token *Bearer* untuk memverifikasi identitas pengguna dan memastikan bahwa hanya pengguna yang memiliki hak akses yang dapat membuat akun. *Header "Origin"* juga digunakan untuk menentukan sumber permintaan, yang membantu dalam mencegah serangan lintas domain dan memastikan bahwa permintaan tersebut sah. Selama pengujian, permintaan *POST* dikirim ke endpoint API yang sesuai dengan data yang diperlukan yaitu *"username"* dan *"password"*. Sistem kemudian memvalidasi data, memeriksa keamanan autentikasi, dan jika semua kriteria terpenuhi, akun pengguna baru berhasil dibuat.

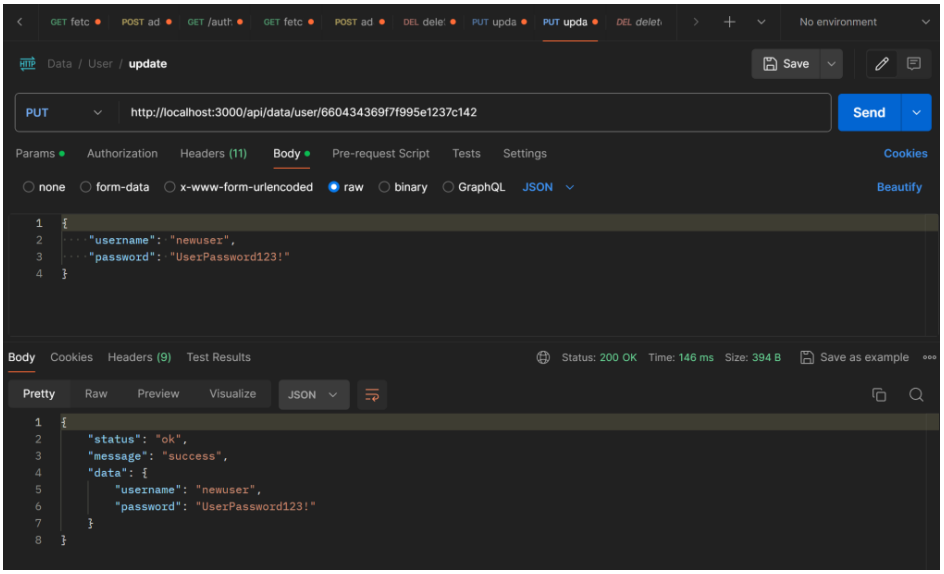
Setelah proses pembuatan pengguna baru, dilakukan proses *fetch* atau pengambilan data pengguna. Pengujian pengambilan data pengguna merupakan langkah penting untuk memastikan bahwa daftar pengguna dapat diakses dengan aman.



Gambar 4. 12 Fetch Data Pengguna

Dalam pengujian yang dilakukan pada Gambar 4.12, sebuah permintaan *GET* dikirimkan ke *endpoint* http://localhost:3000/api/data/user. Ketika server menerima permintaan ini, server akan melakukan verifikasi terhadap token akses JWT yang disertakan dalam header Authorization. Apabila token akses tersebut valid, server akan memberikan izin akses ke sumber daya yang diminta. Selain itu, server juga akan memeriksa nilai header *Origin* untuk memastikan bahwa permintaan berasal dari sumber yang diizinkan sesuai dengan kebijakan CORS yang diterapkan. Setelah proses verifikasi berhasil, server akan merespon dengan mengembalikan kode status 200 OK beserta data pengguna yang berisi username, password yang telah terenkripsi, dan ID untuk setiap pengguna yang terdaftar di dalam sistem.

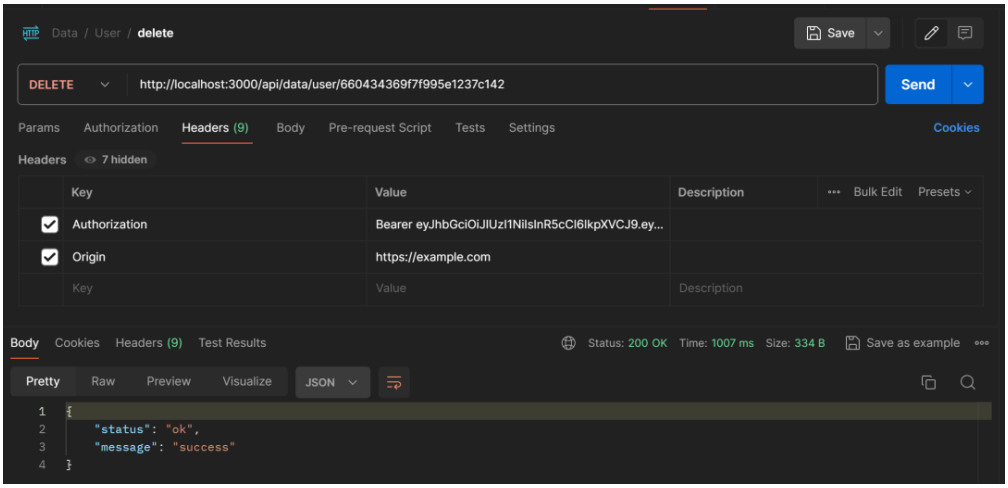
Pengguna juga dapat melakukan perubahan data dengan menggunakan ID pengguna yang didapatkan dari proses *fetch* data sebelumnya. Pengujian ini melibatkan penggunaan header Authorization yang mengandung bearer token untuk memastikan bahwa permintaan tersebut telah terotentikasi dan diberi otorisasi. *Header Origin* juga diperiksa untuk memastikan bahwa permintaan berasal dari sumber yang diizinkan.



Gambar 4. 13 Perubahan Data Pengguna

Pada Gambar 4.13, pengujian dilakukan dengan melakukan request metode PUT pada endpoint /api/data/user/ 660434369f7f995e1237c142 yang merupakan ID pengguna. Setelah melakukan permintaan PUT, respon yang diterima memiliki status “ok” dan pesan “success”, serta menampilkan informasi username dan password yang telah diperbarui dalam format JSON. Proses pengujian ini menekankan pentingnya mekanisme keamanan dalam API untuk melindungi data dan memastikan bahwa hanya permintaan yang sah yang dapat melakukan perubahan pada data pengguna.

Selain dapat melakukan perubahan data, pengguna juga dapat melakukan penghapusan data melalui metode *request DELETE* seperti pada Gambar 4.14 di bawah.



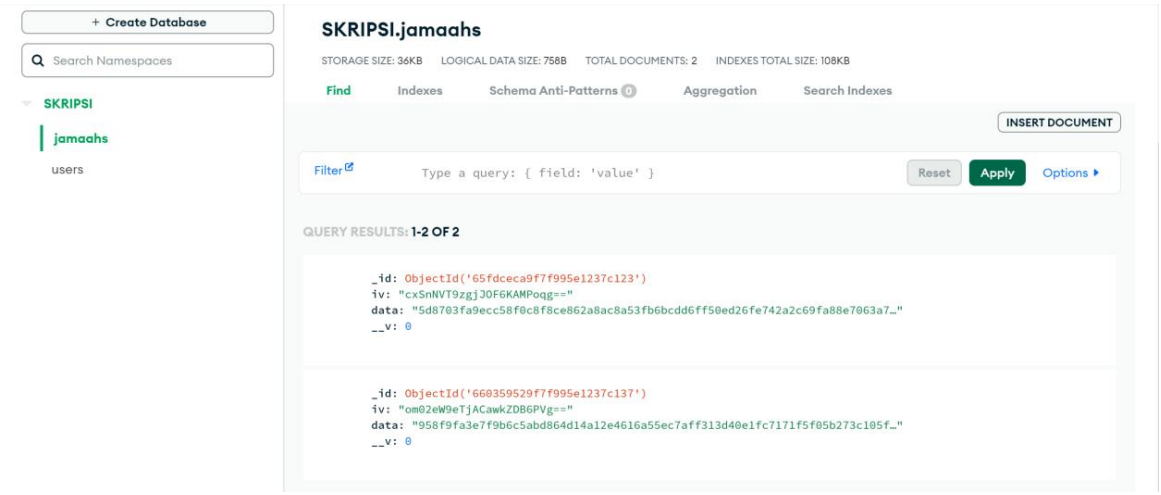
Gambar 4. 14 Penghapusan Data Pengguna

Dalam pengujian penghapusan data pengguna, header authorization dan origin masih tetap digunakan, hal itu berfungsi untuk memastikan bahwa permintaan penghapusan akun

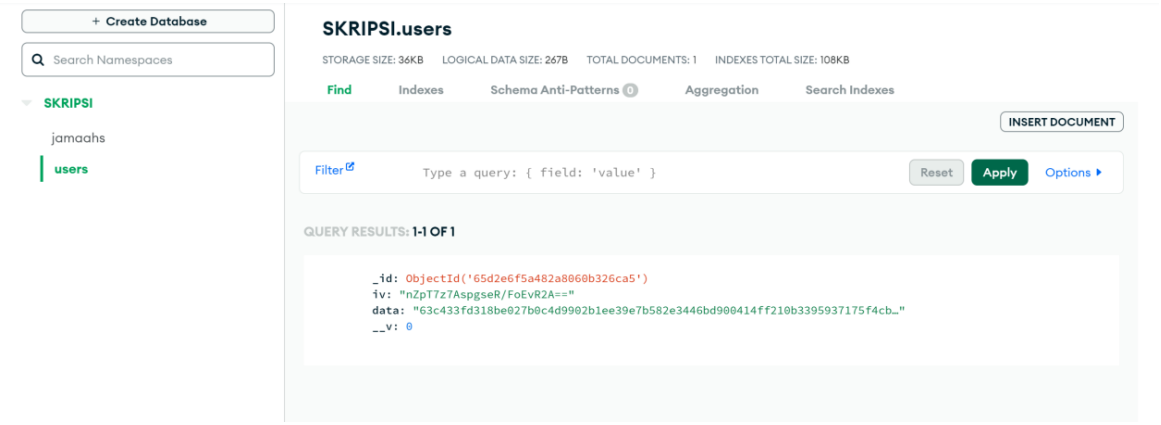
pengguna hanya dapat dilakukan oleh pihak yang berwenang. Proses pengujian berhasil dilakukan dengan mengirimkan request ke /api/data/user/ 660434369f7f995e1237c142 yang merupakan ID pengguna sebagai target akun yang ingin dihapus.

Hasil Enkripsi Database

Setelah uji coba *request* sumber daya dilakukan, seluruh data yang telah diolah akan disimpan di dalam database MongoDB dan dilakukan enkripsi terhadap data-data tersebut. Enkripsi dilakukan untuk mengamankan data dari kebocoran dan memastikan data-data tersebut tetap terlindungi saat sedang disimpan ataupun ditransmisikan.



Gambar 4. 15 Database MongoDB Jamaah Terenkripsi



Gambar 4. 16 Database MongoDB Pengguna Terenkripsi

Pada Gambar 4.15 dan Gambar 4.16 terlihat koleksi data yang terpisah sesuai jenis data yaitu data jamaah dan data user atau pengguna. Hal tersebut memudahkan proses pencarian dan pengambilan data ketika dibutuhkan. Pada setiap koleksi data yang ada pada database tersebut, terdapat dokumen yang di dalamnya memiliki sebuah ID, IV, dan data. ID merupakan

sebuah identifikasi dari setiap data yang berhasil dibuat. IV *(Initialization Vector)* merupakan komponen penting dalam algoritma enkripsi yang memastikan bahwa pesan yang sama tidak akan dienkripsi menjadi teks sandi yang sama, IV dibuat secara acak dan disimpan bersama teks sandi terenkripsi. Sedangkan data adalah tempat di mana semua data jamaah dan pengguna disimpan dalam bentuk terenkripsi. Data jamaah tersebut termasuk informasi seperti nama, alamat, nomor telepon, nomor KK dan NIK, sedangkan data pengguna yaitu email dan password pengguna.

4.4 *Black Box Testing*

Bagian ini akan membahas mengenai serangkaian pengujian yang telah sukses dilakukan. Pengujian tersebut menggunakan metode blackbox testing, yang bertujuan untuk mengevaluasi keamanan data pada sistem yang sudah dibuat. Walaupun tujuan utama dari pengujian adalah untuk mendeteksi kelemahan keamanan, metode blackbox testing ini juga menunjukkan bahwa setiap aspek keamanan yang diterapkan berfungsi dengan dengan baik.

Tabel 4.2 *Black Box Testing*

No	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	*Login* ke dalam sistem	1. Melakukan *request POST* ke *http://localhost:3000/api/auth/login* 2. Memasukkan *username* dan *password yang valid* 3. Memasukkan *header "origin"* dengan *value "https://example.com"*	Dapat masuk ke dalam sistem dengan menerapkan keamanan CORS serta menghasilkan token untuk melakukan *request* selanjutnya.	Sesuai Harapan	*Valid*
2.	Melihat Data Jamaah	1. Melakukan *request GET* ke *http://localhost:3000/api/data/jamaah* 2. Memasukkan *header* "*origin*" dengan value "*https://example.com*"	Dapat melihat data jamaah dengan menerapkan keamanan CORS dan JWT	Sesuai Harapan	*Valid*

		3. Memasukkan token ke dalam *header autorization*.			
3.	Menambah Data Jamaah	1. Melakukan *request POST* ke http://localhost:3000/api/data/jamaah 2. Memasukkan data jamaah berupa nama, alamat, no telepon, NIK, dan KK. 3. Memasukkan header "origin" dengan value "https://example.com" 4. Memasukkan token ke dalam header autorization.	Dapat melakukan penambahan data jamaah dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*
4.	Mengubah Data Jamaah	1. Melakukan request PUT ke http://localhost:3000/api/data/jamaah/(ID) 2. Memasukkan data jamaah yang ingin diubah. 3. Memasukkan header "origin" dengan value "https://example.com". 4. Memasukkan token ke dalam header autorization.	Dapat melakukan perubahan data jamaah dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*
5.	Menghapus Data Jamaah	1. Melakukan request DELETE ke http://localhost:3000/api/data/jamaah/(ID) 2. Memasukkan header "origin" dengan value "https://example.com".	Dapat melakukan penghapusan data jamaah dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*

		3. Memasukkan token ke dalam header autorization.			
6.	Melihat Data Pengguna	1. Melakukan request GET ke http://localhost:3000/api/data/user 2. Memasukkan header “origin” dengan value “https://example.com” 3. Memasukkan token ke dalam header autorization.	Dapat melihat data pengguna dengan menerapkan keamanan CORS dan JWT	Sesuai Harapan	*Valid*
7.	Menambah Data Pengguna	1. Melakukan request POST ke http://localhost:3000/api/data/user 2. Memasukkan data pengguna berupa username dan password. 3. Memasukkan header “origin” dengan value “https://example.com” 4. Memasukkan token ke dalam header autorization.	Dapat melakukan penambahan data pengguna dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*
8.	Mengubah Data Pengguna	1. Melakukan request PUT ke http://localhost:3000/api/data/user/(ID) 2. Memasukkan username password yang ingin diubah. 3. Memasukkan header “origin” dengan value “https://example.com”.	Dapat melakukan perubahan data pengguna dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*

		4. Memasukkan token ke dalam header autorization.			
9.	Menghapus Data Pengguna	1. Melakukan request DELETE ke http://localhost:3000/api/data/user/(ID) 2. Memasukkan header “origin” dengan value “https://example.com”. 3. Memasukkan token ke dalam header autorization.	Dapat melakukan penghapusan data pengguna dengan menerapkan keamanan JWT dan CORS	Sesuai Harapan	*Valid*
10	Enkripsi Data Jamaah dan Pengguna Pada DB	1. Melakukan Request CRUD pada sistem 2. Data berhasil disimpan pada database 3. Masuk ke dalam database mongoDB untuk melihat hasil	Terenkripsinya data jamaah dan pengguna pada tabel database MongoDB dengan menggunakan algoritma AES	Sesuai Harapan	*Valid*

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pada penelitian "Desain Komunikasi dan Keamanan Arsitektur Aplikasi MultiMasjid" beserta hasil pengujian yang telah dilakukan, penulis dapat menyimpulkan bahwa:

1. Perancangan desain komunikasi dan arsitektur keamanan Aplikasi MultiMasjid berhasil dibangun dan diuji dengan membuat API server yang mengimplementasikan *JSON Web Token* (JWT), enkripsi database dengan kriptografi AES, dan *Cross Origin Resource Sharing* (CORS).
2. Implementasi ini bertujuan untuk melindungi data pengguna dan jamaah dari akses tidak sah, serta menjaga kerahasiaan, integritas, dan ketersediaan data.
3. Pengujian yang dilakukan dengan metode *black box testing* menunjukkan bahwa setiap aspek keamanan yang diterapkan berfungsi dengan baik dan sistem mampu menangani berbagai skenario pengujian dengan hasil yang sesuai harapan.

5.2 Saran

Meskipun sistem keamanan data telah berhasil diimplementasikan dan diuji, ada beberapa saran yang dapat dipertimbangkan untuk pengembangan lebih lanjut:

1. Dapat mengimplementasikan sistem keamanan yang telah dibuat ke dalam aplikasi MultiMasjid.
2. Mempertimbangkan implementasi keamanan tambahan seperti enkripsi end-to-end, autentikasi multi-faktor, atau mekanisme keamanan lainnya untuk meningkatkan lapisan keamanan sistem.
3. Melakukan pengujian lebih lanjut dengan skenario yang lebih kompleks dan mencakup berbagai kemungkinan serangan atau eksploitasi keamanan untuk memastikan sistem benar-benar aman.
4. Mendokumentasikan proses pengembangan dan pengujian sistem keamanan dengan baik untuk memudahkan pemeliharaan dan pengembangan lebih lanjut di masa depan.

DAFTAR PUSTAKA

Aan, A., Daroini, S., & Yustanti, W. (2016). PERBANDINGAN PENGGUNAAN NOSQL MONGODB DAN MYSQL PADA BASIS DATA FORUM KOMUNIKASI. In *Jurnal Manajemen Informatika* (Vol. 6).

Chodorow, K., & Dirolf, M. (2019). *MongoDB: The Definitive Guide*. O'Reilly Media.

Elsera, M., Zakir, A., & Masjid Khairiyah, A. (2021). Sistem Informasi E-Smart Application Masjid Berbasis Web. In *Cetak) BuletinUtamaTeknik* (Vol. 16, Issue 2). Online.

Eric A. Fischer. (2005). *CRS Report for Congress Creating a National Framework for Cybersecurity: An Analysis of Issues and Options*.

Fauzi, D., Wibowo, A., Noor Fikri, A., Syukri, A., Larasati, A., Adhi, C., Meifara, H., Azzahra, Lestari, S., & Putri, Z. (2023). Peran CIA (Confidentiality, Integrity, Availability) pada Layanan Internet Banking di Perbankan. *Jurnal Ilmu Multidisplin*. https://doi.org/10.38035/jim.v2i1.230

Herfandi, H., & Hamdani, F. (2022). Implementasi Sistem Informasi Manajemen Masjid Berbasis Web. In *Informatics Journal* (Vol. 7, Issue 3). https://doi.org/10.19184/isj.v7i3.34233

Jayantina, M. (2018). Tantangan Dalam Implementasi Strategi Keamanan Siber Nasional Indonesia Ditinjau Dari Penilaian Global Cybersecurity Index. *Masyarakat Telematika Dan Informasi : Jurnal Penelitian Teknologi Informasi Dan Komunikasi*, *8*, 137. https://doi.org/10.17933/mti.v8i2.108

Junian Dani, author. (2008). *Pengembangan kebijakan keamanan informasi pada perusahaan jasa layanan kurir: studi kasus PT. NCS*. Fakultas Ilmu Komputer Universitas Indonesia. https://lib.ui.ac.id

Kumar, P. R., Raj, P. H., & Jelciana, P. (2018). Exploring Data Security Issues and Solutions in Cloud Computing. *Procedia Computer Science*, *125*, 691–697. https://doi.org/10.1016/J.PROCS.2017.12.089

Mahindrakar, P., & Pujeri, U. (2020). Security Implications for Json Web Token Used in MERN Stack for Developing E Commerce Web Application. *International Journal of Engineering and Advanced Technology*, *10*(1), 39–45. https://doi.org/10.35940/ijeat.A1663.1010120

Mukhtar Harun. (2018). *Kriptografi untuk Keamanan Data*. 7–9.

Mustika, L. (2020). Implementasi Algoritma AES Untuk Pengamanan Login Dan Data Customer Pada E-Commerce Berbasis Web. *JURIKOM (Jurnal Riset Komputer)*, *7*(1), 148. https://doi.org/10.30865/jurikom.v7i1.1943

Noor, M., & Azam, A. (2016). Otentikasi Sistem Dengan Menggunakan One Time Password Memanfaatkan Smartphone Android. *Lintas Sistem Informasi Dan Komputer (LINK)*. https://doi.org/10.31090/link.v24i1.5

Novryaldy, A., & Setiadi, T. (2018). Perancangan Sistem Informasi Profil Masjid Berbasis Website. *Tedy Setiadi Jurnal Ilmiah Teknologi Informasi Terapan*, *IV*(3). https://doi.org/10.33197/jitter.vol4.iss3.2018.172

Patil, A., & Meshram, B. B. (2012). Database Access Control Policies. *International Journal of Engineering Research and Applications (IJERA)*, *2*, 3150–3154. www.ijera.com

Purnasari, M., Hartiwi, Y., & Nurhayati, N. (2022). Perancangan Sistem Informasi Pengelolaan Dana Masjid Berbasis Web Menggunakan Unified Modeling Language (UML). *Resolusi : Rekayasa Teknik Informatika Dan Informasi*, *2*(6), 258–264. https://doi.org/10.30865/RESOLUSI.V2I6.416

Qadaruddin, Q., Nurkidam, A., & Firman, F. (2016). Peran Dakwah Masjid dalam Peningkatan Kualitas Hidup Masyarakat. *Ilmu Dakwah: Academic Journal for Homiletic Studies*, *10*, 222–239. https://doi.org/10.15575/idajhs.v10i2.1078

Qadir, S., & Quadri, S. (2016). Information Availability: An Insight into the Most Important Attribute of Information Security. *Journal of Information Security*, *07*, 185–194. https://doi.org/10.4236/jis.2016.73014

Rahma, F., & Raf'ie, A. (2020). *Keamanan Siber dan Informasi v1*.

Rahmadi, P., & Yunita, H. (2020). Implementasi Pengamanan Basis Data Dengan Teknik Enkripsi. *Jurnal Cendikia*, *19*(1). https://jurnal.dcc.ac.id/index.php/JC/article/view/331

Rahmatulloh, A., Gunawan, R., & Nursuwars, F. M. S. (2019). Performance Comparison of Signed Algorithms on JSON Web Token. *IOP Conference Series: Materials Science and Engineering*, *550*(1), 12023. https://doi.org/10.1088/1757-899X/550/1/012023

Shoufan, A., & Damiani, E. (2017). On Inter-Rater Reliability of Information Security Experts. *Journal of Information Security and Applications*, *37*, 101–111. https://doi.org/10.1016/J.JISA.2017.10.006

Siregar, R. S., Asih, M. S., & Wulan, N. (2019). Penerapan Algoritma RC4 Dan Rail Fence Untuk Enkripsi Database Mahasiswa Pada Kampus Poltekkes Kemenkes Medan. *JiTEKH*, *7*(2), 51–56. https://doi.org/10.35447/JITEKH.V7I02.78

Sukmawan, D. I., Putra, D., & Pertahanan, S. K. (2023). *Hacker, Fear, and Harm: Data Breaches and National Security Peretas, Ketakutan, dan Kerugian: Pelanggaran Data dan Keamanan Nasional*. https://doi.org/10.20473/jgs.17.1.2023.153-182

Syofian, S., Indra, R., & Susanto, S. (2018). Implementasi Management Akses User Untuk Router Cisco Menggunakan Metode AAA (Authentication, Authorization, Accounting) Studi Kasus PT. Proxis Sahabat Indonesia. *Jurnal Sains Dan Teknologi (JST)*.

Toyib, R., & Wijaya, A. (2018). Analisis Perbandingan Algoritma Simetris Rivest Code 5 Dengan Algoritma Simetris Rivest Code 6 (Studi Kasus: SMK Negeri Seluma). *JURNAL INFORMATIKA UPGRIS*, *4*(2). https://doi.org/10.26877/jiu.v4i2.2840

Yel, M. B., & Nasution, M. K. M. (2022). Keamanan Informasi Data Pribadi Pada Media Sosial. *Jurnal Informatika Kaputama (JIK)*, *6*(1), 92–101. https://doi.org/10.59697/jik.v6i1.144

LAMPIRAN