PERANCANGAN DAN IMPLEMENTASI KONTROL AKSES PENGGUNA INTERNAL DENGAN PENERAPAN *LEAST PRIVILEGE* DAN *JUST-IN-TIME ACCESS* PADA *PROTOTYPE* SISTEM DOMPET DIGITAL

TUGAS AKHIR

Diajukan Oleh :

MUHAMMAD SYUKUR

220705058

Mahasiswa Fakultas Sains dan Teknologi

Program Studi Teknologi Informasi

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI AR-RANIRY
BANDA ACEH
2026 M/1448 H

LEMBAR PERSETUJUAN

PERANCANGAN DAN IMPLEMENTASI KONTROL AKSES PENGGUNA INTERNAL DENGAN PENERAPAN *LEAST PRIVILEGE* DAN *JUST-IN-TIME ACCESS* PADA *PROTOTYPE* SISTEM DOMPET DIGITAL

TUGAS AKHIR

Diajukan kepada Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Ar-Raniry Banda Aceh
Sebagai salah satu Beban Studi Memperoleh Gelar Sarjana (S1)
Dalam Ilmu Teknologi Informasi

Oleh :

Muhammad Syukur
NIM. 220705058

Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

Disetujui untuk Dimunaqasyahkan oleh :

Pembimbing I,	Pembimbing II,
Ghufran Ibnu Yasa, M.T.	**Mulkan Fadhli, S.T., M.T.**
NIP. 198409262014031005	**NIP. 198811282020121006**

Mengetahui,
Ketua Program Studi Teknologi Informasi,

Malahayati, M.T.
NIP. 198301272015032003

LEMBAR PENGESAHAN

PERANCANGAN DAN IMPLEMENTASI KONTROL AKSES PENGGUNA INTERNAL DENGAN PENERAPAN *LEAST PRIVILEGE* DAN *JUST-IN-TIME ACCESS* PADA *PROTOTYPE* SISTEM DOMPET DIGITAL

TUGAS AKHIR

Telah Diuji Oleh Panitia Ujian Munaqasyah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S1)
Dalam Program Studi Teknologi Informasi

Pada Hari/Tanggal: Rabu, 12 Agustus 2026 M
28 Shafar 1448 H
Di Darussalam, Banda Aceh

Panitia Ujian Munaqasyah Tugas Akhir:

Ketua,	Sekretaris,
Ghufran Ibnu Yasa, M.T	**Mulkan Fadhli, S.T., M.T**
NIP. 198409262014031005	**NIP. 198811282020121006**
Penguji I,	Penguji II,
Nizam Albar, S.T., M.T.	**Dr.Hendri Ahmadian, M.I.M**
NUPTK. 2849779680130032	**NIP. 198301042014031002**

Mengetahui:
Dekan Fakultas Sains dan Teknologi
UIN Ar-Raniry Banda Aceh

Prof. Dr. Ir. Muhammad Dirhamsyah, M.T., IPU
NIP. 196210021988111001

LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama	:	Muhammad Syukur
NIM	:	220705058
Program Studi	:	Teknologi Informasi
Fakultas	:	Sains dan Teknologi
Judul	:	Perancangan dan Implementasi Kontrol Akses Pengguna Internal dengan Penerapan *Least Privilege* dan *Just-In-Time Access* pada *Prototype* Sistem Dompet Digital

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggungjawabkan;
2. Tidak melakukan plagiasi terhadap naskah atau karya orang lain;
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya;
4. Tidak memanipulasi dan memalsukan data;
5. Mengerjakan sendiri karya ini dan mampu bertanggung jawab atas karya ini.

Apabila di kemudian hari terdapat tuntutan dari pihak lain atas karya saya dan, setelah melalui proses pembuktian yang dapat dipertanggungjawabkan, ditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya bersedia dikenai sanksi sesuai dengan ketentuan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh. Demikian pernyataan ini saya buat dengan sesungguhnya, dalam keadaan sadar, dan tanpa paksaan dari pihak mana pun.

Banda Aceh, 15 Agustus 2026

Yang Menyatakan,

Muhammad Syukur

ABSTRAK

Pengguna internal pada sistem yang mengelola data sensitif memiliki kebutuhan akses yang sah untuk menjalankan tugas operasional, tetapi akses yang terlalu luas atau tersedia secara permanen dapat meningkatkan risiko penyalahgunaan hak akses. Penelitian ini bertujuan merancang, mengimplementasikan, dan mengevaluasi kontrol akses pengguna internal pada *layer* aplikasi dalam *prototype* sistem dompet digital dengan menggunakan *Role-Based Access Control* (RBAC) sebagai kontrol dasar, prinsip *Least Privilege* (LP) sebagai pembatas lingkup akses, serta mekanisme *Just-In-Time* (JIT) *Access* sebagai pembatas akses sementara terhadap fitur sensitif. Penelitian menggunakan pendekatan terapan berbasis *Prototype* dengan fokus pada backend *Access* control *layer*. Customer Support (CS) digunakan sebagai aktor utama dengan alur operasional berbasis ticket. *Least Privilege* diterapkan melalui pembatasan akses berdasarkan assignment ticket, sedangkan *Just-In-Time Access* diterapkan melalui session sementara yang mengikat CS, ticket, feature, status aktif, dan waktu kedaluwarsa. Evaluasi dilakukan menggunakan scenario-based testing dengan pendekatan black-box testing terhadap keputusan backend dalam menerima atau menolak request. Hasil pengujian terhadap 19 skenario menunjukkan seluruh hasil aktual sesuai dengan hasil yang diharapkan. RBAC berhasil membatasi akses berdasarkan role, *Least Privilege* membatasi ruang lingkup akses CS berdasarkan assignment ticket, sedangkan *Just-In-Time Access* membatasi penggunaan fitur sensitif berdasarkan konteks ticket dan masa aktif session. Hasil tersebut menunjukkan bahwa kombinasi ketiga konsep tersebut dapat menghasilkan kontrol akses yang lebih kontekstual dibandingkan RBAC dasar dalam ruang lingkup *Prototype* yang dikembangkan.

Kata Kunci: kontrol akses, *Least Privilege*, *Just-In-Time* Access, *Role-Based Access Control*, pengguna internal

ABSTRACT

Internal users in systems that manage sensitive data require legitimate Access to perform operational tasks; however, Access that is too broad or permanently available may increase the risk of Access misuse. This study aims to design, implement, and evaluate internal user Access control at the application layer in a digital wallet system Prototype by using Role-Based Access Control (RBAC) as the basic control, the Least Privilege (LP) principle to restrict Access scope, and the Just-In-Time (JIT) Access mechanism to provide temporary Access to sensitive features. This study applies an applied research approach using a Prototype, with a focus on the backend Access control layer. Customer Support (CS) is used as the primary actor within a ticket-based operational workflow. Least Privilege is implemented by restricting Access based on ticket assignment, while Just-In-Time Access is implemented through temporary sessions that bind the CS, ticket, feature, active status, and expiration time. The evaluation is conducted using scenario-based testing with a black-box testing approach to assess backend decisions in accepting or rejecting requests. The results of 19 test scenarios show that all actual results match the expected results. RBAC successfully restricts Access based on user roles, Least Privilege limits the scope of CS Access based on ticket assignment, while Just-In-Time Access restricts the use of sensitive features based on ticket context and session validity period. These results indicate that the combination of the three mechanisms can provide more contextual Access control than basic RBAC within the scope of the developed Prototype.

Keywords: *Access control, Least Privilege, Just-In-Time Access, Role-Based Access Control, internal users.*

KATA PENGANTAR

Bismillahirrahmanirrahim.

Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT atas rahmat, kemudahan, dan pertolongan-Nya sehingga tugas akhir berjudul "Perancangan dan Implementasi Kontrol Akses Pengguna Internal dengan Penerapan *Least Privilege* dan *Just-In-Time Access* pada *Prototype* Sistem Dompet Digital" dapat diselesaikan. Shalawat dan salam semoga senantiasa tercurah kepada Nabi Muhammad SAW, keluarga, sahabat, serta umat beliau hingga akhir zaman.

Tugas akhir ini disusun sebagai salah satu syarat memperoleh gelar Sarjana pada Program Studi Teknologi Informasi, Fakultas Sains dan Teknologi, Universitas Islam Negeri Ar-Raniry Banda Aceh. Lebih dari sekadar kewajiban akademik, proses penyusunannya menjadi perjalanan pembelajaran yang mempertemukan penulis dengan berbagai tantangan, pengetahuan, dan pengalaman yang turut membentuk cara berpikir serta kedewasaan dalam menyelesaikan suatu proses.

Terselesaikannya tugas akhir ini tidak terlepas dari bantuan, doa, arahan, dan dukungan berbagai pihak. Dengan penuh rasa hormat dan syukur, penulis menyampaikan terima kasih kepada:

1) Kedua orang tua penulis, atas doa, kasih sayang, pengorbanan, kepercayaan, dan dukungan yang tidak pernah terputus. Segala yang telah diberikan menjadi kekuatan utama bagi penulis dalam menyelesaikan pendidikan ini. Semoga Allah SWT senantiasa memberikan kesehatan, keberkahan, dan balasan terbaik kepada keduanya.
2) Bapak Ghufran Ibnu Yasa, M.T., selaku pembimbing akademik sekaligus Pembimbing I, atas bimbingan, arahan, evaluasi, dan dukungan yang diberikan selama proses akademik hingga penyusunan tugas akhir ini.

3) Bapak Mulkan Fadhli, S.T., M.T., selaku Pembimbing II, atas waktu, perhatian, kritik, dan masukan yang membantu penulis menyusun penelitian ini secara lebih terarah dan sistematis.

4) Seluruh dosen Program Studi Teknologi Informasi, Fakultas Sains dan Teknologi, Universitas Islam Negeri Ar-Raniry Banda Aceh, atas ilmu, wawasan, dan pengalaman yang diberikan selama masa perkuliahan serta menjadi bagian penting dari dasar penyusunan tugas akhir ini.

5) Selama proses penyusunan tugas akhir, Ibu Cut Ida Rahmadiana, S.Si., seorang staf Program Studi Teknologi Informasi, telah memberikan bantuan dan dukungan dalam berbagai urusan akademik.

6) Teman-teman saya baik dari Program Studi Teknologi Informasi dan teman dekat lainnya, atas kebersamaan, bantuan, diskusi, semangat, dan berbagai pengalaman yang mewarnai perjalanan selama masa perkuliahan.

Penulis dengan jujur mengakui bahwa masih banyak kekurangan dalam tugas akhir ini, baik dari segi penulisan teknis maupun konten. Oleh karena itu, penulis dengan rendah hati menerima kritik dan saran yang bermanfaat untuk pengembangan di masa depan. Terakhir, semoga tugas akhir ini menjadi amal baik yang tercatat di sisi Allah SWT dan bermanfaat bagi penulis dan pembaca. Aamiin ya Rabbal ‘alamin.

Banda Aceh, 7 Juli 2026

Penulis,

Muhammad Syukur

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam sistem digital modern yang mengelola data sensitif, keberadaan pengguna internal merupakan bagian yang tidak terpisahkan dari operasional sistem. Pengguna internal memiliki akses terhadap data dan fungsi tertentu untuk mendukung aktivitas seperti verifikasi akun, pemantauan transaksi, dan penanganan permintaan pengguna (Omotunde & Ahmed, 2023). Akses tersebut diberikan secara sah sebagai bagian dari kebutuhan operasional sistem. Namun, di sisi lain, pengguna internal dengan akses sah juga menjadi salah satu sumber risiko utama dalam keamanan sistem (Usman dkk., 2025). Berbeda dengan ancaman eksternal, aktivitas pengguna internal berlangsung dalam lingkup kewenangan yang valid, sehingga penyalahgunaan akses sulit dibedakan dari aktivitas normal. Kondisi ini meningkatkan potensi terjadinya insider threat, yaitu penyalahgunaan akses oleh pihak internal yang memiliki kredensial resmi dalam sistem.

Kasus nyata menunjukkan bahwa penyalahgunaan akses internal merupakan ancaman yang signifikan dalam sistem yang mengelola data sensitif. Salah satu kasus terjadi pada FinWise Bank, di mana seorang mantan pegawai masih memiliki akses ke sistem setelah masa kerja berakhir dan menggunakan akses tersebut untuk mengambil data pelanggan yang berkaitan dengan American First Finance. Insiden ini berdampak pada sekitar 689.000 pengguna dan dilaporkan sebagai pelanggaran data, yang menunjukkan bahwa kegagalan dalam pengelolaan akses internal dapat menyebabkan kebocoran data dalam skala besar (Penta Security, 2025). Kasus lain terjadi pada DBS Bank, di mana seorang petugas penagihan dengan akses ke sistem internal *customer information system* menyalahgunakan hak akses untuk mengambil data nasabah tanpa otorisasi dan membagikannya kepada pihak lain. Dalam laporan resmi disebutkan bahwa pelaku merupakan pengguna internal dengan akses sah, namun melakukan akses di luar kebutuhan tugasnya dan kemudian dijatuhi hukuman pidana. Kasus ini menegaskan bahwa akses yang tidak dibatasi berdasarkan kebutuhan operasional berpotensi disalahgunakan meskipun diberikan secara sah (The Straits Times, 2022).

Selain contoh kasus tersebut, laporan *Data Breach Investigations Report* 2024 dari Verizon menunjukkan bahwa *human element* menjadi komponen dalam 68% pelanggaran data, setelah metrik tersebut mengecualikan *malicious privilege misuse*. Temuan ini menunjukkan bahwa faktor manusia dan pengelolaan akses tetap menjadi aspek penting dalam keamanan sistem, terutama pada sistem yang mengelola data sensitif dan melibatkan pengguna internal dengan akses operasional. Oleh karena itu, pembatasan hak akses tidak cukup hanya bergantung pada keberhasilan autentikasi atau pembagian *role*, tetapi perlu diperkuat dengan pembatasan berdasarkan konteks tugas dan durasi akses (Verizon, 2024).

Kedua kasus dan temuan laporan tersebut menunjukkan pola yang relevan, yaitu bahwa risiko keamanan tidak hanya berasal dari serangan eksternal, tetapi juga dapat muncul dari kelemahan pengelolaan hak akses setelah proses autentikasi. Pada banyak sistem, kontrol akses masih bersifat statis, di mana pengguna internal diberikan hak akses berdasarkan peran tanpa pembatasan terhadap konteks penggunaan maupun durasi akses. Akibatnya, pengguna dapat mengakses data atau fungsi yang tidak selalu diperlukan serta mempertahankan akses tersebut dalam jangka waktu yang tidak terbatas.

Berbagai penelitian telah mengusulkan pendekatan untuk mengatasi permasalahan tersebut, khususnya melalui penerapan prinsip *Least Privilege* dan mekanisme *Just-In-Time Access*. *Least Privilege* membatasi hak akses pengguna hanya pada kebutuhan minimum sesuai tugasnya, sedangkan *Just-In-Time Access* memberikan akses terhadap fungsi sensitif secara sementara berdasarkan kebutuhan. Kombinasi kedua konsep ini banyak digunakan untuk mengurangi standing privilege dan membatasi peluang penyalahgunaan akses internal. Namun demikian, sebagian besar implementasi dan penelitian sebelumnya masih berfokus pada level infrastruktur, seperti pengelolaan akses pada server, sistem operasi, database, serta lingkungan IT dan DevOps. Pendekatan tersebut belum banyak memberikan gambaran implementasi pada *layer* (lapisan) kontrol akses aplikasi, yaitu pada level di mana interaksi langsung antara pengguna internal dan fitur sistem terjadi. Padahal, pada sistem yang mengelola data sensitif, sebagian besar akses terhadap data dan fungsi kritis berlangsung pada *layer* aplikasi. Oleh karena itu, diperlukan pendekatan

yang tidak hanya mengadopsi konsep *Least Privilege* dan *Just-In-Time Access*, tetapi juga mengimplementasikannya secara langsung pada *layer* kontrol akses aplikasi untuk pengguna internal, sehingga pembatasan lingkup dan durasi akses dapat diterapkan secara lebih kontekstual.

Dalam konteks tersebut, *prototype* sistem dompet digital dalam penelitian ini digunakan sebagai media implementasi untuk merepresentasikan sistem yang mengelola data sensitif dan melibatkan pengguna internal dengan hak akses operasional. Berbeda dengan penelitian terdahulu yang umumnya berfokus pada penerapan kontrol akses di level infrastruktur, penelitian ini menitikberatkan pada implementasi mekanisme kontrol akses pada *layer* aplikasi. Oleh karena itu, *prototype* ini digunakan untuk memberikan gambaran bagaimana prinsip *Least Privilege* dan *Just-In-Time Access* diterapkan dalam mengatur akses pengguna internal terhadap data dan fitur sensitif pada level aplikasi secara terstruktur.

Berdasarkan hal tersebut, penelitian ini berfokus pada perancangan dan implementasi *Least Privilege* dan *Just-In-Time Access* pada *layer* kontrol akses pengguna internal dalam *prototype* sistem dompet digital. Dengan menggunakan *Role-Based Access Control* (RBAC) sebagai model dasar, penelitian ini menerapkan pembatasan hak akses berbasis *Least Privilege* serta mekanisme akses sementara melalui *Just-In-Time Access*. Dalam praktik industri finansial berskala besar, mekanisme keamanan dan kontrol akses internal umumnya telah diterapkan melalui arsitektur yang lebih kompleks, matang, dan disesuaikan dengan kebutuhan masing-masing organisasi. Namun, detail implementasi teknis pada *layer* aplikasi komersial umumnya tidak dipublikasikan secara terbuka karena berkaitan dengan strategi keamanan internal. Oleh karena itu, penelitian ini tidak diarahkan untuk menandingi atau mengukur tingkat keamanan sistem industri, melainkan untuk memodelkan secara akademik konsep kontrol akses internal berbasis *Least Privilege* dan *Just-In-Time Access* dalam bentuk *prototype* yang terbuka, terstruktur, dan dapat diverifikasi. Melalui *prototype* ini, konsep yang umumnya diterapkan dalam lingkungan tertutup dapat direpresentasikan pada backend aplikasi sehingga dapat dipelajari, diuji, dan dijadikan rujukan teknis dasar dalam konteks akademik.

Evaluasi dalam penelitian ini dilakukan melalui pengujian skenario untuk memverifikasi kesesuaian kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* dengan rancangan sistem. Hasil pengujian digunakan untuk menunjukkan bahwa mekanisme tersebut mampu membatasi lingkup dan durasi akses pada backend *prototype* sistem dompet digital sebagai penguatan terhadap model RBAC dasar.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, Rumusan masalah dalam penelitian ini adalah sebagai berikut:

1) Bagaimana mengidentifikasi kebutuhan kontrol akses pengguna internal melalui pemodelan peran, data sensitif, dan fitur sistem dalam *prototype* sistem dompet digital?
2) Bagaimana merancang dan mengimplementasikan prinsip *Least Privilege* dan mekanisme *Just-In-Time Access* pada backend *Access control layer* dalam membatasi serta mengatur akses terhadap data dan fitur sensitif?
3) Bagaimana hasil evaluasi kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* dalam membatasi lingkup dan durasi akses pengguna internal sebagai penguatan terhadap model RBAC dasar pada *prototype* sistem dompet digital?

1.3 Tujuan Penelitian

Mengacu pada rumusan masalah yang telah ditetapkan, Tujuan-tujuan tersebut adalah:

1) Mengidentifikasi kebutuhan kontrol akses pengguna internal melalui pemodelan peran, data sensitif, dan fitur sistem dalam *prototype* sistem dompet digital.
2) Merancang dan mengimplementasikan prinsip *Least Privilege* dan mekanisme *Just-In-Time Access* pada backend *Access control layer* dalam membatasi serta mengatur akses terhadap data dan fitur sensitif.
3) Mengevaluasi kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* dalam membatasi lingkup dan durasi akses pengguna internal sebagai penguatan terhadap model RBAC dasar pada *prototype* sistem dompet digital.

1.4 Manfaat Penelitian

Dengan adanya penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1.4.1 Manfaat Teoritis

Penelitian ini memberikan kontribusi dalam pengembangan kajian keamanan sistem informasi, khususnya pada perancangan dan implementasi kontrol akses pengguna internal melalui penerapan prinsip *Least Privilege* dan *Just-In-Time Access* pada *layer* aplikasi. Penelitian ini juga memberikan referensi mengenai bagaimana kedua konsep tersebut dapat diterapkan secara terstruktur dalam pengelolaan hak akses pengguna internal.

1.4.2 Manfaat Praktis

1. Bagi Pengembang Sistem: Penelitian ini memberikan gambaran implementasi kontrol akses pengguna internal pada level aplikasi, khususnya dalam membatasi hak akses berdasarkan peran serta mengatur akses terhadap fitur sensitif secara terkontrol.
2. Bagi Organisasi: Penelitian ini dapat menjadi referensi dalam pengelolaan akses pengguna internal, khususnya dalam mengurangi potensi penyalahgunaan akses dengan menerapkan pembatasan lingkup dan durasi akses pada sistem yang mengelola data sensitif.
3. Bagi Peneliti Selanjutnya: Penelitian ini dapat menjadi dasar untuk pengembangan lebih lanjut terkait perancangan dan implementasi kontrol akses pada *layer* aplikasi, baik dengan pendekatan yang lebih kompleks maupun pada lingkungan sistem yang lebih luas.

1.5 Batasan Penelitian

Untuk menjaga fokus dan ketercapaian tujuan penelitian, maka penelitian ini memiliki beberapa batasan sebagai berikut:

1) Penelitian ini menggunakan *prototype* sistem dompet digital sebagai media implementasi kontrol akses pengguna internal, bukan untuk membangun sistem

dompet digital komersial lengkap, integrasi pembayaran, atau kepatuhan regulasi finansial secara lengkap.

2) Penelitian difokuskan pada penerapan *Role-Based Access Control*, *Least Privilege*, dan *Just-In-Time Access* pada *layer* kontrol akses aplikasi, khususnya backend. Penelitian tidak membahas keamanan infrastruktur seperti server, jaringan, sistem operasi, deployment production, maupun pengamanan administratif database.
3) Fokus utama aktor pengujian adalah Customer Support (CS) sebagai pengguna internal yang menangani ticket dan berinteraksi dengan data serta fitur sensitif. Role User dan Administrator digunakan sebagai aktor pendukung untuk membentuk konteks ticket, pemisahan role, dan pemantauan aktivitas sistem.
4) Pengujian difokuskan pada backend enforcement, yaitu keputusan backend dalam menerima atau menolak request berdasarkan role, assignment ticket, status ticket, session JIT, feature, dan batas waktu akses. Frontend dan komponen realtime hanya digunakan sebagai media demonstrasi *Prototype*, sedangkan evaluasi utama dilakukan melalui respons API, status akses, perubahan database, serta log sebagai bukti pendukung tambahan.
5) Hasil penelitian ini diperoleh dari lingkungan *prototype* dan tidak merepresentasikan kesiapan sistem produksi berskala level industri. Penelitian ini bertujuan memodelkan dan mengakademikkan konsep kontrol akses internal yang umumnya tertutup dalam praktik industri, serta memverifikasi penerapan RBAC, *Least Privilege*, dan *Just-In-Time Access* pada backend *Prototype*, bukan mengklaim keamanan absolut secara keseluruhan.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Beberapa penelitian terdahulu yang membahas pengendalian akses, *Least Privilege*, serta *Just-In-Time Access* telah menjadi landasan dalam perumusan penelitian ini. Penelitian-penelitian tersebut menunjukkan bahwa pembatasan hak akses dan penghapusan akses permanen memiliki peran penting dalam menekan risiko penyalahgunaan hak akses pada sistem yang melibatkan pengguna internal. Berikut ringkasan dari penelitian-penelitian terdahulu disajikan pada Tabel 2.1.

Tabel 2. 1 Penelitian Terdahulu

No	Peneliti	Judul	Kesimpulan
1.	(Rao, 2025)	Strategic Value of *Just-In-Time Access* Control	Membahas JIT *Access* pada lingkungan enterprise/IAM untuk mengurangi standing privilege dan risiko insider threat, namun belum membahas implementasi kontrol akses pada *layer* aplikasi.
2.	(Velur, 2025)	The Evolution of Enterprise Security Systems in Cloud-native Architectures: From Perimeter-Based to Automated, Granular Models	Berfokus pada keamanan cloud-native, IAM, ABAC, dan model akses granular di lingkungan terdistribusi, namun belum membahas implementasi *Least Privilege* dan JIT pada backend *Access control layer* aplikasi

3.	(Raju, 2021)	*Zero Trust* Architecture for Financial Institutions	Membahas Zero Trust dan *Least Privilege* pada lembaga keuangan untuk melindungi data sensitif, tetapi masih pada level arsitektur keamanan institusi.
4.	(Haber & Rolls, 2020)	*Just-In-Time Access* Management	Menjelaskan JIT *Access* Management untuk mengurangi akses permanen pada lingkungan IAM/PAM, namun belum membahas penerapan JIT berbasis ticket dan session pada aplikasi.
5.	(Subramanyam, 2025)	Zero Trust in Practice: Enhancing Privileged *Access* Security with *Just-In-Time* (JIT) and Self-Service Models	Membahas integrasi *Least Privilege* dan JIT dalam Zero Trust PAM untuk mempersempit attack surface, tetapi fokusnya masih pada privileged *Access* enterprise.
6.	(Prajapati, 2025)	Role of Identity and *Access* Management in *Zero Trust* Architecture for Cloud Security: Challenges and Solutions	Membahas IAM, *Least Privilege*, JIT, MFA, dan SIEM pada cloud security, namun belum membahas implementasi kontrol akses pengguna internal pada level backend dalam aplikasi.

Berdasarkan Tabel 2.1, penelitian-penelitian terdahulu menunjukkan bahwa *Least Privilege* dan *Just-In-Time Access* memiliki peran penting dalam pengelolaan hak akses yang aman. Namun, sebagian besar pembahasan masih berfokus pada level infrastruktur, seperti cloud, IAM, PAM, jaringan, server, dan lingkungan enterprise. Belum banyak penelitian yang menyajikan implementasi teknis kedua konsep tersebut

pada *layer* kontrol akses aplikasi, khususnya pada backend yang berinteraksi langsung dengan pengguna internal, data sensitif, dan fitur sensitif.

Dalam sistem berbasis peran, *Role-Based Access Control* (RBAC) umumnya digunakan untuk membagi hak akses berdasarkan peran pengguna. Namun, RBAC dasar lebih berfokus pada pemetaan siapa yang boleh mengakses fitur tertentu berdasarkan role, bukan pada seberapa minimum fitur tersebut seharusnya diberikan dan dalam konteks apa fitur tersebut boleh digunakan. Dengan demikian, RBAC tetap berpotensi menyisakan akses yang terlalu luas apabila sebuah role diberikan izin terhadap fitur sensitif tanpa pembatasan berdasarkan kebutuhan tugas, status operasional, maupun durasi akses. Kondisi inilah yang menyebabkan RBAC perlu diperkuat dengan prinsip *Least Privilege* dan mekanisme *Just-In-Time Access* agar kontrol akses tidak hanya berbasis peran, tetapi juga berbasis konteks dan waktu.

Berangkat dari celah tersebut, penelitian ini memindahkan fokus implementasi ke *layer* aplikasi melalui *prototype* sistem dompet digital. *Prototype* ini digunakan sebagai media representasi untuk menyimulasikan lingkungan backend operasional yang melibatkan Customer Support, ticket, data sensitif, dan fitur sensitif. Dalam konteks ini, RBAC digunakan sebagai model dasar untuk membentuk batas akses awal, *Least Privilege* digunakan untuk membatasi lingkup akses berdasarkan assignment ticket, sedangkan *Just-In-Time Access* digunakan untuk membatasi durasi penggunaan fitur sensitif melalui session sementara. Kombinasi ketiga konsep tersebut dirancang sebagai kontrol akses berlapis yang lebih kontekstual dibandingkan model RBAC dasar.

Meskipun industri finansial berskala besar kemungkinan telah menerapkan mekanisme keamanan tingkat lanjut dalam operasionalnya, rancangan dan implementasi teknis kontrol akses internal pada *layer* aplikasi umumnya tidak dibuka secara rinci karena berkaitan dengan strategi keamanan organisasi. Akibatnya, rujukan akademik terbuka yang membahas bagaimana *Least Privilege* dan *Just-In-Time Access* diterapkan secara teknis pada backend aplikasi masih terbatas. Oleh karena itu, penelitian ini berupaya memodelkan konsep kontrol akses internal yang relevan dengan kebutuhan sistem sensitif ke dalam bentuk *prototype* akademik yang dapat dirancang, diimplementasikan, dan diuji secara terbuka. Dengan demikian, posisi

penelitian ini bukan untuk membangun sistem dompet digital komersial secara utuh atau mengklaim kesetaraan dengan keamanan standar industri, melainkan untuk menyediakan gambaran implementatif mengenai kontrol akses berlapis pada backend aplikasi. Evaluasi dilakukan melalui pengujian skenario fungsional untuk menunjukkan bagaimana kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* dapat membatasi lingkup dan durasi akses pengguna internal sebagai penguatan terhadap model RBAC dasar pada *Prototype* sistem dompet digital.

2.2 Sistem Dompet Digital

Sistem dompet digital (*e-wallet*) merupakan salah satu bentuk inovasi teknologi finansial (*fintech*) yang menyediakan layanan penyimpanan instrumen pembayaran, penampungan dana, serta pelaksanaan transaksi keuangan secara elektronik (Kraiwanit dkk., 2024). Secara arsitektur perangkat lunak, sistem ini umumnya terbagi menjadi dua sisi utama, yaitu sisi pengguna akhir yang merepresentasikan interaksi pengguna dengan akun, serta sisi operasional internal yang digunakan untuk penanganan bantuan teknis dan pengelolaan akun. Tingginya volume transaksi dan kebutuhan integrasi secara *real-time* menuntut dompet digital untuk memiliki infrastruktur aplikasi yang andal sekaligus tata kelola operasional yang terstruktur di belakang layar.

Sebagai platform yang menjembatani aktivitas finansial, sistem dompet digital mengelola berbagai klasifikasi data yang bersifat sangat sensitif. Data tersebut mencakup Informasi Identitas Pribadi (*Personally Identifiable Information* / PII) seperti nomor induk kependudukan dan kontak pengguna, serta data rekam jejak finansial berupa informasi saldo, riwayat mutasi, dan rincian transaksi (Mohamed dkk., 2026). Karakteristik data yang memiliki nilai tinggi ini menjadikan sistem dompet digital sebagai entitas yang membutuhkan perlindungan ketat, khususnya pada tata kelola akses terhadap data di dalam ruang lingkup *layer* aplikasi itu sendiri.

Dalam operasional sehari-hari, pemeliharaan dan pengelolaan ekosistem dompet digital ini sangat bergantung pada interaksi dari pengguna internal, seperti administrator sistem, staf operasional, tim audit, maupun layanan teknis pengguna (*customer support*) (Deliu & Moldovan, 2025). Para pengguna internal ini membutuhkan akses ke dalam *layer* aplikasi *back-end* untuk menjalankan tugas

fungsional yang sah, seperti melakukan verifikasi akun (*Know Your Customer* / KYC), menangani keluhan transaksi pengguna, hingga memantau laporan keuangan (Gupta dkk., 2023). Interaksi intensif antara pengguna internal dengan fitur-fitur sensitif dalam aplikasi inilah yang menuntut adanya konsep kontrol akses yang presisi.

Oleh karena itu, dalam konteks penelitian ini, *Prototype* sistem dompet digital digunakan sebagai media implementasi yang sangat representatif. *Prototype* ini tidak dikembangkan untuk memenuhi fungsionalitas bisnis finansial secara menyeluruh, melainkan dirancang khusus untuk mensimulasikan lingkungan operasional *back-end* beserta fitur-fitur administratifnya. Melalui *Prototype* inilah konsep pembatasan hak akses internal dapat diterapkan pada *layer* aplikasi, sehingga pemodelan peran, data sensitif, dan fitur sistem dapat dievaluasi secara terukur dan relevan.

2.3 Kontrol Akses

Kontrol akses merupakan dasar keamanan yang krusial dalam sistem informasi untuk mengatur siapa saja yang diizinkan berinteraksi dengan sumber daya tertentu (Ragothaman dkk., 2023). Secara mendasar, kontrol akses bertujuan untuk menjamin integritas dan kerahasiaan data dengan cara membatasi akses hanya kepada pengguna yang memiliki hak sah. Dalam ekosistem perangkat lunak, kontrol akses bekerja sebagai filter yang menentukan apakah sebuah permintaan (*request*) dari pengguna dapat diproses atau ditolak berdasarkan aturan keamanan yang telah ditetapkan sebelumnya (Park dkk., 2025).

Secara teknis, proses kontrol akses melibatkan tiga tahapan utama, yaitu identifikasi, autentikasi, dan otorisasi. Identifikasi merupakan tahap awal di mana pengguna menyatakan identitasnya kepada sistem (seperti penggunaan *username*). Autentikasi adalah proses verifikasi untuk membuktikan keabsahan identitas tersebut (misalnya melalui kata sandi). Setelah identitas terverifikasi, tahap otorisasi menentukan cakupan tindakan atau data apa saja yang boleh diakses oleh pengguna tersebut. Penelitian ini menitikberatkan pada tahap otorisasi, di mana pembatasan hak akses dilakukan secara lebih spesifik pada fungsionalitas di dalam aplikasi (Kumar, 2025).

Terdapat beberapa model kontrol akses yang umum diterapkan dalam pengembangan sistem informasi. Beberapa di antaranya adalah *Discretionary Access Control* (DAC) yang memberikan wewenang penuh kepada pemilik data untuk mengatur akses, serta *Mandatory Access Control* (MAC) yang menggunakan klasifikasi keamanan ketat dari pusat sistem. Selain itu, terdapat pula *Attribute-Based Access Control* (ABAC) yang memberikan akses berdasarkan karakteristik atau atribut pengguna dan lingkungan (Farhadighalati dkk., 2025). Namun, di antara berbagai model tersebut, *Role-Based Access Control* (RBAC) menjadi salah satu yang paling luas digunakan karena kemudahannya dalam manajemen hak akses berdasarkan struktur organisasi atau peran fungsional.

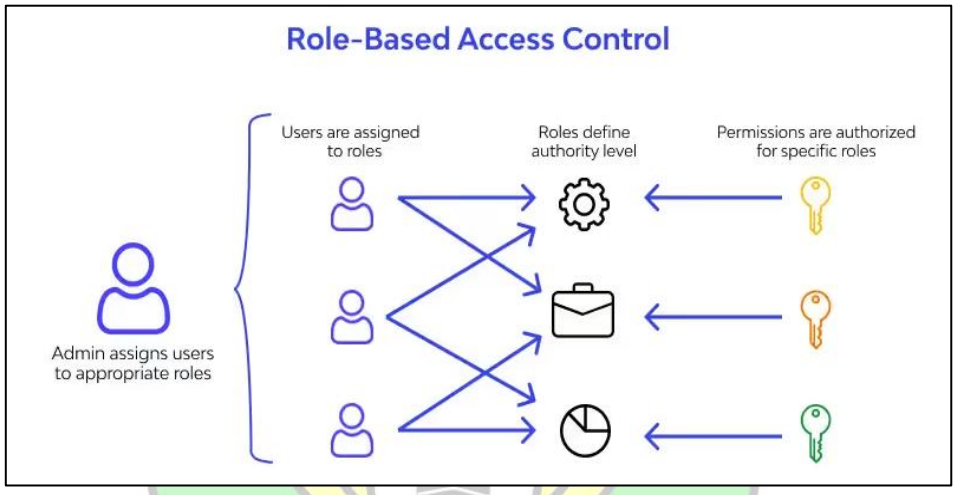
Implementasi kontrol akses pada *layer* aplikasi menjadi fokus utama dalam penelitian ini guna melindungi data sensitif secara lebih kontekstual. Berbeda dengan kontrol akses pada *layer* jaringan atau infrastruktur, kontrol akses aplikasi memungkinkan pengaturan yang lebih detail terhadap fitur-fitur spesifik di dalam sistem, seperti akses ke tombol transaksi atau tampilan data mutasi keuangan. Dengan menerapkan kontrol akses yang tepat di level aplikasi, sistem dapat memastikan bahwa setiap aktivitas operasional yang dilakukan oleh pengguna internal tetap berada dalam batasan teknis yang telah ditentukan oleh kebijakan organisasi.

2.4 *Role-Based Access Control* (RBAC)

Role-Based Access Control (RBAC) merupakan model pengontrolan akses yang berpusat pada penetapan wewenang berdasarkan peran (role) pengguna di dalam suatu organisasi, bukan berdasarkan identitas individu secara langsung. Dalam model ini, hak akses atau privilege dikaitkan dengan peran tertentu, dan pengguna kemudian ditugaskan pada peran yang sesuai dengan tanggung jawab serta fungsionalitas pekerjaan mereka (Waheed dkk., 2025). Pendekatan ini sangat efisien diterapkan pada aplikasi skala menengah hingga besar, seperti operasional back-end sistem dompet digital, karena menyederhanakan proses pengelolaan keamanan ketika terjadi rotasi staf atau perubahan tugas administratif.

Secara struktural, model dasar RBAC beroperasi melalui interaksi tiga elemen utama: Pengguna (*User*), Peran (*Role*), dan Izin Akses (*Permission*). *User* merepresentasikan entitas manusia yang berinteraksi dengan sistem, sedangkan *Role*

mewakili fungsi pekerjaan atau level otoritas tertentu, misalnya peran sebagai "Auditor" atau "Customer Support". Sementara itu, *Permission* adalah persetujuan teknis untuk mengeksekusi operasi tertentu terhadap data, seperti hak untuk membaca riwayat transaksi atau menyetujui penarikan dana (Atlam & Yang, 2025). Dengan memisahkan penugasan *User-to-Role* dan *Role-to-Permission*, sistem dapat mengelompokkan hak akses secara terpusat tanpa harus mengonfigurasi setiap pengguna secara individual. Ilustrasi gambaran umum RBAC dasar dapat dilihat pada gambar 2.1.



Gambar 2. 1 RBAC dasar
Sumber : IBS Digital College

Meskipun RBAC menawarkan kemudahan dalam administrasi dan skalabilitas, model dasar ini memiliki keterbatasan krusial ketika diimplementasikan pada lingkungan yang mengelola data sangat sensitif. Kelemahan RBAC terletak pada sifatnya yang statis dan berpotensi menghasilkan *standing privileges*, yaitu kondisi di mana hak akses selalu melekat pada pengguna secara permanen (Ebad & Amara, 2026). Oleh karena itu, RBAC digunakan sebagai fondasi pemisahan akses berdasarkan role, kemudian diperkuat dengan *Least Privilege* dan *Just-In-Time Access* untuk menambahkan pembatasan berdasarkan konteks tugas dan durasi akses.

Sifat hak akses statis yang melekat secara terus-menerus ini memperbesar celah kerentanan terhadap penyalahgunaan wewenang sah oleh pengguna internal. Oleh karena itu, dalam perancangan kontrol akses aplikasi yang modern, implementasi RBAC murni dinilai tidak lagi mencukupi (Bates dkk., 2025). Penelitian ini memilih

RBAC sebagai fondasi karena RBAC merupakan salah satu model kontrol akses yang paling umum digunakan dalam praktik sistem informasi untuk memisahkan hak akses berdasarkan peran pengguna. Struktur tersebut sesuai dengan kebutuhan *Prototype*, karena aktor sistem dapat dipetakan secara jelas menjadi User, Customer Support, dan Administrator. Fokus penelitian ini bukan mengganti RBAC dengan model lain, melainkan menunjukkan bahwa RBAC dasar masih dapat diperkuat melalui pembatasan tambahan berbasis *Least Privilege* dan *Just-In-Time Access*. ABAC memang dapat memberikan kontrol akses yang lebih granular karena mempertimbangkan atribut pengguna, objek, tindakan, dan kondisi lingkungan, tetapi penerapannya membutuhkan definisi atribut serta kebijakan akses yang lebih kompleks dan tidak selalu tersedia dalam *Prototype*. Oleh karena itu, penelitian ini tetap menggunakan RBAC sebagai fondasi awal, kemudian menambahkan pembatasan berdasarkan assignment ticket, status ticket, fitur sensitif, dan masa aktif session JIT agar akses internal tidak hanya bergantung pada role, tetapi juga pada konteks tugas dan durasi akses.

2.5 *Least Privilege*

Prinsip *Least Privilege* (LP) adalah landasan keamanan informasi yang menetapkan bahwa setiap pengguna, proses, maupun sistem hanya boleh diberikan hak akses seminimal mungkin yang benar-benar diperlukan untuk menyelesaikan suatu tugas spesifik (Plachkinova & Knapp, 2023). Prinsip ini bertujuan untuk membatasi ruang lingkup kerusakan yang dapat terjadi, baik akibat kesalahan manusia (*human error*) maupun penyalahgunaan akses yang sah. Dalam implementasi pada *layer* aplikasi, *Least Privilege* menuntut pembatasan akses yang sangat granular, di mana hak akses tidak lagi diberikan secara luas berdasarkan posisi jabatan, melainkan berdasarkan kebutuhan fungsional paling dasar dari sebuah instruksi kerja.

Sering kali terdapat kerancuan yang menganggap bahwa penerapan *Role-Based Access Control* (RBAC) secara otomatis telah memenuhi prinsip *Least Privilege*. Namun, secara fundamental keduanya memiliki entitas yang berbeda, RBAC adalah sebuah konsep administratif untuk mengelompokkan hak akses, sedangkan *Least Privilege* adalah prinsip keamanan yang mengatur kedalaman dan batasan dari hak akses tersebut (Olorunlana, 2025). Tanpa penerapan *Least Privilege*

yang ketat, model RBAC cenderung bersifat terlalu luas (*over-privileged*), di mana sebuah peran (*role*) diberikan sekumpulan izin akses yang mencakup seluruh fungsi dalam suatu departemen, padahal pengguna mungkin hanya membutuhkan satu atau dua fungsi saja untuk menyelesaikan tugas hariannya (Plachkinova & Knapp, 2023).

Perbedaan krusial lainnya terletak pada tingkat granularitas kontrolnya. Dalam RBAC tradisional, kontrol akses sering kali berhenti pada level "siapa boleh mengakses fitur apa". Sementara itu, *Least Privilege* masuk lebih dalam ke level "data apa yang boleh diakses dan tindakan minimal apa yang diizinkan" (Rose dkk., 2020). Sebagai contoh, dalam sistem dompet digital, seorang staf layanan teknis pengguna mungkin diberikan peran (*role*) untuk melihat data transaksi pengguna. Namun, berdasarkan prinsip LP, staf tersebut hanya diizinkan melihat data transaksi yang sedang dikeluhkan oleh pengguna, bukan seluruh riwayat transaksi global, dan hanya memiliki izin baca (*read-only*) tanpa izin untuk mengubah data tersebut.

Penerapan prinsip *Least Privilege* pada *layer* aplikasi dalam penelitian ini berfungsi untuk membedah kembali peran-peran statis dalam RBAC menjadi izin akses yang lebih mikro dan spesifik. Dengan membatasi hak akses pada level fungsionalitas yang paling rendah, potensi paparan data sensitif dapat diminimalisir. Namun, meskipun LP mampu membatasi lingkup akses, prinsip ini tetap menyisakan celah jika akses tersebut bersifat permanen. Oleh karena itu, prinsip *Least Privilege* perlu dikolaborasikan dengan mekanisme kontrol akses dinamis berbasis waktu guna mencapai tingkat keamanan yang lebih komprehensif pada sistem dompet digital.

2.6 *Just-In-Time Access*

Just-In-Time (JIT) *Access* adalah mekanisme kontrol akses dinamis yang memberikan hak akses kepada pengguna hanya pada saat dibutuhkan dan untuk jangka waktu yang terbatas (Behringer & Baumann, 2025). Prinsip utama dari JIT adalah penghapusan *standing privileges*, yaitu kondisi di mana pengguna memiliki hak akses administratif atau sensitif yang selalu aktif setiap saat (Olaide dkk., 2025). Dengan menerapkan JIT, hak akses tidak lagi bersifat permanen, melainkan diberikan secara temporer berdasarkan permintaan atau pemicu tertentu, dan akan dicabut secara otomatis setelah tugas selesai atau durasi waktu yang ditetapkan telah berakhir.

Mekanisme JIT bekerja dengan cara meminimalkan jendela kerentanan pada sistem. Dalam operasional aplikasi, seorang pengguna internal mungkin memiliki peran yang sah, namun ia tidak selalu membutuhkan seluruh hak akses administratif tersebut dalam aktivitas rutinnya. Melalui JIT, sistem menyediakan jalur di mana pengguna harus melewati proses verifikasi atau persetujuan tambahan sebelum mendapatkan peningkatan hak akses sementara. Hal ini memastikan bahwa hak akses tingkat tinggi hanya ada pada durasi yang sangat singkat, sehingga secara signifikan memperkecil ruang lingkup yang dapat disalahgunakan.

Dalam konteks *layer* aplikasi dompet digital, implementasi JIT memungkinkan pengelolaan akses terhadap fitur sensitif dilakukan secara lebih kontekstual. Sebagai contoh, ketika seorang staf layanan teknis pengguna perlu mengakses data transaksi detail untuk menangani keluhan pengguna, sistem akan memberikan akses tersebut hanya untuk durasi waktu tertentu, misalnya 15 menit. Setelah waktu tersebut habis, *session* atau izin akses tersebut akan kedaluwarsa secara otomatis di level aplikasi. Pendekatan ini memberikan perlindungan tambahan dibandingkan kontrol akses tradisional yang membiarkan akun administratif memiliki akses penuh selama staf tersebut masih berstatus karyawan aktif.

Integrasi antara JIT dengan prinsip *Least Privilege* dan model RBAC membentuk mekanisme kontrol akses berlapis pada *layer* kontrol akses pengguna internal. RBAC berperan sebagai fondasi awal untuk memisahkan akses berdasarkan role, *Least Privilege* membatasi ruang lingkup akses agar sesuai dengan kebutuhan minimum, sedangkan JIT membatasi durasi penggunaan fitur sensitif melalui session sementara. Dengan kombinasi tersebut, akses pengguna internal terhadap data dan fitur sensitif tidak hanya ditentukan oleh role, tetapi juga oleh konteks tugas dan waktu penggunaan akses.

Pendekatan pembatasan akses berbasis konteks tersebut juga sejalan dengan prinsip *Zero Trust*. (Rose dkk., 2020) melalui NIST SP 800-207 menjelaskan bahwa akses terhadap sumber daya tidak cukup hanya didasarkan pada lokasi jaringan atau keberhasilan autentikasi awal, tetapi perlu divalidasi secara berkelanjutan berdasarkan identitas, konteks, dan kebijakan akses yang berlaku. Prinsip ini relevan dengan penelitian ini karena akses pengguna internal tidak hanya dibatasi berdasarkan role

melalui RBAC, tetapi juga diperiksa berdasarkan *assignment ticket*, status *ticket*, fitur yang diminta, serta masa aktif *session* JIT. Dengan demikian, RBAC digunakan sebagai fondasi awal, sedangkan *Least Privilege* dan *Just-In-Time Access* digunakan sebagai penguatan agar kontrol akses pada backend aplikasi menjadi lebih kontekstual, terbatas, dan tidak bersifat permanen.

2.7 Relevansi pada Sistem Dompet Digital

Sistem dompet digital dipilih sebagai lingkungan implementasi karena memiliki karakteristik yang relevan dengan permasalahan kontrol akses pengguna internal. Sistem seperti ini mengelola data sensitif, seperti data identitas pengguna, informasi akun, saldo, riwayat aktivitas, serta data *Know Your Customer* (KYC). Dalam operasionalnya, pengguna internal seperti *Customer Support* (CS) dapat memiliki kebutuhan sah untuk mengakses data atau menjalankan fitur tertentu dalam rangka menangani permasalahan pengguna. Namun, akses internal tersebut tetap berisiko apabila hanya dibatasi berdasarkan role tanpa memperhatikan konteks tugas dan durasi penggunaan akses.

Dalam penelitian ini, *Prototype* sistem dompet digital digunakan untuk merepresentasikan lingkungan backend operasional yang melibatkan pengguna internal, *ticket*, data sensitif, dan fitur sensitif. RBAC digunakan sebagai model dasar untuk memisahkan akses berdasarkan role. Namun, RBAC dasar masih berpotensi menghasilkan akses yang terlalu luas apabila tidak diperkuat dengan pembatasan yang lebih kontekstual. Oleh karena itu, penelitian ini menambahkan prinsip *Least Privilege* dan mekanisme *Just-In-Time Access* sebagai penguatan terhadap RBAC dasar. *Least Privilege* digunakan untuk membatasi akses CS berdasarkan *assignment ticket*, sedangkan *Just-In-Time Access* digunakan untuk membatasi penggunaan fitur sensitif melalui *session* sementara.

Perbandingan antara RBAC dasar dan RBAC yang diperkuat dengan *Least Privilege* serta *Just-In-Time Access* ditunjukkan pada Tabel 2.2. Perbandingan ini digunakan sebagai evaluasi untuk menilai apakah konsep yang dirancang mampu membatasi akses berdasarkan lingkup tugas dan durasi akses.

Tabel 2. 2 Perbandingan Model Kontrol Akses

No	Aspek	RBAC Dasar	RBAC dengan LP & JIT
1.	Lingkup akses	Luas berdasarkan role	Dibatasi sesuai assignment ticket
2.	Status akses	Cenderung melekat pada role	Aktif sesuai kebutuhan operasional
3.	Akses data sensitif	Berpotensi tersedia selama role memiliki izin	Terbatas pada konteks ticket yang valid
4.	Aktivasi fitur sensitif	Dapat aktif jika akses diberikan pada role	Memerlukan session JIT sementara
5.	Durasi akses	Mengikuti *session* login atau izin role	Dibatasi oleh masa aktif session JIT
6.	Potensi paparan akses	Permission dalam role berpotensi tetap selalu tersedia lebih luas dari kebutuhan tugas.	Paparan akses dibatasi berdasarkan assignment ticket dan masa aktif session

Berdasarkan perbandingan tersebut, penelitian ini menempatkan *Least Privilege* dan *Just-In-Time Access* sebagai mekanisme penguatan terhadap RBAC dasar dalam mengurangi risiko akses internal yang terlalu luas dan permanen. Evaluasi dilakukan untuk memverifikasi apakah penguatan tersebut berjalan pada backend.

BAB III
METODE PENELITIAN

3.1 Pendekatan Penelitian

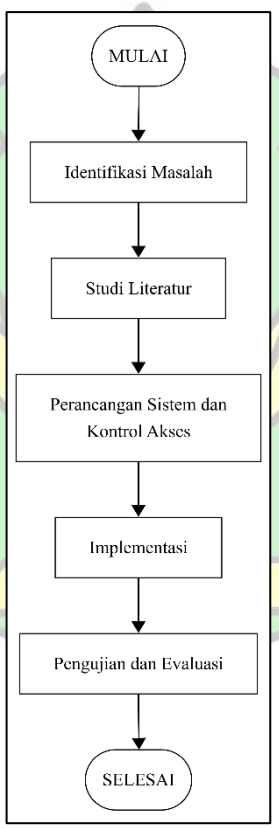
Penelitian ini merupakan penelitian terapan yang berfokus pada perancangan, implementasi, dan evaluasi mekanisme kontrol akses pengguna internal pada *layer* aplikasi. Pendekatan yang digunakan adalah berbasis *Prototype*, yaitu dengan membangun representasi lingkungan operasional backend dalam *Prototype* sistem dompet digital sebagai media untuk menerapkan dan menguji mekanisme kontrol akses. *Prototype* ini tidak diarahkan untuk membangun sistem dompet digital komersial secara lengkap, melainkan digunakan sebagai lingkungan terkontrol untuk memverifikasi mekanisme kontrol akses yang dirancang.

Fokus penelitian berada pada backend *Access control layer* dalam mengatur akses pengguna internal terhadap data dan fitur sensitif. Mekanisme yang diterapkan merupakan kombinasi *Role-Based Access Control* (RBAC), *Least Privilege*, dan *Just-In-Time Access*. RBAC digunakan sebagai batas awal berdasarkan role, *Least Privilege* digunakan untuk membatasi lingkup akses berdasarkan assignment ticket, sedangkan *Just-In-Time Access* digunakan untuk membatasi durasi penggunaan fitur sensitif melalui session sementara. Konteks role Customer Support (CS) dipilih karena role ini merepresentasikan pengguna internal yang memiliki kebutuhan operasional untuk menangani ticket pengguna dan berpotensi berinteraksi dengan data sensitif, seperti data identitas, status akun, dan fitur bantuan akun dalam level aplikasi. Dengan pendekatan ini, *Prototype* dompet digital digunakan sebagai media terkontrol untuk mengevaluasi bagaimana kombinasi mekanisme kontrol akses tersebut dapat memperkuat model RBAC dasar pada backend aplikasi.

Evaluasi penelitian dilakukan melalui pengujian skenario operasional Customer Support. Pengujian difokuskan pada backend enforcement, validasi hak akses, pembatasan lingkup akses, serta pemberian dan pencabutan akses sementara terhadap fitur sensitif. Hasil pengujian digunakan untuk menilai apakah mekanisme kontrol akses berjalan sesuai rancangan dalam membatasi lingkup dan durasi akses pengguna internal.

3.2 Alur Penelitian

Untuk menjaga agar proses penelitian berjalan secara sistematis, penelitian ini dilakukan melalui beberapa tahapan yang saling berkaitan, mulai dari identifikasi masalah hingga pengujian dan evaluasi mekanisme yang telah diimplementasikan. Alur penelitian disusun untuk memastikan bahwa perancangan kontrol akses tetap sesuai dengan fokus penelitian, yaitu penerapan RBAC sebagai dasar, *Least Privilege* sebagai pembatas lingkup akses, dan *Just-In-Time Access* sebagai pembatas akses sementara pada backend *Access control layer* dalam *Prototype* sistem dompet digital. Alur tahapan penelitian dapat dilihat pada Gambar 3.1.



Gambar 3. 1 Alur Penelitian

Adapun rincian dari masing-masing tahapan pada Gambar 3.1 adalah sebagai berikut:

1. Identifikasi Masalah : Tahap ini dilakukan untuk mengidentifikasi permasalahan utama yang berkaitan dengan risiko penyalahgunaan akses oleh pengguna internal. Permasalahan tersebut muncul karena pengguna internal, seperti

Customer Support, memiliki akses sah terhadap sistem, namun akses tersebut berpotensi menjadi terlalu luas apabila tidak dibatasi berdasarkan role, konteks tugas, dan kebutuhan operasional.

2. Studi Literatur : Tahap ini dilakukan dengan mengkaji referensi yang berkaitan dengan kontrol akses, *Role-Based Access Control*, *Least Privilege*, *Just-In-Time Access*, insider threat, serta penerapan kontrol akses pada sistem yang mengelola data sensitif. Studi literatur digunakan sebagai dasar untuk merumuskan posisi penelitian dan menentukan pendekatan yang sesuai dalam merancang mekanisme kontrol akses pada *layer* aplikasi.

3. Perancangan Sistem dan Kontrol Akses **:** Tahap ini mencakup perancangan mekanisme kontrol akses pada backend *Prototype*. Perancangan dilakukan dengan memetakan aktor sistem, data sensitif, fitur sensitif, serta alur operasional Customer Support berbasis ticket. Pada tahap ini juga dirancang RBAC sebagai kontrol dasar, *Least Privilege* sebagai pembatas lingkup akses, dan *Just-In-Time Access* sebagai pembatas durasi akses terhadap fitur sensitif.

4. Implementasi Mekanisme : Tahap ini dilakukan dengan menerapkan rancangan kontrol akses ke dalam backend *Prototype* menggunakan route, middleware, database, serta log pendukung. Implementasi difokuskan pada backend enforcement agar pembatasan akses tidak hanya bergantung pada tampilan antarmuka, tetapi benar-benar divalidasi pada sisi server.

5. Pengujian dan Evaluasi : Tahap akhir dilakukan melalui pengujian skenario untuk memverifikasi mekanisme RBAC, *Least Privilege*, dan *Just-In-Time Access*. Pengujian difokuskan pada akses sesuai role, penolakan akses lintas role, pembatasan akses berdasarkan assignment ticket, serta pemberian dan pencabutan session JIT pada fitur sensitif. Hasil pengujian dianalisis berdasarkan respons API, status akses, perubahan data pada database, serta log pendukung untuk memastikan bahwa mekanisme berjalan sesuai rancangan.

3.3 Analisis Kebutuhan Kontrol Akses

Analisis kebutuhan kontrol akses dilakukan untuk mengidentifikasi aktor sistem, data sensitif, fitur sensitif, serta konteks operasional yang menjadi dasar

penerapan *Role-Based Access Control* (RBAC), *Least Privilege*, dan *Just-In-Time Access*. Analisis ini difokuskan pada *layer* kontrol akses aplikasi, khususnya pada alur bantuan pengguna yang melibatkan Customer Support (CS) sebagai pengguna internal utama.

Prototype sistem dompet digital pada penelitian ini tidak dirancang untuk membangun keseluruhan proses bisnis finansial secara lengkap. *Prototype* digunakan sebagai media representasi untuk menggambarkan bagaimana pengguna internal dapat berinteraksi dengan data dan fitur sensitif melalui mekanisme ticket. Oleh karena itu, kebutuhan kontrol akses difokuskan pada pembatasan akses berdasarkan role, assignment ticket, status penanganan, dan durasi akses terhadap fitur sensitif.

3.3.1 Identifikasi Aktor Sistem

Aktor dalam *Prototype* sistem ini terdiri dari User, Customer Support (CS), dan Administrator. User berperan sebagai pengguna akhir yang membuat ticket dan menjadi pemilik data yang dilindungi. Customer Support (CS) menjadi aktor utama penelitian karena memiliki kebutuhan operasional untuk menangani ticket serta berpotensi berinteraksi dengan data dan fitur sensitif pengguna. Administrator juga termasuk pengguna internal, namun tidak menjadi aktor utama dalam pengujian *Least Privilege* dan *Just-In-Time Access*. Pada penelitian ini, Administrator berperan sebagai pengelola sistem dan pemantau aktivitas melalui audit log serta terminal log. Dengan pembagian aktor tersebut, penelitian dapat memisahkan peran antara pemilik data, pelaksana operasional, dan pemantau sistem. Identifikasi peran yang digunakan dalam batasan *Prototype* ini diuraikan pada Tabel 3.1.

Tabel 3. 1 Pengidentifikasian Peran dalam Sistem

No	Role	kategori	Peran dalam Sistem	Keterlibatan dalam Penelitian
1.	Customer Support (CS)	Pengguna internal	Menangani ticket bantuan pengguna dan menjalankan proses bantuan akun sesuai konteks ticket yang ditugaskan.	Aktor utama pengujian *Least Privilege* dan *Just-In-Time Access* karena berinteraksi langsung dengan data serta fitur sensitif.
2.	Administrator	Pengguna internal	Mengelola akun sistem serta memantau aktivitas melalui audit log dan terminal log.	Pendukung, sebagai aktor observability dan audit, bukan pelaksana utama akses JIT.
3.	User	Pengguna akhir	Membuat ticket bantuan dan menjadi pemilik data yang dilindungi, seperti data akun, status akun, dan data KYC.	Pendukung, sebagai sumber konteks ticket dan pemilik data sensitif.

Berdasarkan identifikasi tersebut, Customer Support (CS) dipilih sebagai fokus utama karena role ini memiliki akses operasional yang sah, tetapi tetap berisiko apabila tidak dibatasi berdasarkan konteks tugas. Oleh karena itu, *Least Privilege* digunakan untuk membatasi ruang lingkup akses CS berdasarkan assignment ticket, sedangkan *Just-In-Time Access* digunakan untuk membatasi penggunaan fitur sensitif agar hanya aktif sementara ketika dibutuhkan.

3.3.2 Identifikasi Data dan Fitur Sensitif

Dalam *Prototype* sistem dompet digital, data dan fitur tertentu dikategorikan sebagai sensitif karena berkaitan dengan identitas pengguna, status akun, serta tindakan perubahan pada akun pengguna. Data dan fitur tersebut menjadi objek utama yang perlu dibatasi aksesnya melalui mekanisme *Least*

Privilege dan *Just-In-Time Access*. Pemetaan data dan fitur sensitif yang digunakan dalam penelitian ini disajikan pada Tabel 3.2.

Tabel 3. 2 Pemetaan Data dan Fitur Sensitif

No	Objek Sensitif	Jenis	Alasan Pembatasan Akses	Mekanisme Kontrol
1.	Data KYC / VIEW_KYC	Data dan fitur sensitif	Berisi informasi identitas pengguna yang tidak boleh diakses secara bebas oleh pengguna internal.	LP + JIT
2.	Nomor telepon	Data sensitif	Termasuk data pribadi pengguna	LP
3.	Saldo / informasi akun	Data sensitif	Berkaitan dengan kondisi akun pengguna dan hanya relevan dalam konteks ticket tertentu.	LP
4.	RESET_PASSWORD	Fitur sensitif	Mengubah kredensial akun pengguna sehingga tidak boleh aktif secara default.	JIT
5.	CHANGE_EMAIL	Fitur sensitif	Mengubah identitas akses akun dan berpotensi disalahgunakan apabila terbuka permanen.	JIT
6.	RESET_PIN	Fitur sensitif	Berkaitan dengan kredensial keamanan akun pengguna.	JIT
7.	UNBLOCK_ACCOUNT	Fitur sensitif	Mengubah status keamanan akun pengguna.	JIT

Berdasarkan identifikasi tersebut, data sensitif dibatasi melalui prinsip *Least Privilege* agar hanya dapat diakses dalam konteks ticket yang sah. Sementara itu, fitur sensitif dikendalikan melalui *Just-In-Time Access* agar tidak aktif secara permanen dan hanya dapat digunakan sementara ketika dibutuhkan dalam proses penanganan ticket.

3.3.3 Konteks Operasional *Prototype*

Penerapan *Least Privilege* dan *Just-In-Time Access* pada penelitian ini disesuaikan dengan konteks operasional Customer Support (CS) dalam *Prototype* sistem dompet digital. Dalam alur ini, User membuat ticket bantuan, kemudian Customer Support menangani ticket tersebut sesuai assignment yang diberikan oleh sistem.

Konteks ticket digunakan sebagai dasar pembatasan akses karena setiap tindakan CS harus memiliki hubungan langsung dengan permasalahan pengguna yang sedang ditangani. Dengan demikian, CS tidak diberikan akses global terhadap seluruh data pengguna, melainkan hanya dapat mengakses data yang relevan dengan ticket yang ditugaskan kepadanya. Adapun fitur sensitif tetap membutuhkan session JIT yang valid sebelum dapat dijalankan. Pendekatan ini digunakan untuk merepresentasikan praktik kontrol akses internal yang umumnya tertutup di lingkungan industri ke dalam bentuk *Prototype* akademik, sehingga mekanisme pembatasan akses dapat diuji dan dianalisis secara terbuka.

3.4 Perancangan Kontrol Akses

Perancangan kontrol akses pada penelitian ini difokuskan pada backend *Access control layer* yang berfungsi sebagai lapisan validasi terhadap setiap request yang masuk ke sistem. Konsep ini dirancang agar akses pengguna internal, khususnya Customer Support (CS), tidak hanya ditentukan oleh role, tetapi juga oleh konteks ticket dan status akses sementara terhadap fitur sensitif. Pendekatan yang digunakan bersifat berlapis, terdiri dari:

1. *Role-Based Access Control* (RBAC) sebagai validasi dasar berdasarkan role pengguna.
2. *Least Privilege* (LP) sebagai pembatas ruang lingkup.
3. *Just-In-Time* (JIT) *Access* sebagai kontrol pemberian dan pembatasan durasi akses fitur sensitif.

Ketiga lapisan tersebut diterapkan pada backend melalui middleware dan validasi usecase, sehingga pembatasan akses tidak hanya bergantung pada tampilan antarmuka, tetapi benar-benar ditegakkan pada sisi server.

3.4.1 Perancangan *Role-Based Access Control* (RBAC)

RBAC dirancang sebagai lapisan awal otorisasi untuk membatasi area akses berdasarkan role pengguna. Pada *Prototype* ini, terdapat tiga role utama, yaitu User, Customer Support (CS), dan Administrator. User berperan sebagai pembuat ticket dan pemilik data, Customer Support berperan sebagai pengguna internal yang menangani ticket bantuan, sedangkan Administrator berperan dalam pengelolaan dan pemantauan aktivitas sistem. Pemetaan dasar role dalam sistem dirancang pada Tabel 3.3.

Tabel 3. 3 Perancangan *Role-Based Access Control* (RBAC)

No	Role	Hak Akses Dasar	Batasan Akses
1.	Customer Support (CS)	Melihat ticket yang tersedia, melakukan claim ticket, menangani ticket yang ditugaskan, dan mengajukan JIT untuk fitur sensitif.	Tidak dapat mengakses panel Administrator dan tidak memiliki akses global ke seluruh data pengguna.
2.	Administrator	Melihat audit log, terminal log, dashboard sistem, serta mengelola akun tertentu.	Tidak menjadi pelaksana utama dalam alur JIT pada ticket pengguna.
3.	User	Membuat ticket, melihat ticket miliknya, mengirim pesan, dan menutup ticket yang sudah selesai.	Tidak dapat mengakses area CS dan Administrator.

RBAC pada penelitian ini berfungsi sebagai fondasi awal otorisasi untuk memisahkan area akses berdasarkan role. Namun, RBAC belum cukup untuk membatasi akses secara lebih spesifik karena role CS tetap membutuhkan pembatasan berdasarkan konteks ticket dan durasi akses fitur sensitif. Oleh karena

itu, RBAC diperkuat dengan *Least Privilege* sebagai pembatas ruang lingkup akses dan *Just-In-Time Access* sebagai pembatas akses sementara terhadap fitur sensitif.

3.4.2 Perancangan *Least Privilege* (LP)

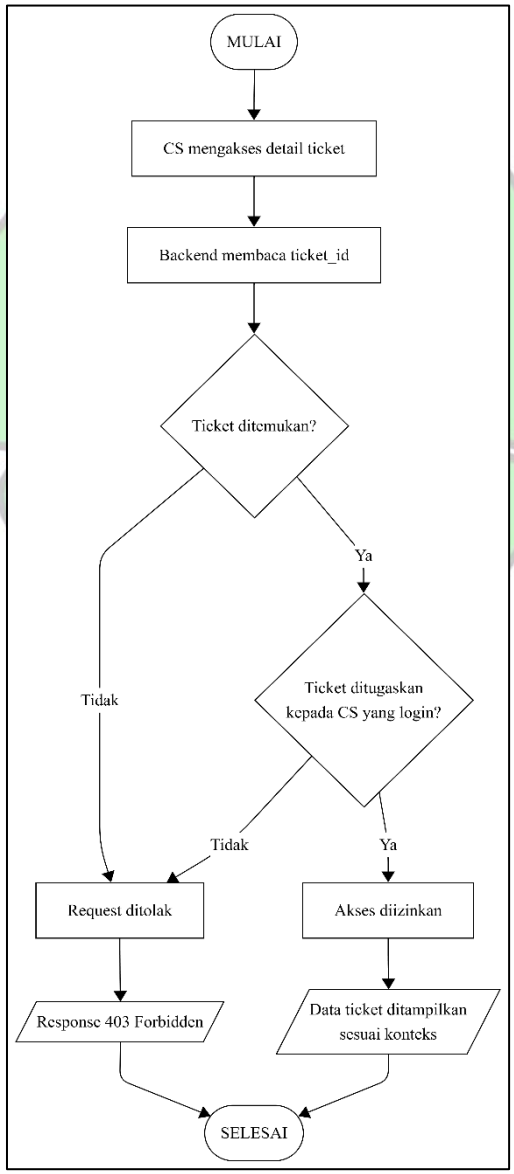
Least Privilege dirancang untuk membatasi ruang lingkup akses *Customer Support* (CS) agar hanya memiliki akses minimum yang diperlukan dalam proses penanganan *ticket*. Pada penelitian ini, LP tidak dimaknai sebagai penghapusan seluruh fungsi CS, melainkan sebagai pembatasan agar fitur dan data yang dapat diakses CS tetap sesuai dengan kebutuhan operasionalnya.

Penerapan LP pada sistem dilakukan melalui pembatasan akses berbasis ticket. Setelah CS login dan melewati validasi RBAC, sistem tidak menyediakan akses global untuk mencari atau membuka seluruh data pengguna secara bebas. CS hanya diarahkan pada area kerja ticket, dapat melihat ticket yang tersedia, melakukan claim ticket, dan membuka detail ticket yang telah ditugaskan kepadanya. Dengan demikian, akses CS tidak melekat secara luas hanya karena role yang dimilikinya, tetapi tetap dibatasi oleh konteks. Pada alur operasional tersebut, proses claim ticket digunakan untuk menetapkan bahwa sebuah ticket telah diambil dan ditugaskan kepada CS tertentu. Status CLAIMED menunjukkan tahap penugasan awal, sedangkan status IN_PROGRESS menunjukkan bahwa ticket sudah mulai ditangani secara aktif. Perbedaan status ini penting karena mekanisme JIT tidak diberikan hanya karena ticket telah di-claim, tetapi hanya dapat diajukan ketika ticket tersebut telah ditugaskan kepada CS yang bersangkutan dan berada pada status IN_PROGRESS. Dalam konteks penelitian ini, LP diterapkan melalui beberapa aturan berikut:

1. CS tidak memiliki akses global terhadap seluruh data pengguna.
2. CS hanya dapat mengakses detail ticket yang ditugaskan kepadanya.
3. Data pengguna yang ditampilkan tetap dibatasi sesuai kebutuhan penanganan ticket.

4. Fitur sensitif tidak dibuka oleh LP, melainkan tetap dikunci dan hanya dapat digunakan melalui mekanisme JIT.

Pada rancangan *Least Privilege*, proses validasi difokuskan pada pembatasan ruang lingkup akses berdasarkan assignment ticket. Backend memeriksa ticket_id yang dikirimkan pada request, kemudian mencocokkannya dengan data penugasan ticket pada database. Apabila ticket tidak ditemukan, belum berada dalam konteks penugasan yang sah, atau ticket tersebut ditugaskan kepada CS lain, maka request ditolak oleh backend. Alur validasi *Least Privilege* dapat dilihat pada Gambar 3.2



Gambar 3. 2 Mekanisme Alur validasi *Least Privilege*

Berdasarkan alur validasi pada Gambar 3.2, *Least Privilege* berperan sebagai pembatas ruang lingkup akses CS. Backend tidak hanya memeriksa role CS, tetapi juga memastikan bahwa ticket yang diakses benar-benar berada dalam konteks penugasan yang sah. CS tetap dapat menjalankan tugas operasional seperti melihat, melakukan claim, dan menangani ticket, tetapi tidak memiliki akses global terhadap seluruh data pengguna. Dengan rancangan ini, akses CS terhadap data pengguna hanya diberikan ketika terdapat hubungan yang sah antara CS dan ticket yang ditugaskan, sedangkan fitur sensitif tetap dikendalikan melalui mekanisme *Just-In-Time Access*.

3.4.3 Perancangan *Just-In-Time* (JIT) *Access*

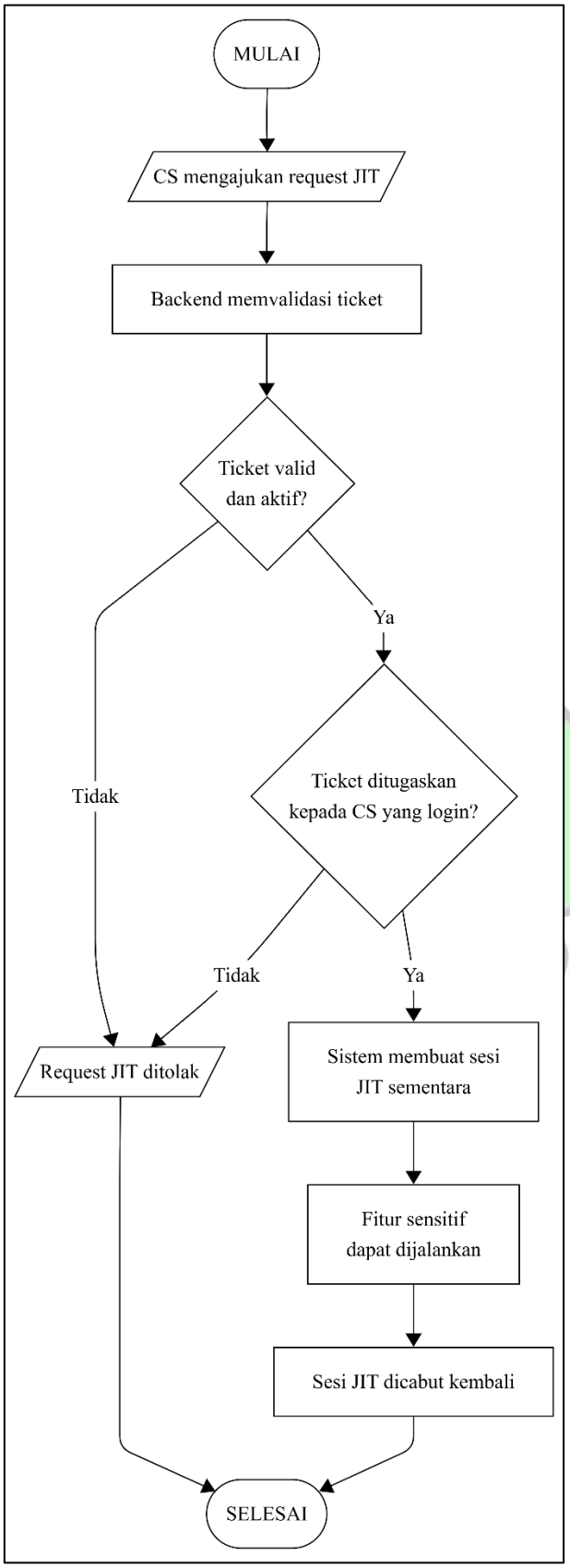
Just-In-Time Access dirancang sebagai lapisan tambahan setelah *Least Privilege*. Jika *Least Privilege* membatasi ruang lingkup akses Customer Support (CS), maka *Just-In-Time Access* membatasi waktu dan kondisi penggunaan fitur sensitif. Dengan demikian, JIT tidak menggantikan *Least Privilege*, tetapi bekerja di atas batasan *Least Privilege* untuk memastikan bahwa fitur sensitif hanya dapat digunakan sementara dan tetap berada dalam konteks ticket yang sah. Fitur seperti VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT tidak aktif secara default. Meskipun CS telah memiliki role yang sah dan ticket telah ditugaskan kepadanya, CS tetap tidak dapat menjalankan fitur sensitif sebelum memiliki session JIT yang valid.

Pada penelitian ini, pemberian *session* JIT dilakukan melalui validasi otomatis pada backend berdasarkan aturan yang telah ditentukan (Contextual Auto-Approval). Dengan pendekatan ini, *session* JIT hanya dapat dibuat apabila CS memenuhi seluruh konteks akses yang disyaratkan. Sebelum *session* JIT diberikan, backend akan melakukan validasi sebagai berikut :

1. ticket_id harus valid dan tersedia pada database.
2. *Ticket* harus ditugaskan kepada CS yang mengajukan request JIT.
3. ticket harus berada pada status IN_PROGRESS.

4. Feature yang diminta harus termasuk dalam daftar fitur sensitif yang dikendalikan melalui mekanisme JIT.
5. *Session* JIT aktif sebelumnya untuk kombinasi CS, ticket, dan feature yang sama dinonaktifkan sebelum session baru dibuat.

Jika seluruh validasi terpenuhi, sistem membuat *session* JIT sementara pada tabel jit_sessions. *Session* tersebut menyimpan informasi CS, *ticket*, fitur sensitif yang diminta, status aktif, serta waktu kedaluwarsa akses. Ketika fitur sensitif dijalankan, backend memeriksa bahwa feature pada endpoint yang diakses sesuai dengan feature yang tersimpan pada session JIT. Pada rancangan ini, *session* JIT berlaku selama 15 menit dan hanya dapat digunakan satu kali. Durasi 15 menit dipilih sebagai batas waktu pada *Prototype* untuk menunjukkan bahwa akses terhadap fitur sensitif tidak diberikan secara permanen. Angka tersebut tidak dimaksudkan sebagai standar baku, melainkan sebagai parameter pengujian agar mekanisme kedaluwarsa akses dapat dirancang, diterapkan, dan diverifikasi. Dalam penerapan nyata, durasi *session* JIT dapat disesuaikan dengan kebijakan organisasi, tingkat sensitivitas fitur, dan kebutuhan penanganan *ticket*. Alur validasi *Just-In-Time Access* dapat dilihat pada Gambar 3.3.



Gambar 3.3 Mekanisme Alur validasi *Just-In-Time Access*

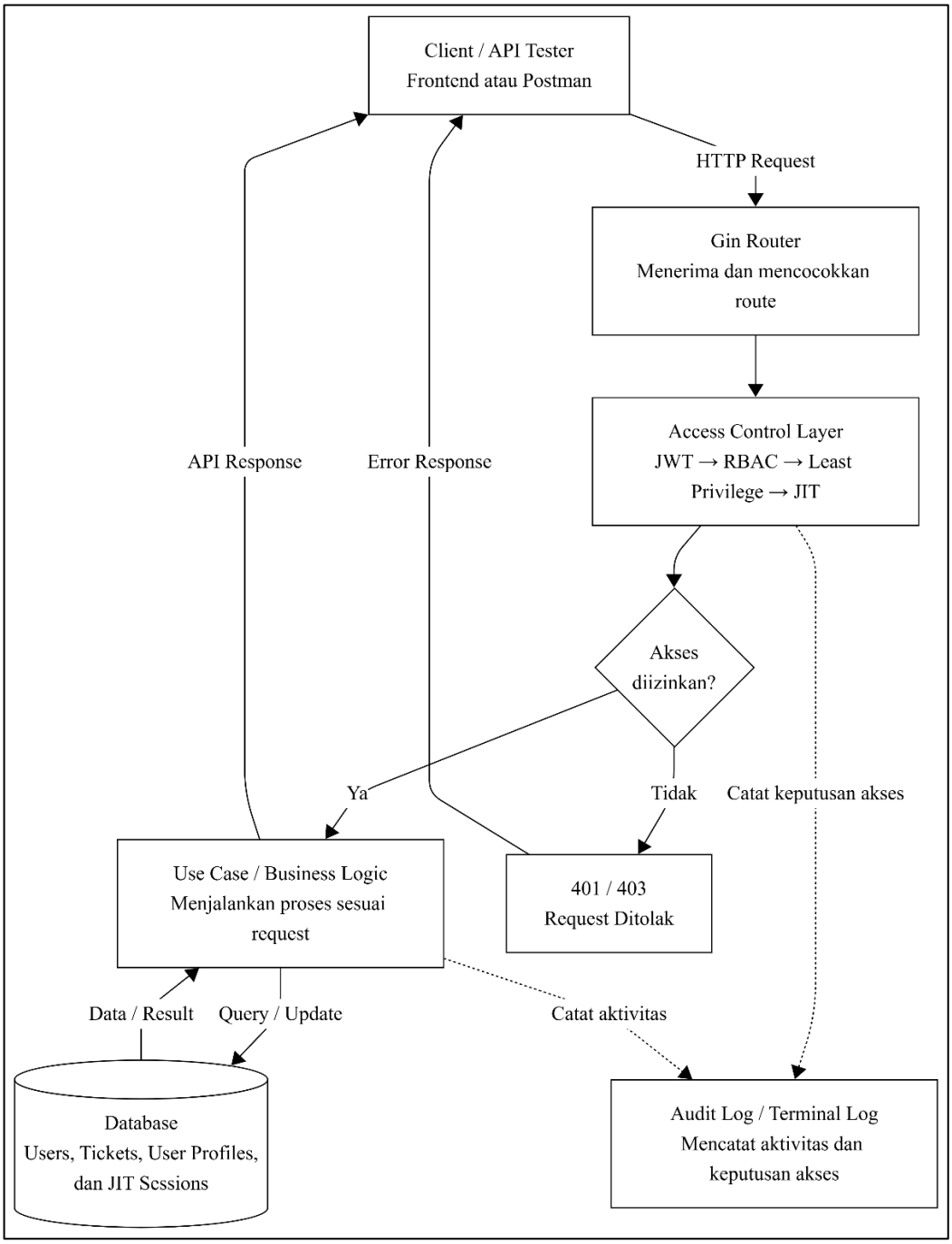
Pencabutan akses JIT dilakukan apabila waktu akses telah kedaluwarsa, fitur sensitif telah berhasil dijalankan, *ticket* ditutup, atau CS melakukan *logout*. Dengan rancangan ini, akses sensitif tidak melekat secara permanen pada role CS. JIT hanya memberikan akses sementara terhadap fitur tertentu, pada *ticket* tertentu, dan untuk CS yang memenuhi validasi backend.

Berdasarkan rancangan tersebut, *Least Privilege* dan *Just-In-Time Access* memiliki fungsi yang berbeda namun saling melengkapi. *Least Privilege* membatasi ruang lingkup akses CS berdasarkan konteks *ticket*, sedangkan *Just-In-Time Access* membatasi waktu penggunaan fitur sensitif melalui *session* sementara. Kombinasi keduanya digunakan untuk memperkuat RBAC dasar agar akses pengguna internal tidak hanya dibatasi berdasarkan role, tetapi juga berdasarkan konteks tugas dan masa berlaku akses.

3.5 Arsitektur Kontrol Akses pada Backend

Arsitektur sistem pada penelitian ini dirancang dengan menempatkan mekanisme kontrol akses pada sisi backend sebagai lapisan utama validasi request. Pendekatan ini digunakan agar pembatasan akses tidak hanya bergantung pada tampilan antarmuka, tetapi benar-benar ditegakkan pada sisi server melalui proses autentikasi, otorisasi role, validasi assignment ticket, dan validasi session JIT.

Secara umum, alur arsitektur backend pada *Prototype* ini dimulai dari request yang dikirim oleh client, baik melalui frontend maupun API tester seperti Postman. Request tersebut diterima oleh Gin Router, kemudian diteruskan ke *Access control layer* untuk divalidasi sebelum masuk ke logika aplikasi dan database. Hasil validasi dan proses sistem kemudian dikembalikan dalam bentuk response API. Arsitektur kontrol akses pada backend dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Arsitektur Kontrol Akses pada Backend

Komponen utama dalam arsitektur backend ini adalah sebagai berikut:

1. Client / API Tester : Client merupakan sumber request yang dapat berasal dari frontend atau Postman. Frontend digunakan sebagai media demonstrasi alur

sistem, sedangkan pengujian utama difokuskan pada respons backend dan validasi akses melalui API.

2. Gin Router : Gin Router berfungsi menerima request dan mengarahkannya ke endpoint yang sesuai. Pada tahap ini, route dikelompokkan berdasarkan area akses, seperti User, Customer Support, Administrator, serta endpoint fitur sensitif.

3. Middleware Kontrol Akses: *Access control layer* merupakan lapisan utama validasi akses pada backend. Lapisan ini mencakup validasi JWT untuk memastikan identitas pengguna, RBAC untuk memeriksa kesesuaian role, *Least Privilege* untuk memeriksa assignment ticket, serta JIT middleware untuk memastikan bahwa fitur sensitif hanya dapat dijalankan ketika session JIT valid.

4. Logika Aplikasi : Setelah request melewati *Access control layer*, request diteruskan ke logika aplikasi yang menjalankan proses sesuai fungsi sistem, seperti claim ticket, membuka detail ticket, membuat session JIT, atau menjalankan fitur sensitif. Pada bagian ini, beberapa validasi konteks tetap dapat dilakukan kembali sebagai penguatan terhadap validasi middleware.

5. Basis Data (*Database*) : Database digunakan untuk menyimpan data utama *Prototype*, seperti users, user_profiles, tickets, messages, jit_sessions, dan audit_logs. Data pada tabel tickets digunakan sebagai dasar validasi assignment ticket, sedangkan tabel jit_sessions digunakan untuk menyimpan session akses sementara pada mekanisme JIT.

6. Audit Log dan Terminal Log : Audit log digunakan untuk mencatat aktivitas penting seperti request JIT, penolakan akses, eksekusi fitur sensitif.

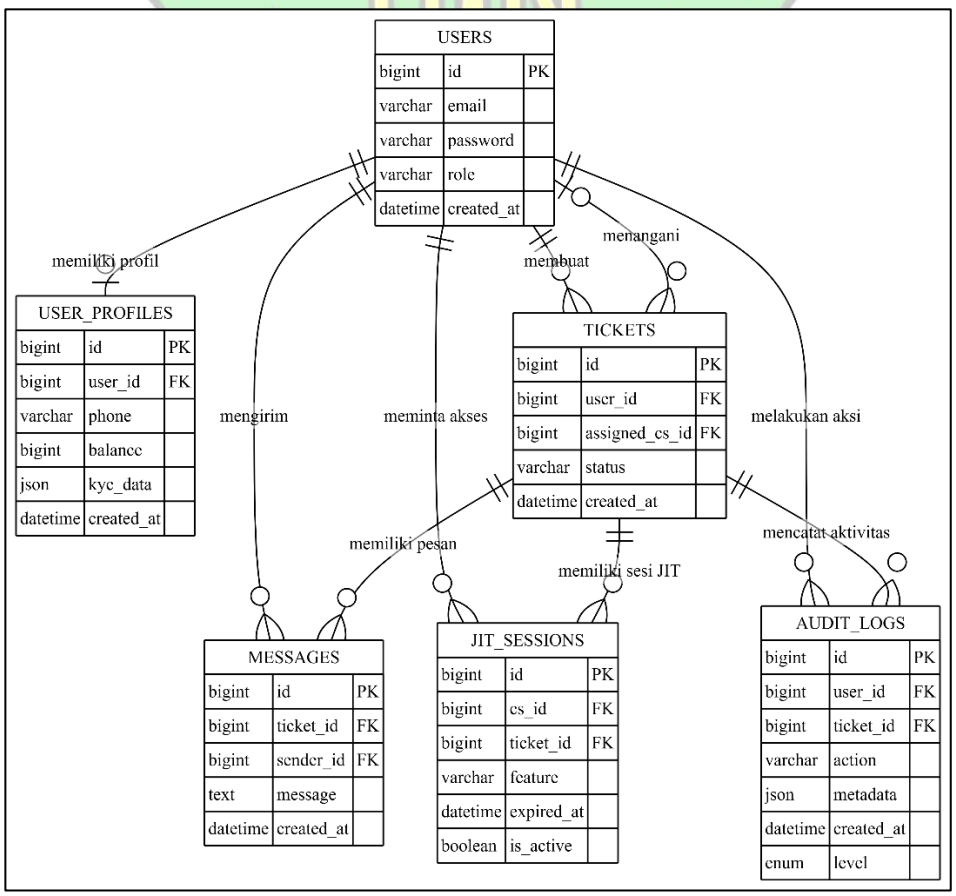
Dan untuk memperjelas komponen backend yang digunakan dalam implementasi dan pengujian, dilakukan pemetaan terhadap endpoint utama yang berhubungan dengan autentikasi, ticket, *Least Privilege*, *Just-In-Time Access*, audit log, dan terminal log. Pemetaan gambaran besar endpoint tersebut dapat dilihat pada Tabel 3.4

Tabel 3. 4 Pemetaan Rancangan Endpoint Penting

No	Aksi / Endpoint	Role	Fungsi	Mekanisme Kontrol
1.	POST /auth/login	Semua	Memberikan token JWT sesuai role	Autentikasi
2.	GET /cs/tickets/open	CS	Melihat ticket pool tersedia	RBAC
3.	POST /cs/tickets/:ticket_id/claim	CS	Mengambil ticket yang tersedia	RBAC + LP
4.	GET /cs/tickets/:ticket_id	CS	Membuka detail ticket	RBAC + LP
5.	GET /cs/tickets/:ticket_id/user/profile	CS	Membuka ringkasan data pengguna yang terkait dengan ticket	RBAC + LP
6.	POST /cs/tickets/:ticket_id/jit/request	CS	Mengajukan session JIT untuk fitur sensitif tertentu	RBAC + LP + JIT
7.	POST /cs/tickets/:ticket_id/sensitive/view-kyc	CS	Membuka data KYC pengguna dalam konteks ticket	JIT
8.	POST /cs/tickets/:ticket_id/sensitive/reset-password	CS	Melakukan reset password pengguna	JIT
9.	POST /cs/tickets/:ticket_id/sensitive/change-email	CS	Mengubah email pengguna	JIT
10.	POST /cs/tickets/:ticket_id/sensitive/reset-pin	CS	Melakukan reset PIN pengguna	JIT
11.	POST /cs/tickets/:ticket_id/sensitive/unblock-account	CS	Membuka blokir akun pengguna	JIT
12	GET /admin/audit-logs	Administrator	Melihat riwayat aktivitas penting pada sistem	RBAC

Berdasarkan pemetaan tersebut, endpoint pada *Prototype* tidak hanya berfungsi sebagai jalur akses fitur, tetapi juga menjadi titik penerapan kontrol akses. Endpoint area CS terlebih dahulu dibatasi melalui RBAC, kemudian akses terhadap ticket diperketat melalui *Least Privilege* berdasarkan assignment ticket. Sementara itu, endpoint fitur sensitif hanya dapat dijalankan apabila session JIT yang sesuai masih aktif dan valid. Dengan demikian, setiap request tetap melewati validasi backend sebelum diizinkan mengakses data atau menjalankan fitur sensitif.

Selain pemetaan endpoint, sistem juga dirancang dengan struktur database yang mendukung penerapan kontrol akses berbasis role, assignment ticket, dan session *Just-In-Time Access*. Struktur database memuat beberapa entitas utama, yaitu users, user_profiles, tickets, messages, jit_sessions, dan audit_logs. Relasi antar entitas digunakan untuk menghubungkan pengguna, data profil, ticket bantuan, pesan, session akses sementara, serta catatan aktivitas sistem. Rancangan Entity Relationship Diagram (ERD) database dapat dilihat pada Gambar 3.5.



Gambar 3. 5 Rancangan Entity Relationship Diagram (ERD) Database

Berdasarkan ERD tersebut, tabel users menyimpan identitas dan role pengguna sebagai dasar RBAC, tabel tickets menghubungkan User sebagai pembuat ticket dengan Customer Support melalui assigned_cs_id sebagai dasar *Least Privilege*, sedangkan tabel jit_sessions menyimpan session sementara yang mengikat CS, ticket, feature, status aktif, dan waktu kedaluwarsa sebagai dasar *Just-In-Time Access*. Adapun tabel audit_logs digunakan sebagai komponen pendukung untuk mencatat aktivitas penting pada sistem. Dengan struktur ini, pembatasan akses pada backend dapat diterapkan secara berlapis berdasarkan role, assignment ticket, feature, dan durasi akses.

3.6 Lingkungan Implementasi

Lingkungan implementasi digunakan untuk mendukung proses pengembangan, pengujian, dan demonstrasi *Prototype* sistem. Karena fokus penelitian berada pada backend *Access control layer*, lingkungan implementasi disusun untuk menjalankan backend API, database, frontend sebagai media demonstrasi, serta tools pengujian API.

3.6.1 Kebutuhan Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan dalam penelitian ini berfungsi untuk menjalankan proses pengembangan sistem, server lokal, database, dan pengujian secara lokal. Spesifikasi perangkat keras yang digunakan dapat dilihat pada Tabel 3.5 berikut:

Tabel 3.5 Kebutuhan Perangkat Keras (*Hardware*)

No	Komponen	Spesifikasi
1.	Prosesor	AMD Ryzen 7 5700
2.	Memori (RAM)	24 GB
3.	Penyimpanan	SSD (Minimal 512 GB)
4.	Perangkat Tambahan	Laptop/PC dengan koneksi internet

Spesifikasi tersebut digunakan untuk menjalankan backend server, database MySQL, serta tools pengujian API secara lokal.

3.6.2 Kebutuhan Perangkat Lunak (*Software*) dan Teknologi

Perangkat lunak yang digunakan dalam penelitian ini dipilih untuk mendukung implementasi backend, penyimpanan data, komunikasi realtime, serta pengujian mekanisme kontrol akses. Rincian perangkat lunak dan teknologi yang digunakan dapat dilihat pada Tabel 3.6.

Tabel 3.6 Kebutuhan Perangkat Lunak (*Software*) dan Teknologi

No	Komponen	Teknologi	Fungsi	Alasan Pemilihan
1.	Backend API	Golang Gin	Membangun endpoint, routing, middleware RBAC, *Least Privilege*, dan JIT *Access*.	Ringan dan mendukung middleware.
2.	Database	MySQL	Menyimpan data users, tickets, user_profiles, jit_sessions, messages, dan audit_logs.	Cocok untuk data terstruktur.
3.	Autentikasi	JWT	Membawa identitas dan role pengguna pada setiap request.	Praktis untuk validasi *session* akses.
4.	Realtime Socket	Gorilla WebSocket	Mendukung fitur realtime seperti live chat atau tampilan log secara langsung	Menyesuaikan kebutuhan *Prototype*, bukan fokus utama pengujian.
5.	Pengujian API	Postman	Menguji respons backend pada konteks akses valid, akses ditolak, dan akses JIT.	Memudahkan pengujian API.
6.	Frontend Demo	Next.js	Menampilkan alur *Prototype* dan membantu demonstrasi area kerja	Mendukung visualisasi alur sistem sebagai bukti pendukung

7.	Pengembangan	Visual Studio Code, Git	Mendukung penulisan kode dan pengelolaan versi kode sumber.	Mendukung proses pengembangan.

Dengan komponen tersebut, mekanisme RBAC, *Least Privilege*, dan *Just-In-Time Access* dapat diuji melalui request API, status akses, perubahan data pada database, session JIT, serta log sebagai bukti pendukung.

3.7 Pengujian dan Metode Evaluasi

Pengujian pada penelitian ini dilakukan menggunakan metode *scenario-based testing* dengan pendekatan *black-box testing*. Pengujian difokuskan pada mekanisme kontrol akses yang menjadi inti penelitian, yaitu *Role-Based Access Control*, *Least Privilege*, dan *Just-In-Time Access*. Pengujian tidak diarahkan untuk menguji seluruh fitur *Prototype* dompet digital secara *end-to-end*, melainkan untuk memverifikasi bahwa aturan akses yang telah dirancang benar-benar dijalankan pada backend.

Pengujian ini menekankan aspek *backend enforcement*. Artinya, pembatasan akses tidak hanya dinilai dari ada atau tidaknya tombol pada frontend. Meskipun antarmuka tidak menyediakan tombol tertentu atau menyembunyikan fitur tertentu, backend tetap harus menolak request yang tidak sah apabila endpoint dipanggil langsung melalui Postman atau API client.

Evaluasi dilakukan dengan membandingkan hasil yang diharapkan dan hasil aktual pada setiap skenario pengujian. Indikator evaluasi yang digunakan adalah sebagai berikut:

1. Respons API yang diberikan oleh backend.
2. Status akses, yaitu request diterima atau ditolak.
3. Perubahan state pada database.
4. Audit log dan terminal log sebagai bukti pendukung.
5. Kesesuaian hasil aktual terhadap rancangan mekanisme kontrol akses.

Dengan pendekatan tersebut, pengujian pada penelitian ini bersifat *verification-based*, yaitu digunakan untuk memverifikasi kesesuaian implementasi terhadap rancangan, bukan untuk mengukur performa atau mengklaim keamanan absolut sistem.

3.7.1 Cakupan dan Fokus Pengujian

Cakupan pengujian pada penelitian ini dibatasi pada mekanisme kontrol akses backend. Unit evaluasi utama adalah keputusan backend dalam menerima atau menolak request berdasarkan aturan akses yang telah dirancang. Fokus pengujian ditetapkan sebagai berikut:

1. Pengujian difokuskan pada mekanisme kontrol akses backend, bukan pada pengujian seluruh proses bisnis *Prototype* dompet digital.
2. Pengujian tidak diarahkan untuk menilai performa sistem, usability, penetration testing, ataupun kesiapan sistem produksi berskala industri.
3. Pengujian dilakukan untuk memverifikasi keputusan backend dalam menerima atau menolak request.
4. Parameter validasi akses yang diuji meliputi role pengguna, assignment ticket, status ticket, session JIT, feature yang diminta, dan batas waktu akses.
5. Frontend digunakan sebagai media pendukung pengujian, sedangkan evaluasi utama tetap dilakukan melalui request API, database, dan log pendukung.

Skenario pengujian disusun berdasarkan tiga lapisan kontrol akses sebagai berikut:

1. *Role-Based Access Control* digunakan untuk menguji pemisahan area akses dasar antar-role, yaitu User, Customer Support, dan Administrator.
2. *Least Privilege* digunakan untuk menguji pembatasan ruang lingkup akses Customer Support berdasarkan assignment ticket.
3. *Just-In-Time Access* digunakan untuk menguji pembatasan akses fitur sensitif melalui session sementara yang valid, sesuai feature, masih aktif, belum digunakan, dan belum melewati batas waktu akses.

Fitur sensitif yang dikendalikan melalui *Just-In-Time Access* meliputi VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT. Seluruh fitur tersebut menggunakan pola validasi akses yang sama pada backend, yaitu pemeriksaan role, assignment ticket, status ticket, session JIT, feature, dan waktu kedaluwarsa. Oleh karena itu, pengujian eksekusi fitur sensitif dilakukan menggunakan CHANGE_EMAIL sebagai endpoint representatif. Pemilihan CHANGE_EMAIL sebagai endpoint representatif tidak dimaksudkan untuk menyatakan bahwa seluruh logika bisnis fitur sensitif diuji secara penuh. Pengujian ini digunakan untuk memverifikasi bahwa lapisan kontrol akses JIT bekerja sebelum fitur sensitif dijalankan. Dengan demikian, fokus pengujian berada pada validasi akses terhadap fitur sensitif, bukan pada perbandingan masing-masing fitur.

Pengukuran keberhasilan pada penelitian ini tidak berupa pengukuran kuantitatif seperti latency, throughput, atau beban sistem. Keberhasilan ditentukan berdasarkan kesesuaian antara expected result dan actual result pada setiap skenario. Hasil pengujian kemudian dirangkum dalam matriks pengujian pada Bab IV agar hubungan antara skenario, hasil aktual, dan bukti pengujian dapat ditelusuri secara sistematis.

3.7.2 Pengujian *Role-Based Access Control*

Pengujian *Role-Based Access Control* dilakukan untuk memverifikasi bahwa backend mampu membatasi area akses dasar berdasarkan role pengguna. Pada tahap ini, RBAC ditempatkan sebagai lapisan awal otorisasi untuk memastikan bahwa User, Customer Support, dan Administrator hanya dapat mengakses area yang sesuai dengan perannya. Pengujian RBAC pada penelitian ini tidak diarahkan untuk menguji seluruh kombinasi role terhadap seluruh endpoint aplikasi. Endpoint yang digunakan dipilih sebagai endpoint representatif dari area User, Customer Support, dan Administrator. Dengan demikian, pengujian difokuskan pada keputusan backend dalam menerima akses sesuai role dan menolak akses lintas role yang tidak sah. Pengujian RBAC meliputi:

1. Semua Pengguna mengakses endpoint yang sesuai dengan role masing-masing. Sistem diharapkan menerima request karena role yang digunakan sesuai dengan area akses yang dituju.
2. Customer Support mencoba mengakses endpoint Administrator. Sistem diharapkan menolak request karena Customer Support tidak memiliki kewenangan pada area administratif.
3. Administrator mencoba mengakses endpoint operasional Customer Support. Sistem diharapkan menolak request karena Administrator pada *Prototype* ini diposisikan sebagai pengelola dan pemantau sistem, bukan pelaksana penanganan ticket pengguna.
4. User mencoba mengakses endpoint internal yang direpresentasikan melalui endpoint Customer Support atau Administrator. Sistem diharapkan menolak request karena User merupakan pengguna akhir dan tidak memiliki kewenangan untuk mengakses area internal.

Pengujian RBAC pada bagian ini digunakan untuk memastikan bahwa batas awal antar-role telah diterapkan pada backend. Pembuktian pada tahap ini tidak dimaksudkan sebagai pengujian kombinasi menyeluruh terhadap seluruh endpoint setiap role, melainkan sebagai verifikasi bahwa akses sesuai role diterima dan akses lintas role ditolak. Pembatasan yang lebih spesifik, seperti akses berdasarkan assignment ticket dan penggunaan fitur sensitif secara sementara, diuji secara terpisah melalui *Least Privilege* dan *Just-In-Time Access*.

3.7.3 Pengujian *Least Privilege*

Pengujian *Least Privilege* dilakukan untuk memverifikasi bahwa Customer Support tidak memiliki akses global terhadap seluruh ticket, data pengguna, maupun fitur administratif sistem. Pada tahap ini, pengujian difokuskan pada pembatasan ruang lingkup akses Customer Support berdasarkan konteks assignment ticket. Dengan demikian, role Customer Support saja tidak cukup untuk membuka seluruh data, karena backend tetap memeriksa hubungan antara Customer Support yang sedang login dan ticket yang diakses. Pengujian *Least Privilege* pada penelitian ini tidak diarahkan untuk menguji seluruh fitur

operasional Customer Support, melainkan untuk memastikan bahwa akses Customer Support tetap berada pada ruang kerja yang relevan dengan tugas penanganan ticket. Fitur sensitif seperti VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT tidak menjadi objek utama pembuktian *Least Privilege*, karena aktivasi fitur tersebut diuji secara khusus melalui mekanisme *Just-In-Time Access*. Pada tahap *Least Privilege*, fokus pengujian berada pada pembatasan lingkup akses sebelum fitur sensitif dapat digunakan. Pengujian *Least Privilege* meliputi:

1. Customer Support login dan hanya memperoleh akses ke area kerja ticket. Sistem tidak menyediakan akses global untuk melihat seluruh user, data pengguna atau fitur sensitif.
2. Customer Support melakukan claim terhadap ticket yang masih tersedia. Sistem diharapkan berhasil menerima request dan menghubungkan ticket tersebut dengan Customer Support tersebut.
3. Customer Support membuka detail ticket yang telah ditugaskan kepadanya. Sistem diharapkan menerima request karena ticket berada dalam konteks penugasan yang sah.
4. Customer Support mencoba membuka ticket yang telah ditugaskan (assigned) kepada Customer Support lain. Sistem diharapkan menolak request karena ticket berada di luar konteks penugasan Customer Support tersebut.
5. Customer Support mencoba melakukan claim terhadap ticket yang telah ditugaskan kepada Customer Support lain. Sistem diharapkan menolak request dengan respons 409 Conflict karena assignment ticket telah terbentuk.

Pengujian *Least Privilege* pada bagian ini digunakan untuk memastikan bahwa akses Customer Support dibatasi berdasarkan konteks tugas, bukan hanya berdasarkan role. Pembuktian pada tahap ini menunjukkan bahwa Customer Support tetap dapat menjalankan tugas operasional yang sah, seperti melihat area ticket, melakukan claim, dan membuka ticket miliknya, tetapi tidak memiliki akses bebas terhadap seluruh ticket atau data pengguna. Pembatasan terhadap aktivasi fitur sensitif kemudian diuji secara terpisah pada mekanisme *Just-In-Time Access*.

3.7.4 Pengujian *Just-In-Time Access*

Pengujian *Just-In-Time Access* dilakukan untuk memverifikasi bahwa akses terhadap fitur sensitif tidak diberikan secara permanen kepada Customer Support. Pada tahap ini, sistem diuji untuk memastikan bahwa fitur sensitif hanya dapat dijalankan apabila backend berhasil memvalidasi role, assignment ticket, status ticket, session JIT, feature yang diminta, serta batas waktu akses. Fitur sensitif yang digunakan sebagai endpoint representatif dalam pengujian ini adalah CHANGE_EMAIL. Endpoint tersebut dipilih karena termasuk fitur sensitif dan melewati pola validasi JIT yang sama dengan VIEW_KYC, RESET_PASSWORD, RESET_PIN, dan UNBLOCK_ACCOUNT. Dengan demikian, pengujian difokuskan pada perilaku kontrol akses JIT sebelum fitur sensitif dijalankan, bukan pada perbandingan logika bisnis dari masing-masing fitur sensitif. Pengujian *Just-In-Time Access* meliputi:

1. Customer Support mengajukan request JIT untuk fitur sensitif CHANGE_EMAIL pada ticket yang valid, telah ditugaskan kepadanya, dan berada pada status IN_PROGRESS. Sistem diharapkan menerima request dan membuat session JIT sementara.
2. Customer Support menjalankan fitur sensitif CHANGE_EMAIL ketika session JIT masih aktif, sesuai dengan CS yang sedang login, sesuai ticket, sesuai feature, belum digunakan, dan belum melewati batas waktu akses. Sistem diharapkan menerima request dan menjalankan fitur tersebut.
3. Customer Support mencoba menjalankan fitur sensitif CHANGE_EMAIL tanpa session JIT yang valid. Sistem diharapkan menolak request.
4. Customer Support mencoba menjalankan kembali fitur sensitif CHANGE_EMAIL menggunakan session JIT yang sudah digunakan atau sudah tidak aktif. Sistem diharapkan menolak request karena session JIT tidak dapat digunakan kembali.
5. Customer Support mencoba menjalankan fitur sensitif CHANGE_EMAIL setelah session JIT melewati nilai expired_at. Pengujian ini dilakukan dengan menunggu durasi session berakhir atau menyimulasikan nilai expired_at

menjadi waktu yang telah lewat. Sistem diharapkan menolak request karena session JIT sudah kedaluwarsa.

6. Customer Support mencoba menggunakan session JIT untuk feature yang berbeda dari feature yang diminta saat session dibuat. Misalnya, session JIT dibuat untuk CHANGE_EMAIL, tetapi digunakan untuk RESET_PASSWORD. Sistem diharapkan menolak request karena session JIT hanya berlaku untuk feature tertentu.
7. Customer Support mencoba menjalankan fitur sensitif setelah konteks ticket tidak lagi valid. Pada skenario ini, session JIT dibuat ketika ticket masih berstatus IN_PROGRESS, kemudian ticket berubah menjadi CLOSED, lalu Customer Support mencoba menjalankan fitur sensitif kembali. Sistem diharapkan menolak request karena konteks operasional ticket sudah tidak valid.

Pengujian *Just-In-Time Access* pada bagian ini digunakan untuk memastikan bahwa fitur sensitif tidak dapat dijalankan hanya berdasarkan role Customer Support atau assignment ticket. Akses terhadap fitur sensitif harus melalui session JIT yang valid, sesuai feature, masih aktif, belum digunakan, dan belum melewati batas waktu akses. Dengan demikian, pengujian ini memverifikasi pembatasan durasi dan kondisi akses terhadap fitur sensitif pada backend *Prototype*.

3.7.5 Teknik Evaluasi Hasil

Teknik evaluasi hasil pada penelitian ini dilakukan dengan membandingkan hasil yang diharapkan (*expected result*) dan hasil aktual (*actual result*) pada setiap skenario pengujian. Setiap skenario dinilai berdasarkan keputusan backend dalam menerima atau menolak request sesuai aturan kontrol akses yang telah dirancang. Evaluasi hasil pengujian menggunakan beberapa indikator sebagai berikut:

1. Respons API yang diberikan oleh backend, seperti status 200, 201, 403, atau respons penolakan lain yang relevan.

2. Status akses, yaitu apakah request diterima atau ditolak sesuai dengan rancangan.
3. Perubahan state pada database, seperti perubahan assigned_cs_id, pembuatan data pada tabel jit_sessions, perubahan is_active, dan nilai expired_at.
4. Audit log dan terminal log sebagai bukti pendukung untuk menunjukkan proses validasi yang terjadi pada backend.
5. Kesesuaian hasil aktual terhadap skenario pengujian yang telah ditentukan.

Suatu skenario dinyatakan sesuai apabila hasil aktual sama dengan hasil yang diharapkan. Pada pengujian RBAC, skenario dinyatakan sesuai apabila akses berdasarkan role yang benar diterima dan akses lintas role ditolak. Pada pengujian *Least Privilege*, skenario dinyatakan sesuai apabila Customer Support hanya dapat mengakses ruang kerja dan ticket yang berada dalam konteks assignment-nya, serta ditolak ketika mencoba mengakses ticket milik Customer Support lain. Pada pengujian *Just-In-Time Access*, skenario dinyatakan sesuai apabila session JIT hanya dibuat pada ticket yang valid, fitur sensitif hanya dapat dijalankan ketika session JIT aktif dan sesuai feature, serta request ditolak ketika session tidak tersedia, sudah digunakan, expired, atau konteks ticket tidak valid.

Hasil pengujian pada Bab IV disajikan dalam bentuk matriks pengujian yang memuat ID skenario, mekanisme yang diuji, aktor, aksi pengujian, expected result, actual result, status pengujian, dan bukti pengujian. Matriks tersebut digunakan untuk memperjelas cakupan pengujian dan memudahkan penelusuran antara rancangan skenario, hasil aktual, dan bukti pendukung. Evaluasi pada penelitian ini bersifat *verification-based*. Artinya, hasil pengujian digunakan untuk memverifikasi bahwa kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* berjalan sesuai rancangan pada backend *Prototype*. Evaluasi ini tidak digunakan untuk mengukur performa sistem, melakukan penetration testing, atau menyatakan bahwa sistem telah aman secara absolut dan siap digunakan pada lingkungan produksi berskala industri.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Sistem

Pada bagian ini dijelaskan hasil implementasi dari mekanisme kontrol akses pengguna internal yang telah dirancang pada bab sebelumnya. *Prototype* sistem dompet digital pada penelitian ini diimplementasikan dengan fokus utama pada backend *Access control layer*, yaitu lapisan backend yang bertugas memvalidasi dan menentukan apakah suatu request dapat diterima atau harus ditolak. Implementasi backend dibangun menggunakan Golang dengan framework Gin sebagai pengelola routing dan API. Sistem juga menggunakan MySQL sebagai basis data untuk menyimpan data pengguna, ticket, session JIT, serta data pendukung lain yang dibutuhkan dalam proses kontrol akses. Implementasi sistem tidak diarahkan untuk membahas seluruh fitur dompet digital, melainkan untuk menunjukkan bagaimana mekanisme RBAC, *Least Privilege*, dan *Just-In-Time Access* diterapkan pada backend *Prototype*.

4.1.1 Implementasi *Role-Based Access Control* (RBAC)

Role-Based Access Control (RBAC) diimplementasikan sebagai lapisan awal otorisasi pada backend. Mekanisme ini digunakan untuk memisahkan area akses dasar berdasarkan role pengguna, yaitu User, Customer Support (CS), dan Administrator. Pemisahan role tersebut menjadi fondasi awal sebelum sistem menerapkan pembatasan yang lebih spesifik melalui *Least Privilege* dan *Just-In-Time Access*.

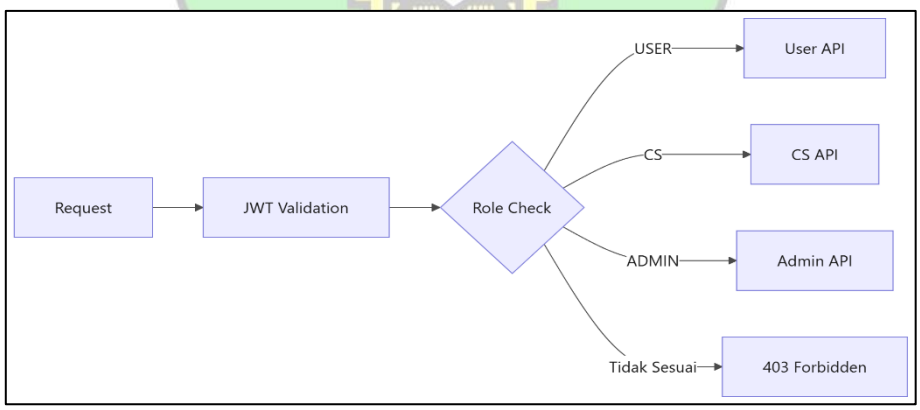
Pada implementasinya, informasi role disimpan pada tabel users. Setiap akun memiliki identitas dan role yang digunakan sebagai dasar penentuan hak akses awal. Akun Customer Support digunakan untuk mengakses area kerja CS, akun User digunakan untuk area pengguna akhir, sedangkan akun Administrator digunakan untuk area administratif. Struktur data role pada tabel users dapat dilihat pada Gambar 4.1.

	id	email	password	role	created_at
Edit Copy Delete	1	cs01@test.com	$2a$10$SDYUFjXaqLJt8xMT.nBFPORaeMBGN8Wus.t8CSlaAJs...	cs	2026-05-02
Edit Copy Delete	2	cs02@test.com	$2a$10$SDYUFjXaqLJt8xMT.nBFPORaeMBGN8Wus.t8CSlaAJs...	cs	2026-05-02
Edit Copy Delete	3	user@test.com	$2a$10$8fWDmS1XjQ7abWcl4rTFNO3rD8Vrx47m7HZ1mMWpYFx...	user	2026-05-02
Edit Copy Delete	4	admin@test.com	$2a$10$XFRYF9ZUZ8688eos2No2tO38h3GJV7lv7fHL9mPueZB...	admin	2026-05-02

Gambar 4. 1 Data Role Pengguna pada Tabel users

Setelah proses login berhasil, backend menghasilkan token JWT yang membawa informasi identitas pengguna, termasuk user_id dan role. Informasi tersebut digunakan oleh middleware untuk memvalidasi setiap request yang masuk. Dengan demikian, sistem tidak hanya memeriksa apakah pengguna telah terautentikasi, tetapi juga memeriksa apakah role pengguna sesuai dengan endpoint yang dituju.

Alur validasi RBAC dimulai dari request yang masuk ke backend, kemudian melewati validasi JWT. Setelah token dinyatakan valid, sistem melakukan pemeriksaan role. Jika role sesuai, request diteruskan ke area API yang sesuai, seperti User API, CS API, atau Admin API. Jika role tidak sesuai, backend menolak request dengan respons 403 Forbidden. Alur validasi RBAC pada backend dapat dilihat pada Gambar 4.2.



Gambar 4. 2 Alur Validasi RBAC pada Backend

Penerapan RBAC secara teknis dilakukan melalui pengelompokan route pada backend. Route /user dipasangkan dengan middleware RBAC untuk role User, route /cs dipasangkan dengan middleware RBAC untuk role CS, sedangkan route /admin dipasangkan dengan middleware RBAC untuk role Administrator.

Dengan pola ini, setiap endpoint berada pada kelompok akses yang jelas dan hanya dapat diakses oleh role yang sesuai. Implementasi route grouping dan middleware RBAC dapat dilihat pada Gambar 4.3.

```
auth := r.Group("/")
auth.Use(middleware.JWTAuth(cfg))

user := auth.Group("/user")
user.Use(middleware.RBAC("user"))

cs := auth.Group("/cs")
cs.Use(middleware.RBAC("cs"))

admin := auth.Group("/admin")
admin.Use(middleware.RBAC("admin"))
```

Gambar 4. 3 Implementasi Route Grouping dan Middleware RBAC

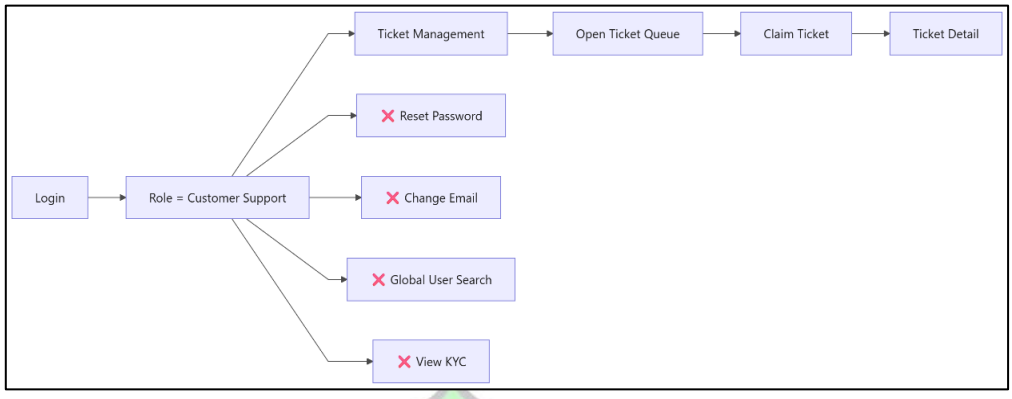
Dalam konteks penelitian ini, RBAC berfungsi sebagai gerbang awal untuk memastikan bahwa pengguna hanya dapat memasuki area kerja sesuai role yang dimilikinya. Namun, RBAC tidak dijadikan satu-satunya mekanisme pembatasan akses. Role Customer Support memang dapat mengakses area kerja CS, tetapi hal tersebut belum berarti CS dapat membuka seluruh data pengguna atau menjalankan fitur sensitif secara bebas. Pembatasan ruang lingkup akses CS tetap dikendalikan melalui *Least Privilege* berbasis ticket, sedangkan akses terhadap fitur sensitif dikendalikan melalui *Just-In-Time Access*.

4.1.2 Implementasi *Least Privilege*

Least Privilege diimplementasikan untuk membatasi ruang lingkup akses Customer Support (CS) agar hanya berada pada kebutuhan operasional penanganan ticket. Setelah proses login dan validasi RBAC berhasil, sistem tidak langsung memberikan akses luas kepada CS. Akses CS diarahkan pada area Ticket Management, sedangkan fitur lain seperti pencarian pengguna secara global, perubahan data akun, reset password, dan akses data KYC tidak disediakan sebagai akses langsung.

Pola pembatasan ini menunjukkan bahwa role CS tidak otomatis memiliki kewenangan terhadap seluruh data dan fitur internal. CS hanya diberi akses awal

untuk melihat antrean ticket, melakukan claim ticket, dan membuka detail ticket yang berada dalam konteks penugasannya. Alur pembatasan akses CS berdasarkan prinsip *Least Privilege* dapat dilihat pada Gambar 4.4.



Gambar 4. 4 Alur Pembatasan Akses CS Berdasarkan *Least Privilege*

Setelah ticket berhasil di-claim, sistem membentuk hubungan assignment antara ticket dan CS yang menangani ticket tersebut. Hubungan ini menjadi dasar pembatasan akses pada level data. Dengan demikian, akses terhadap detail ticket tidak diberikan secara global kepada semua CS, melainkan hanya kepada CS yang memang ditugaskan pada ticket tersebut.

Secara teknis, validasi *Least Privilege* dilakukan pada backend melalui pemeriksaan assignment ticket. Ketika CS mengakses detail ticket, backend terlebih dahulu mengambil data ticket berdasarkan ticket_id. Jika ticket tidak ditemukan, request dihentikan. Jika ticket ditemukan, backend membandingkan identitas CS yang sedang login dengan nilai CS yang tersimpan pada data ticket. Apabila ticket tidak ditugaskan kepada CS tersebut, backend menolak request dengan status 403 Forbidden. Potongan implementasi validasi assignment ticket dapat dilihat pada Gambar 4.5.

```
ticket, err := ticketRepo.GetByID(ticketID)
if err != nil {
    c.AbortWithStatus(http.StatusNotFound)
    return
}

if ticket.CSID != userID {
    c.AbortWithStatus(http.StatusForbidden)
    return
}

c.Next()
```

Gambar 4. 5 Validasi Assignment Ticket pada Middleware Backend

Dengan implementasi tersebut, *Least Privilege* tidak hanya diterapkan pada tampilan antarmuka, tetapi juga ditegakkan pada backend. CS tetap dapat menjalankan tugas operasional seperti menangani ticket, namun tidak dapat mengakses ticket milik CS lain atau data pengguna tanpa konteks ticket yang sah. Adapun fitur sensitif seperti VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT tetap berada di luar cakupan akses langsung *Least Privilege* dan hanya dapat dibuka melalui mekanisme *Just-In-Time Access*.

4.1.3 Implementasi Mekanisme *Just-In-Time Access*

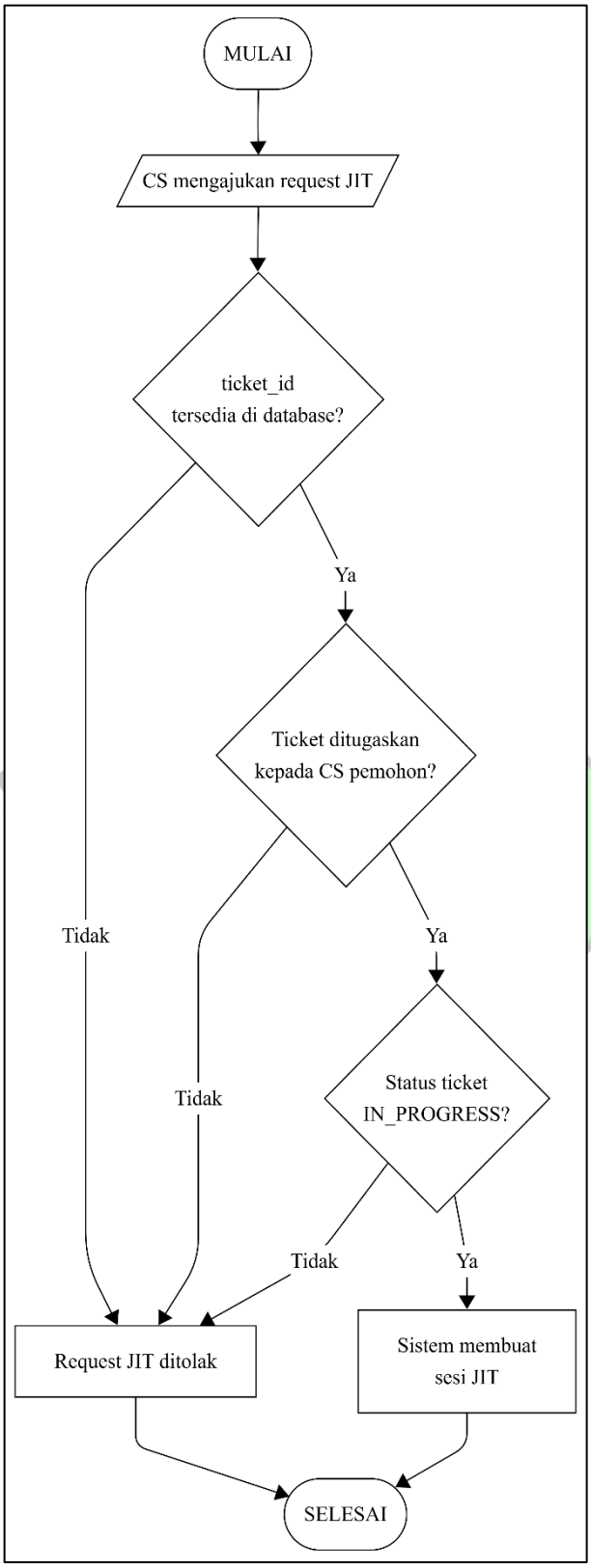
Just-In-Time Access diimplementasikan sebagai lapisan kontrol tambahan setelah *Least Privilege*. Jika *Least Privilege* membatasi ruang lingkup akses Customer Support (CS) berdasarkan ticket, maka *Just-In-Time Access* membatasi penggunaan fitur sensitif agar tidak aktif secara permanen. Fitur seperti VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT tetap berada dalam kondisi terkunci meskipun CS telah login dan berada pada area kerja yang sesuai.

Pada implementasinya, CS harus mengajukan request JIT terlebih dahulu sebelum menjalankan fitur sensitif. Request tersebut diproses oleh backend melalui alur validasi berikut:

1. Backend memeriksa apakah ticket_id yang diminta tersedia pada database.

2. Backend memeriksa apakah ticket tersebut memang ditugaskan kepada CS yang mengajukan request.
3. Backend memeriksa status ticket. Pada *Prototype* ini, ticket memiliki status OPEN, CLAIMED, IN_PROGRESS, RESOLVED, dan CLOSED, namun session JIT hanya dapat diberikan apabila ticket berada pada status IN_PROGRESS.
4. Backend memeriksa apakah feature yang diminta termasuk dalam daftar fitur sensitif yang dikendalikan melalui mekanisme JIT.

Apabila salah satu tahapan validasi tersebut tidak terpenuhi, request JIT ditolak dan session tidak diterbitkan. Sebaliknya, apabila seluruh validasi terpenuhi, backend membuat session JIT sementara untuk feature sensitif yang diminta. Pada *Prototype* ini, session JIT berlaku selama 15 menit dan hanya dapat digunakan satu kali. Durasi tersebut digunakan sebagai konfigurasi akses sementara pada *Prototype* dan menjadi bagian dari skenario pengujian kedaluwarsa session pada subbab pengujian. Alur validasi request JIT dapat dilihat pada Gambar 4.6.



Gambar 4. 6 Alur validasi pemberian mekanisme JIT

Session JIT yang berhasil diterbitkan disimpan pada tabel jit_sessions. Data yang disimpan mencakup identitas CS, ticket yang terkait, feature yang diizinkan, waktu kedaluwarsa melalui nilai expired_at, serta status keaktifan session melalui nilai is_active. Struktur ini menunjukkan bahwa akses JIT tidak berlaku secara umum, melainkan hanya sah untuk CS tertentu, ticket tertentu, dan feature tertentu. Sebagai contoh, session JIT untuk CHANGE_EMAIL tidak dapat digunakan untuk menjalankan RESET_PASSWORD, karena setiap session hanya berlaku untuk feature yang diminta. Bentuk penyimpanan session JIT pada database dapat dilihat pada Gambar 4.7.

id	cs_id	ticket_id	feature	expired_at	is_active
1	2	1	RESET_PASSWORD	2026-05-12 14:19:23.668	1
2	2	2	UNBLOCK_ACCOUNT	2026-05-12 15:56:46.981	0
16	2	8	CHANGE_EMAIL	2026-06-10 17:12:42.255	0

Gambar 4. 7 Tabel jit_sessions Yang Menyimpan JIT Session

Dengan implementasi tersebut, *Just-In-Time Access* tidak memberikan akses sensitif secara permanen kepada Customer Support. Akses hanya diberikan ketika ticket valid, ticket ditugaskan kepada CS yang benar, status ticket berada pada kondisi in_progress, feature yang diminta sesuai, session JIT masih aktif, belum digunakan, dan belum melewati waktu kedaluwarsa. Mekanisme ini menjadi dasar bagi pengujian pada subbab berikutnya, terutama untuk memverifikasi pembuatan session yang valid, eksekusi fitur sensitif saat session aktif, serta penolakan akses ketika session tidak tersedia, expired, telah digunakan, atau konteks ticket tidak valid.

4.1.4 Implementasi Audit Log dan Terminal Log

Audit log dan terminal log diimplementasikan sebagai komponen pendukung untuk membantu penelusuran aktivitas sistem dan proses validasi backend. Komponen ini tidak berperan sebagai mekanisme utama dalam memberikan atau menolak akses, karena keputusan akses tetap ditentukan oleh validasi RBAC, *Least Privilege*, dan *Just-In-Time Access* pada backend. Dalam

penelitian ini, bukti utama pengujian tetap difokuskan pada respons API, status akses, dan perubahan data pada database, sedangkan audit log dan terminal log digunakan sebagai informasi tambahan untuk memperlihatkan bahwa aktivitas penting pada *Prototype* dapat ditelusuri. Contoh data audit log dapat dilihat pada Gambar 4.8.

id	user_id	role	action	ticket_id	metadata	created_at
20	2	cs	JIT_REQUEST	1	{"feature": "RESET_PASSWORD", "expired_at": "2026-...	2026-05-12 14:04:23.677
45	2	cs	JIT_REQUEST	2	{"feature": "UNBLOCK_ACCOUNT", "expired_at": "2026...	2026-05-12 15:41:46.985
48	2	cs	JIT_REQUEST	2	{"feature": "CHANGE_EMAIL", "expired_at": "2026-05...	2026-05-12 15:41:57.159
77	2	cs	TICKET_STATUS_UPDATE	1	{"status": "RESOLVED"}	2026-05-12 16:55:50.310
79	2	cs	JIT_REQUEST	3	{"feature": "VIEW_KYC", "expired_at": "2026-05-12T...	2026-05-12 17:23:43.686
106	2	cs	TICKET_STATUS_UPDATE	3	{"status": "RESOLVED"}	2026-05-13 13:23:51.509
181	2	cs	TICKET_STATUS_UPDATE	4	{"status": "RESOLVED"}	2026-05-19 14:21:08.248
195	2	cs	TICKET_STATUS_UPDATE	8	{"status": "IN_PROGRESS"}	2026-06-10 16:57:29.107
196	2	cs	JIT_REQUEST	8	{"reason": "Seluruh pemeriksaan backend lolos dan ...	2026-06-10 16:57:42.259

Gambar 4. 8 Data Aktivitas pada Tabel Audit Log

Selain audit log, terminal log digunakan sebagai pendukung saat proses pengembangan dan pengujian untuk melihat respons backend secara langsung, seperti request yang diterima, query database, status respons, dan proses update session. Terminal log tidak dijadikan fokus utama evaluasi, tetapi membantu memastikan bahwa proses validasi berjalan sesuai alur ketika pengujian dilakukan. Contoh proses validasi backend melalui terminal log dapat dilihat pada Gambar 4.9.

```
2026/06/22 15:29:19 C:/laragon/www/test1/internal/repository/mysql/ticket_repository.go:26
[1.078ms] [rows:1] SELECT * FROM `tickets` WHERE `tickets`.`id` = 9 ORDER BY `tickets`.`id` LIMIT 1
[GIN] 2026/06/22 - 15:29:19 | 200 |      2.1082ms |             ::1 | GET      "/cs/tickets/9"
[GIN] 2026/06/22 - 15:29:19 | 403 |      2.1082ms |             ::1 | GET      "/cs/tickets/9/messages?

2026/06/22 15:29:19 C:/laragon/www/test1/internal/repository/mysql/ticket_repository.go:26
[0.532ms] [rows:1] SELECT * FROM `tickets` WHERE `tickets`.`id` = 9 ORDER BY `tickets`.`id` LIMIT 1

2026/06/22 15:29:19 C:/laragon/www/test1/internal/repository/mysql/user_profile_repository.go:23
[1.756ms] [rows:1] SELECT * FROM `user_profiles` WHERE user_id = 3 ORDER BY `user_profiles`.`id` LIMIT

2026/06/22 15:29:19 C:/laragon/www/test1/internal/repository/mysql/jit_session_repository.go:44
[1.261ms] [rows:0] UPDATE `jit_sessions` SET `is_active`=false WHERE is_active = 1 AND expired_at <= '2
```

Gambar 4. 9 Contoh Riwayat Terminal Logs

Dengan demikian, audit log dan terminal log ditempatkan sebagai komponen pendukung dalam *Prototype*. Pembuktian utama tetap mengacu pada respons API, status akses, dan perubahan data pada database, sedangkan log

digunakan untuk membantu menelusuri aktivitas dan proses backend apabila diperlukan.

4.2 Matriks Pengujian Backend *Access* Control

Pengujian pada Bab IV disajikan dalam bentuk matriks untuk memperjelas hubungan antara skenario, mekanisme yang diuji, hasil yang diharapkan, hasil aktual, dan bukti pengujian. Matriks ini digunakan untuk menunjukkan bahwa pengujian tidak diarahkan pada seluruh fitur *Prototype* dompet digital, melainkan pada verifikasi tiga lapisan kontrol akses, yaitu RBAC, *Least Privilege*, dan *Just-In-Time Access* pada backend.

Tabel 4. 1 Matriks Pengujian

NO	Aspek	Skenario Pengujian	Expected	Actual	Status	Bukti
1.	RBAC	User mengakses endpoint area User	Diterima	200 OK	Sesuai	API
2.	RBAC	CS mengakses endpoint area CS	Diterima	200 OK	Sesuai	API/ Gambar
3.	RBAC	Admin mengakses endpoint area Admin	Diterima	200 OK	Sesuai	API
4.	RBAC	CS mengakses endpoint Admin	Ditolak	403 Forbidden	Sesuai	API/ Gambar
5.	RBAC	User mengakses endpoint area CS	Ditolak	403 Forbidden	Sesuai	API
6.	RBAC	User mengakses endpoint area Admin	Ditolak	403 Forbidden	Sesuai	API
7.	RBAC	Admin mengakses endpoint operasional CS	Ditolak	403 Forbidden	Sesuai	API
8.	LP	CS login dan hanya memperoleh akses ke area kerja ticket	Dibatasi pada area ticket	Area CS hanya menampilkan konteks ticket	Sesuai	Frontend/API

9.	LP	CS melakukan claim terhadap ticket yang masih tersedia	Diterima	200 OK dan assigned_cs_id terisi	Sesuai	API/Database
10.	LP	CS mencoba claim ticket yang sudah ditugaskan kepada CS lain	Ditolak	409 Conflict	Sesuai	API/Database
11.	LP	CS membuka detail ticket yang ditugaskan kepadanya	Diterima	200 OK	Sesuai	API
12.	LP	CS membuka detail ticket milik CS lain	Ditolak	403 Forbidden	Sesuai	API
13.	JIT	CS mengajukan request JIT untuk CHANGE_EMAIL pada ticket valid dan in_progress	Session JIT dibuat	201 Created	Sesuai	API/ Database
14.	JIT	CS menjalankan CHANGE_EMAIL saat session JIT aktif	Diterima	200 OK	Sesuai	API
15.	JIT	CS menjalankan CHANGE_EMAIL tanpa session JIT valid	Ditolak	403 Forbidden	Sesuai	API
16.	JIT	CS menjalankan ulang CHANGE_EMAIL setelah session digunakan/tidak aktif	Ditolak	403 Forbidden	Sesuai	API/Database
17.	JIT	CS menjalankan CHANGE_EMAIL setelah expired_at melewati batas waktu	Ditolak	403 Forbidden	Sesuai	API/Database
18.	JIT	Session JIT CHANGE_EMAIL digunakan untuk feature berbeda, misalnya RESET_PASSWORD	Ditolak	403 Forbidden	Sesuai	API/Database
19.	JIT	CS menjalankan fitur sensitif setelah ticket	Ditolak	403 Forbidden	Sesuai	API/Database

		berubah menjadi closed				

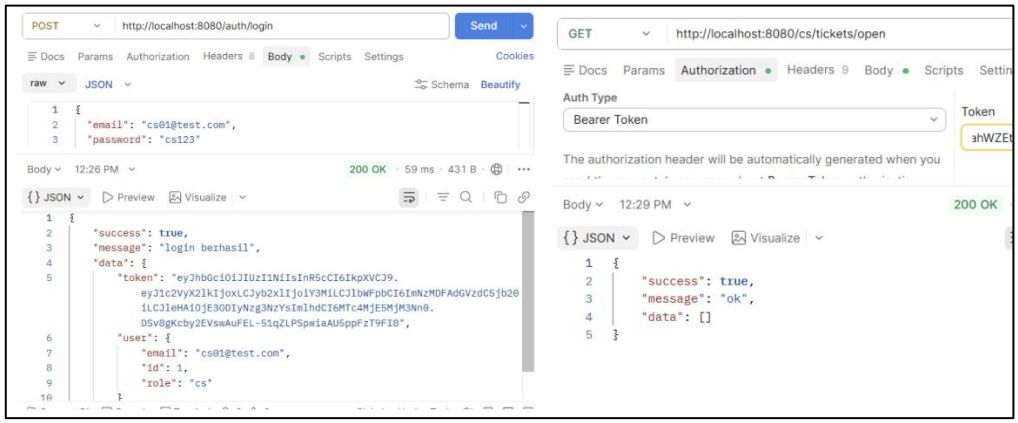
Berdasarkan Tabel 4.1, skenario pengujian disusun untuk mewakili tiga lapisan kontrol akses. Skenario RBAC digunakan untuk memverifikasi batas awal berdasarkan role. Skenario *Least Privilege* digunakan untuk memverifikasi pembatasan lingkup akses Customer Support berdasarkan assignment ticket. Skenario *Just-In-Time Access* digunakan untuk memverifikasi bahwa fitur sensitif hanya dapat dijalankan melalui session sementara yang valid, sesuai feature, masih aktif, dan belum melewati batas waktu akses.

4.3 Pengujian *Role-Based Access Control*

Pengujian *Role-Based Access Control* dilakukan untuk memverifikasi bahwa sistem dapat membatasi akses awal berdasarkan role pengguna. Pada bagian ini, pengujian difokuskan pada pemisahan area akses dasar antara User, Customer Support (CS), dan Administrator. Sesuai dengan fokus penelitian, pembahasan utama diarahkan pada role Customer Support, sedangkan User dan Administrator digunakan sebagai pembanding untuk memastikan batas akses dasar antar-role. Pengujian ini mengacu pada skenario RBAC yang telah dirangkum dalam matriks pengujian pada Tabel 4.1.

4.3.1 Pengujian Akses Sesuai Role

Pengujian pertama dilakukan untuk memastikan bahwa pengguna dengan role yang sah dapat mengakses area sistem sesuai kewenangannya. Pada pengujian ini, akun dengan role Customer Support berhasil login dan digunakan untuk mengakses area kerja CS melalui endpoint /cs/tickets/open. Hasil pengujian menunjukkan bahwa backend menerima request dengan respons 200 OK karena role yang terbaca pada token sesuai dengan endpoint yang diakses. Meskipun data ticket yang dikembalikan kosong, respons 200 OK menunjukkan bahwa endpoint area CS dapat diakses oleh role yang sesuai. Bukti akses CS terhadap area kerja yang sesuai dapat dilihat pada Gambar 4.10.

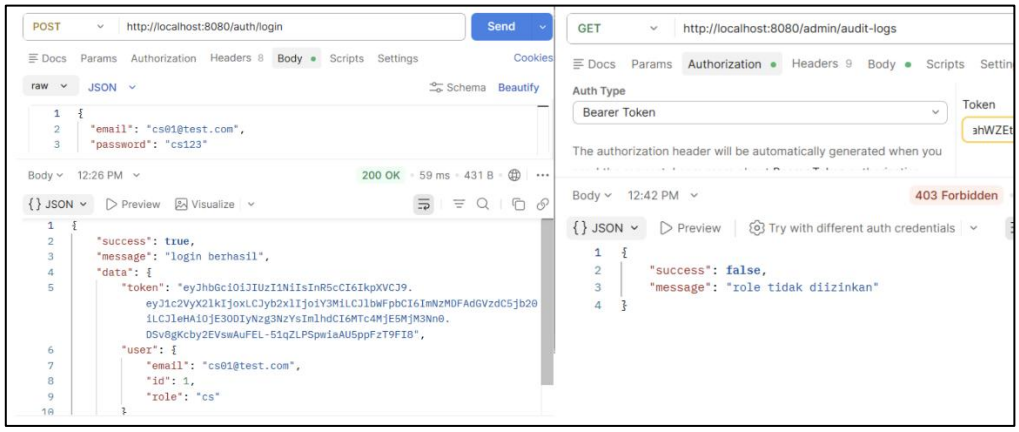


Gambar 4. 10 Role CS Berhasil Mengakses Area Endpoint Role CS

Pengujian dengan pola yang sama juga diterapkan pada role User dan Administrator sebagai pembanding. User diarahkan pada area pengguna akhir, sedangkan Administrator diarahkan pada area administratif. Hasil ringkas pengujian akses sesuai role tersebut dicantumkan pada matriks pengujian, sehingga RBAC dapat ditelusuri berdasarkan expected result, actual result, status, dan bukti pengujian.

4.3.2 Pengujian Penolakan Akses Lintas Role

Pengujian berikutnya dilakukan untuk memastikan bahwa pengguna tidak dapat mengakses area yang berada di luar role-nya. Pada skenario ini, token milik Customer Support digunakan untuk mengakses endpoint Administrator, yaitu /admin/audit-logs. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 403 Forbidden karena role CS tidak memiliki kewenangan untuk mengakses area Administrator. Penolakan ini menunjukkan bahwa validasi role dilakukan pada backend, sehingga endpoint tetap ditolak meskipun dipanggil langsung melalui Postman atau API client. Bukti penolakan akses lintas role dapat dilihat pada Gambar 4.11.



Gambar 4. 11 Request Ditolak Akses Lintas Role

Berdasarkan hasil pengujian tersebut, RBAC telah berfungsi sebagai lapisan awal otorisasi pada backend. Sistem menerima request ketika role pengguna sesuai dengan endpoint yang dituju dan menolak request ketika pengguna mencoba mengakses area di luar role-nya. Pengujian ini dilakukan pada endpoint representatif untuk membuktikan pemisahan area akses dasar antar-role, bukan untuk menguji seluruh endpoint aplikasi satu per satu. Hasil ringkas skenario RBAC lainnya telah dicantumkan pada matriks pengujian, sedangkan bukti visual pada bagian ini digunakan sebagai contoh representatif dari akses sesuai role dan penolakan akses lintas role. Hasil ini menjadi dasar sebelum pengujian dilanjutkan pada pembatasan yang lebih spesifik melalui *Least Privilege* dan *Just-In-Time Access*.

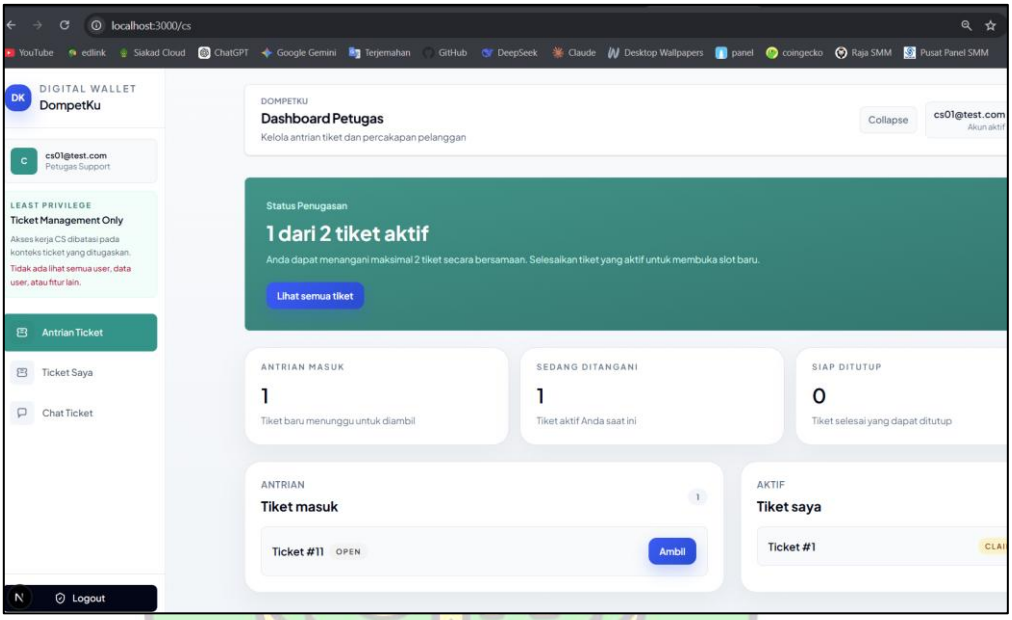
4.4 Pengujian *Least Privilege*

Pengujian *Least Privilege* dilakukan untuk memverifikasi bahwa akses Customer Support (CS) dibatasi pada ruang kerja yang sesuai dengan tugas penanganan ticket. Pada bagian ini, pengujian diarahkan untuk memastikan bahwa CS tidak memiliki akses global terhadap seluruh data pengguna, tidak memiliki menu daftar seluruh user, dan hanya dapat mengakses ticket yang berada dalam konteks penugasannya.

4.4.1 Pengujian Lingkup Ruang Kerja CS dan Claim Ticket

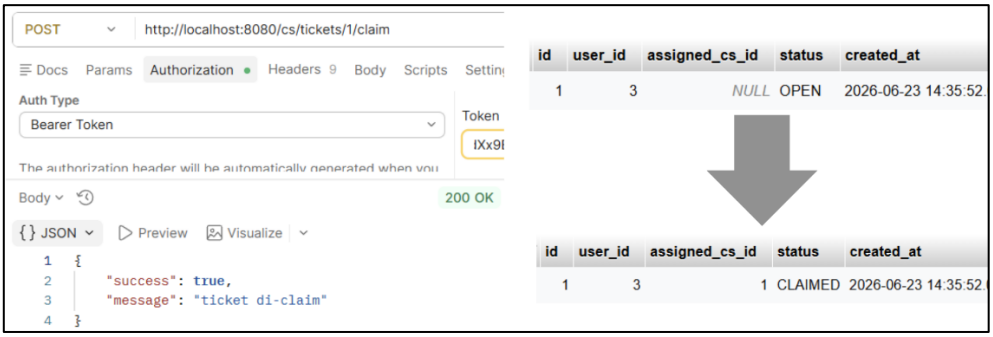
Pengujian pertama dilakukan setelah akun Customer Support berhasil login. Hasil pengujian menunjukkan bahwa CS hanya diarahkan pada area kerja

ticket. Sistem tidak menyediakan menu untuk melihat daftar seluruh user, data pengguna global, maupun fitur administratif lain di luar kewenangan CS. Hal ini menunjukkan bahwa akses awal CS telah diminimalkan sesuai prinsip *Least Privilege*. Bukti ruang kerja CS yang terbatas dapat dilihat pada Gambar 4.12.



Gambar 4. 12 Tampilan Kerja Area CS Yang Dibatasi Pada Konteks Ticket

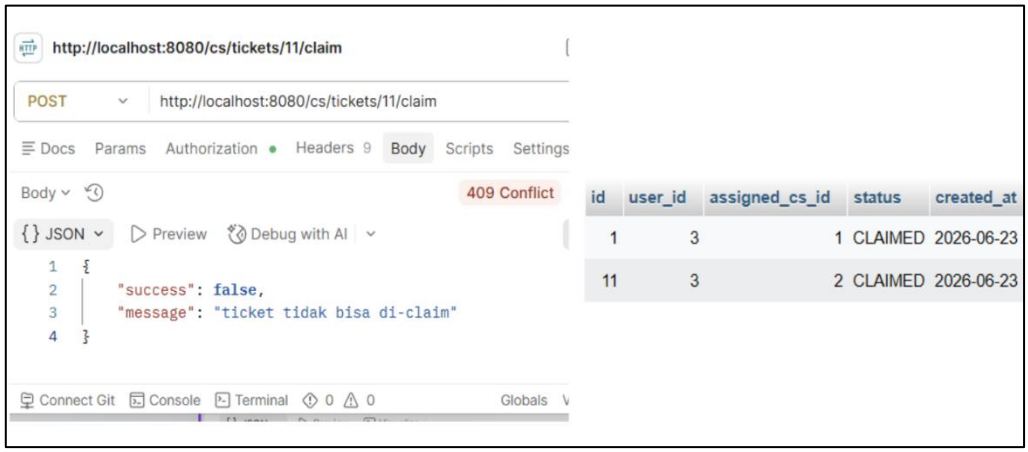
Selanjutnya, pengujian dilakukan menggunakan akun Customer Support cs01@test.com dengan ID 1. Akun tersebut melakukan claim terhadap ticket dengan ID 1 melalui endpoint POST /cs/tickets/1/claim. Hasil pengujian menunjukkan bahwa backend menerima request dengan respons 200 OK. Setelah proses claim dilakukan, data pada tabel tickets berubah dari assigned_cs_id = NULL menjadi assigned_cs_id = 1, dan status ticket berubah dari OPEN menjadi CLAIMED, sebagaimana ditunjukkan pada Gambar 4.13.



Gambar 4. 13 Ticket Berhasil Di-assign ke Customer Support ID 1

Status CLAIMED menunjukkan bahwa ticket telah berhasil diambil oleh CS. Pada tahap berikutnya, ticket yang sedang diproses dapat berada pada status in_progress, yang menjadi salah satu syarat validasi sebelum session JIT diterbitkan. Berdasarkan hasil tersebut, CS tetap dapat menjalankan tugas operasional yang sah, yaitu melihat area ticket dan melakukan claim ticket. Namun, akses tersebut tidak bersifat global karena tetap bergantung pada hubungan antara CS dan ticket yang ditugaskan.

Selanjutnya untuk memastikan bahwa ticket yang sudah ditugaskan tidak dapat diambil ulang oleh Customer Support lain, pengujian dilanjutkan dengan mencoba melakukan claim terhadap ticket yang telah memiliki assignment. Pada skenario ini, akun Customer Support dengan ID 1 mencoba melakukan claim terhadap ticket dengan ID 11 melalui endpoint POST /cs/tickets/11/claim. Berdasarkan data pada tabel tickets, ticket tersebut telah memiliki nilai assigned_cs_id = 2 dan berstatus CLAIMED, sehingga ticket berada dalam penugasan Customer Support lain. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 409 Conflict dan pesan “ticket tidak bisa di-claim”. Penolakan ini menunjukkan bahwa sistem tidak mengizinkan Customer Support mengambil alih ticket yang sudah ditugaskan kepada Customer Support lain. Dengan demikian, proses assignment ticket yang telah terbentuk tetap dipertahankan oleh backend. Bukti penolakan claim ticket yang sudah ditugaskan dapat dilihat pada Gambar 4.14.

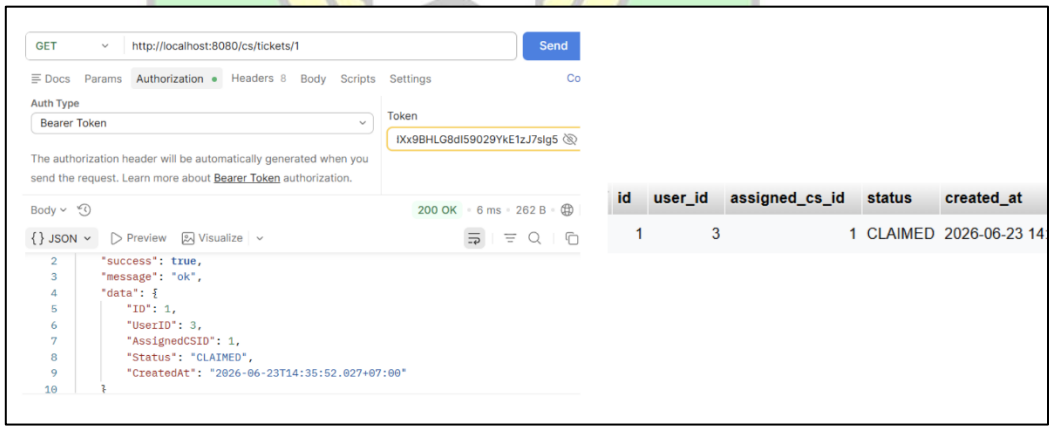


Gambar 4. 14 Penolakan Claim Ticket yang Sudah Ditugaskan kepada CS Lain

Berdasarkan rangkaian pengujian pada subbagian ini, Customer Support tetap dapat menjalankan tugas operasional yang sah, seperti melihat area ticket dan melakukan claim terhadap ticket yang tersedia. Namun, akses tersebut tidak bersifat global karena backend tetap membatasi proses claim berdasarkan status ticket dan assignment yang telah terbentuk.

4.4.2 Pengujian Akses Berdasarkan Assignment Ticket

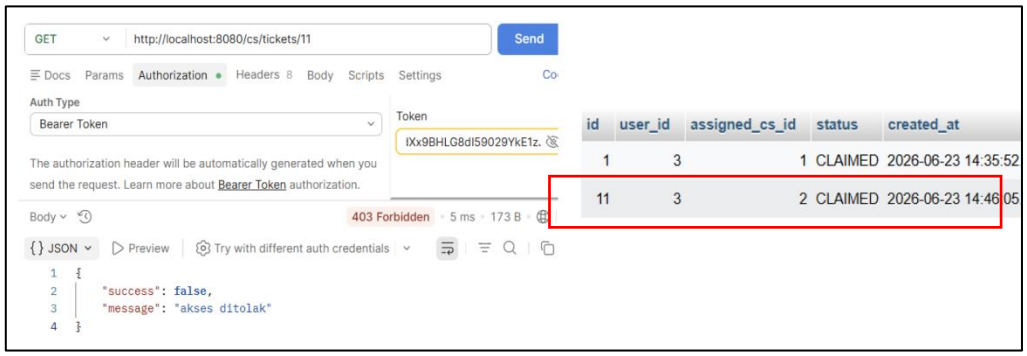
Pengujian berikutnya dilakukan untuk memastikan bahwa CS hanya dapat membuka ticket yang berada dalam konteks penugasannya. Pada skenario ini, akun CS dengan ID 1 mengakses detail ticket dengan ID 1 melalui endpoint GET /cs/tickets/1. Ticket tersebut sebelumnya telah di-assign kepada CS yang sama, sebagaimana ditunjukkan pada tabel tickets dengan nilai assigned_cs_id = 1. Hasil pengujian menunjukkan bahwa backend menerima request dan mengembalikan respons 200 OK. Respons tersebut menampilkan data ticket dengan ID 1, UserID = 3, AssignedCSID = 1, dan status CLAIMED. Bukti akses detail ticket yang sesuai assignment dapat dilihat pada Gambar 4.15.



Gambar 4. 15 Akses Detail Ticket yang Di-assign ke CS 1 Berhasil

Setelah pengujian akses ticket milik sendiri berhasil dilakukan, pengujian dilanjutkan dengan mencoba membuka ticket yang ditugaskan kepada CS lain. Pada skenario ini, akun CS dengan ID 1 mencoba mengakses ticket dengan ID 11 melalui endpoint GET /cs/tickets/11. Berdasarkan data pada tabel tickets, ticket tersebut memiliki nilai assigned_cs_id = 2, sehingga ticket berada di luar konteks penugasan CS yang sedang login. Hasil pengujian menunjukkan bahwa backend

menolak request dengan respons 403 Forbidden dan pesan akses ditolak, sebagaimana ditunjukkan pada Gambar 4.16.



Gambar 4. 16 Akses Detail Ticket Milik CS Lain Ditolak

Hasil pengujian *Least Privilege* menunjukkan bahwa role Customer Support saja tidak cukup untuk mengakses seluruh ticket atau data pengguna. Backend tetap memeriksa hubungan assignment antara CS yang sedang login dan ticket yang diakses. Dengan demikian, akses CS dibatasi pada konteks ticket yang ditugaskan kepadanya, sehingga *Least Privilege* berjalan sebagai pembatas ruang lingkup akses sebelum fitur sensitif dikendalikan lebih lanjut melalui mekanisme *Just-In-Time Access*.

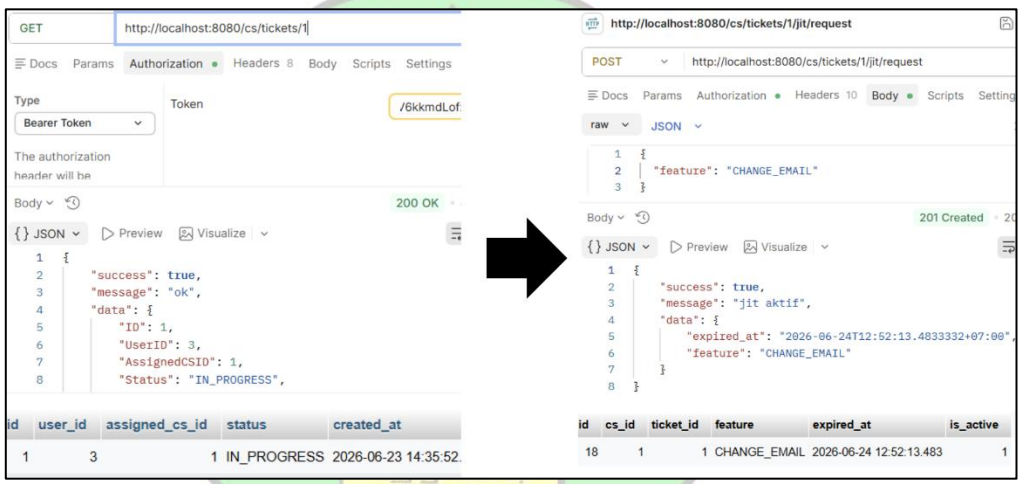
4.5 Pengujian *Just-In-Time Access*

Pengujian *Just-In-Time Access* dilakukan untuk memverifikasi bahwa fitur sensitif tidak dapat digunakan secara permanen oleh Customer Support (CS). Pada bagian ini, pengujian difokuskan pada pembuatan session JIT, penggunaan fitur sensitif saat session aktif, serta penolakan akses ketika session tidak tersedia, sudah digunakan, expired, tidak sesuai feature, atau konteks ticket tidak valid. Endpoint sensitif yang digunakan sebagai contoh representatif adalah CHANGE_EMAIL, karena endpoint tersebut melewati pola validasi JIT yang sama dengan fitur sensitif lain seperti VIEW_KYC, RESET_PASSWORD, RESET_PIN, dan UNBLOCK_ACCOUNT.

4.5.1 Pengujian Request dan Penggunaan Session JIT

Pengujian pertama dilakukan dengan mengajukan request JIT pada ticket yang valid. Pada skenario ini, akun Customer Support dengan ID 1 mengakses

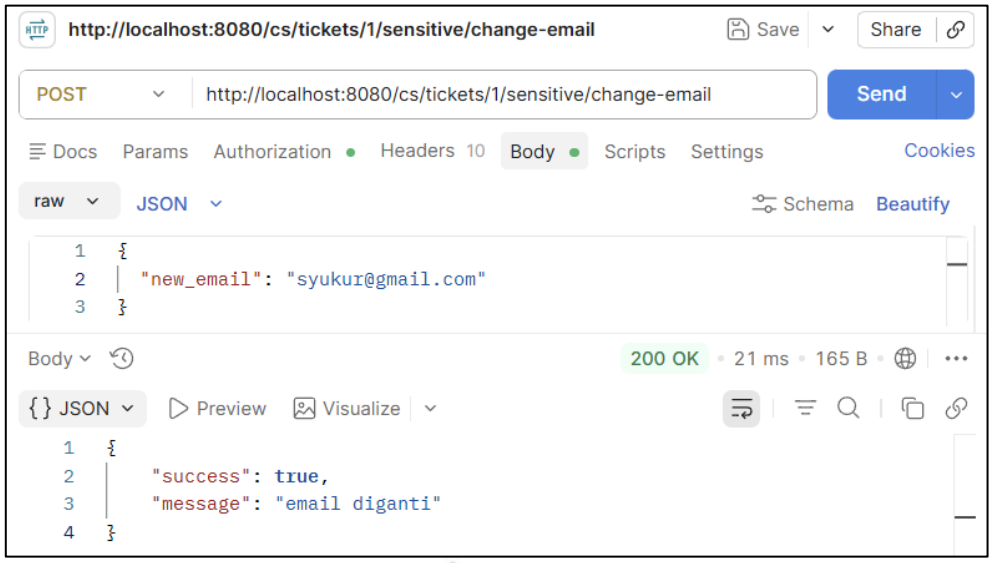
ticket dengan ID 1. Ticket tersebut telah ditugaskan kepada CS yang sama, ditunjukkan melalui nilai AssignedCSID = 1, dan berada pada status IN_PROGRESS. Kondisi tersebut memenuhi syarat agar request JIT dapat diproses oleh backend. Setelah konteks ticket terpenuhi, CS mengajukan request JIT untuk feature CHANGE_EMAIL melalui endpoint POST /cs/tickets/1/jit/request. Hasil pengujian menunjukkan bahwa backend menerima request dengan respons 201 Created dan menerbitkan session JIT sementara. Data pada tabel jit_sessions menunjukkan bahwa session dibuat untuk cs_id = 1, ticket_id = 1, feature CHANGE_EMAIL, memiliki nilai expired_at, dan berada dalam kondisi aktif melalui nilai is_active = 1. Bukti pembuatan session JIT dapat dilihat pada Gambar 4.17.



Gambar 4. 17 Request JIT Berhasil dan Session Dibuat

Hasil tersebut menunjukkan bahwa session JIT tidak diterbitkan hanya karena pengguna memiliki role Customer Support. Backend tetap memvalidasi ticket, assignment CS, status ticket, dan feature yang diminta sebelum session sementara dibuat.

Setelah session JIT aktif, pengujian dilanjutkan dengan menjalankan fitur sensitif CHANGE_EMAIL melalui endpoint POST /cs/tickets/1/sensitive/change-email. Pada skenario ini, CS mengirimkan data email baru melalui request API. Hasil pengujian menunjukkan bahwa backend menerima request dengan respons 200 OK dan pesan “email diganti”. Bukti eksekusi fitur sensitif saat session JIT aktif dapat dilihat pada Gambar 4.18.



Gambar 4. 18 Fitur CHANGE_EMAIL Berhasil Dijalankan Saat Session JIT Aktif

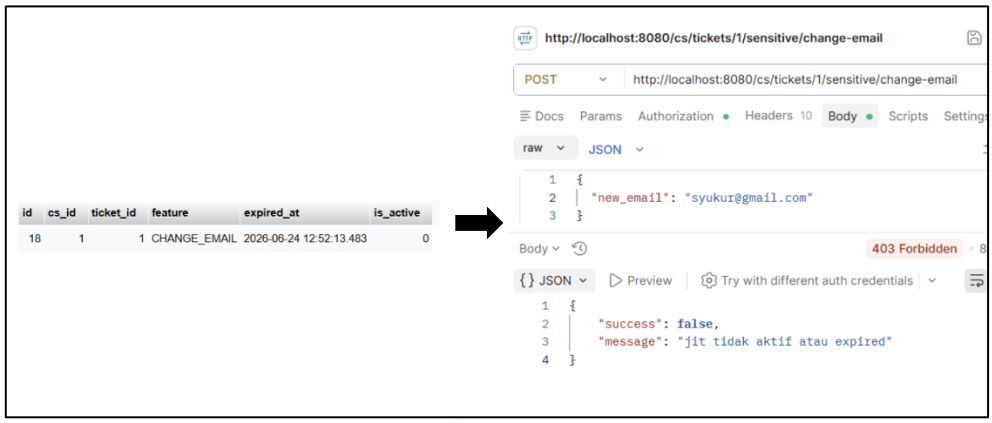
Hasil ini menunjukkan bahwa fitur sensitif dapat dijalankan apabila session JIT masih aktif, sesuai dengan CS yang sedang login, sesuai ticket, sesuai feature, belum digunakan, dan belum melewati batas waktu akses.

4.5.2 Pengujian Penolakan Akses Tanpa Session dan Setelah Session Tidak Aktif

Pengujian berikutnya dilakukan untuk memastikan bahwa fitur sensitif tidak dapat dijalankan tanpa session JIT yang valid. Pada skenario ini, CS mencoba menjalankan endpoint POST /cs/tickets/1/sensitive/change-email tanpa session JIT aktif yang sesuai. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 403 Forbidden. Hal ini menunjukkan bahwa role CS dan assignment ticket saja belum cukup untuk menjalankan fitur sensitif.

Pengujian juga dilakukan setelah session JIT digunakan atau sudah tidak aktif. Pada skenario ini, CS mencoba menjalankan kembali fitur CHANGE_EMAIL menggunakan session JIT yang sebelumnya telah digunakan. Berdasarkan data pada tabel jit_sessions, session untuk cs_id = 1, ticket_id = 1, dan feature CHANGE_EMAIL telah memiliki nilai is_active = 0. Ketika CS kembali memanggil endpoint POST /cs/tickets/1/sensitive/change-email, backend menolak request dengan respons 403 Forbidden dan pesan “jit tidak aktif atau

expired". Bukti penolakan akses setelah session JIT tidak aktif dapat dilihat pada Gambar 4.19.

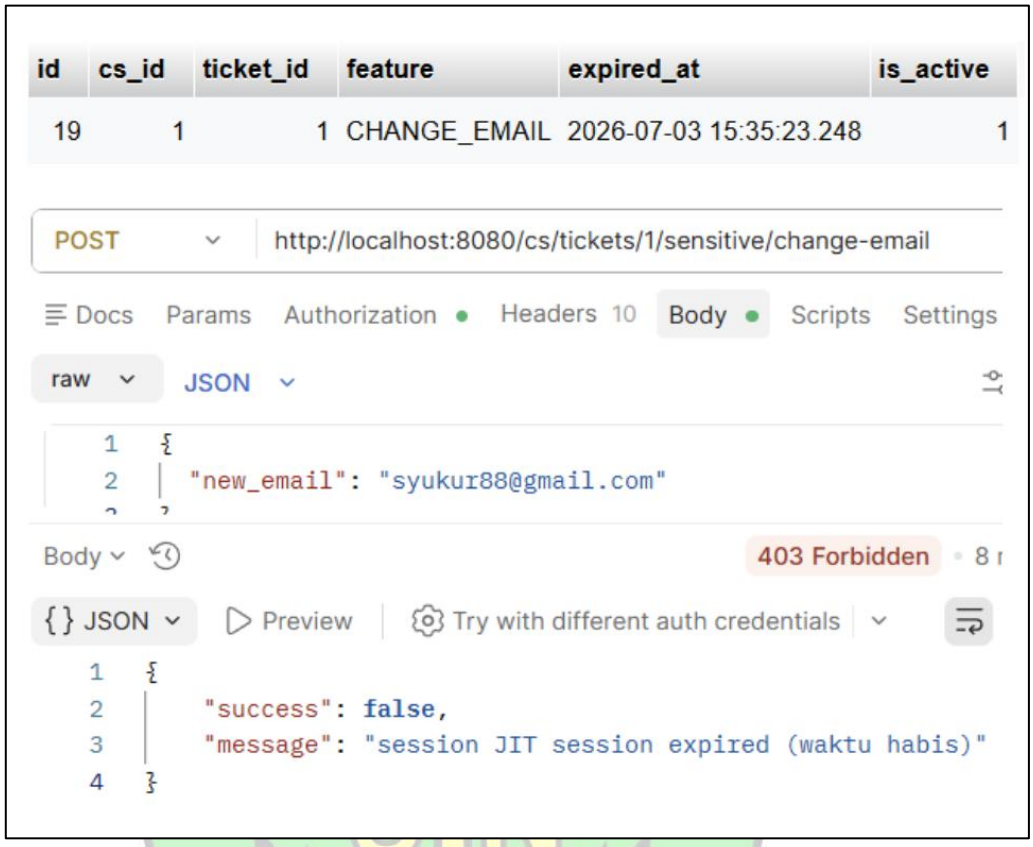


Gambar 4. 19 Penolakan Akses Setelah Session JIT Tidak Aktif

Hasil ini menunjukkan bahwa session JIT tidak dapat digunakan secara berulang. Dengan demikian, akses terhadap fitur sensitif bersifat sementara dan tidak berubah menjadi akses permanen bagi role Customer Support.

4.5.3 Pengujian Penolakan Akses karena Session Expired

Pengujian ini dilakukan untuk memverifikasi bahwa session *Just-In-Time Access* memiliki batas waktu penggunaan. Pada skenario ini, session JIT untuk feature CHANGE_EMAIL telah berhasil dibuat dan tercatat pada tabel jit_sessions dengan nilai is_active = 1. Selanjutnya, nilai expired_at dibiarkan melewati batas waktu akses atau disimulasikan menjadi waktu yang telah lewat. Pada saat request fitur sensitif dijalankan, waktu sistem telah melewati nilai expired_at yang tercatat pada session JIT, sehingga backend menganggap session tersebut sudah kedaluwarsa. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 403 Forbidden dan pesan bahwa session JIT telah expired. Meskipun session masih tercatat pada tabel jit_sessions, backend tetap menolak akses karena nilai expired_at telah melewati waktu saat request dijalankan. Bukti penolakan akses setelah session expired dapat dilihat pada Gambar 4.20.



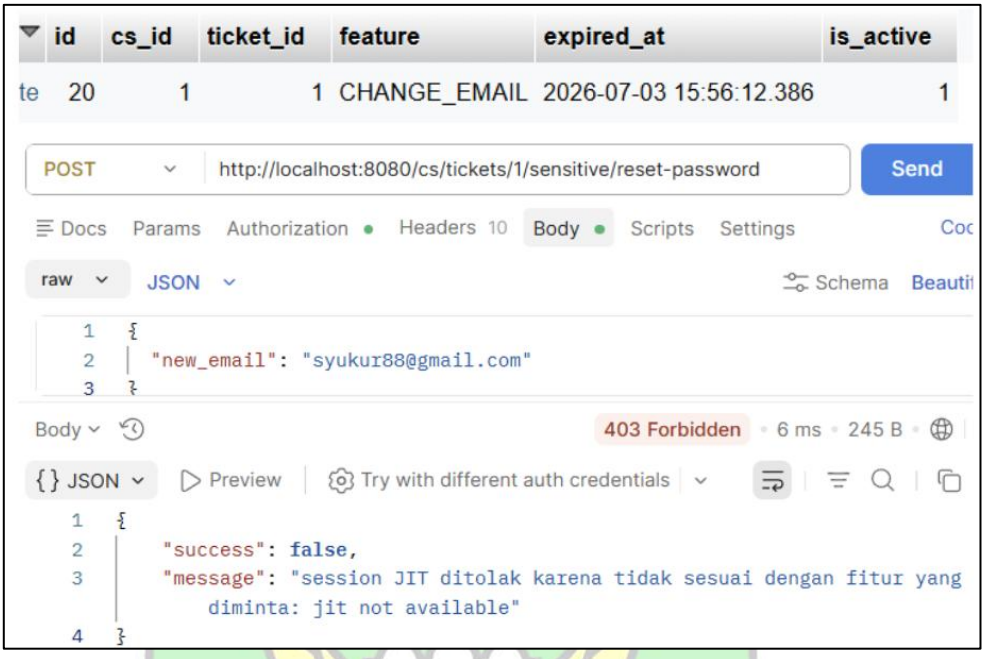
Gambar 4. 20 Penolakan Akses Setelah Session JIT Expired

Hasil ini menunjukkan bahwa validasi JIT tidak hanya bergantung pada keberadaan session atau status aktifnya, tetapi juga pada batas waktu akses yang tersimpan pada nilai expired_at. Dengan demikian, session JIT tidak berlaku sebagai akses permanen, melainkan hanya dapat digunakan selama durasi yang telah ditentukan pada backend.

4.5.4 Pengujian Penolakan Akses karena Feature Tidak Sesuai

Pengujian ini dilakukan untuk memastikan bahwa session *Just-In-Time Access* hanya berlaku untuk feature yang diminta saat session dibuat. Pada skenario ini, session JIT dibuat untuk feature CHANGE_EMAIL dan masih berada dalam kondisi aktif, ditunjukkan melalui nilai is_active = 1 pada tabel jit_sessions. Namun, session tersebut kemudian digunakan untuk menjalankan feature berbeda melalui endpoint POST /cs/tickets/1/sensitive/reset-password. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 403 Forbidden. Penolakan tersebut terjadi karena feature yang diminta pada endpoint tidak sesuai

dengan feature yang tersimpan pada session JIT. Dengan demikian, session JIT untuk CHANGE_EMAIL tidak dapat digunakan untuk menjalankan feature lain seperti RESET_PASSWORD. Bukti penolakan akses karena feature tidak sesuai dapat dilihat pada Gambar 4.21.



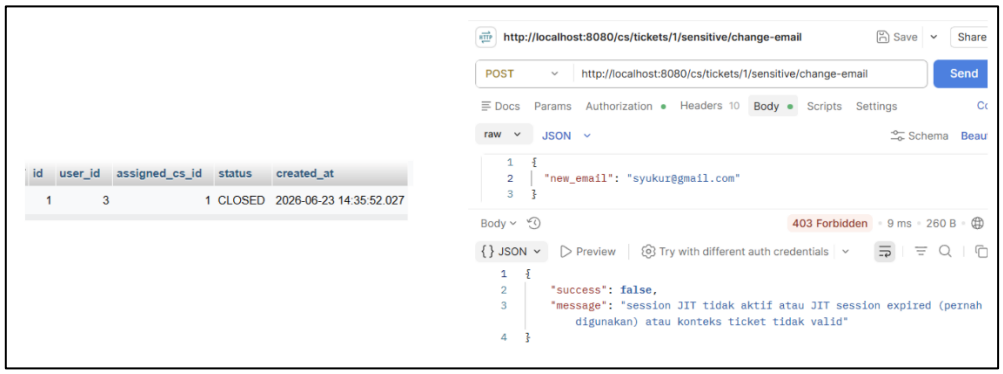
Gambar 4. 21 Penolakan Akses karena Feature Session JIT Tidak Sesuai

Hasil ini menunjukkan bahwa pembatasan JIT tidak hanya bergantung pada status aktif session, tetapi juga pada kesesuaian feature. Session JIT tidak berlaku secara umum untuk seluruh fitur sensitif, melainkan hanya berlaku untuk feature tertentu sesuai request JIT yang diterbitkan.

4.5.5 Pengujian Penolakan Akses pada Konteks Ticket Tidak Valid

Pengujian terakhir dilakukan untuk memastikan bahwa session JIT tetap bergantung pada konteks ticket. Pada skenario ini, session JIT dibuat ketika ticket masih berada pada status IN_PROGRESS. Setelah itu, status ticket berubah menjadi CLOSED, lalu CS mencoba menjalankan fitur sensitif kembali melalui endpoint POST /cs/tickets/1/sensitive/change-email. Hasil pengujian menunjukkan bahwa backend menolak request dengan respons 403 Forbidden. Pesan yang dikembalikan sistem menunjukkan bahwa session JIT tidak aktif,

expired, pernah digunakan, atau konteks ticket tidak valid. Bukti penolakan akses ketika ticket berada pada status CLOSED dapat dilihat pada Gambar 4.22.



Gambar 4. 22 Penolakan Eksekusi Fitur Sensitif pada Konteks JIT Tidak Valid

Hasil ini menunjukkan bahwa fitur sensitif tidak dapat dijalankan apabila konteks operasional ticket sudah tidak valid. Meskipun CS memiliki role yang sah dan endpoint dipanggil langsung melalui Postman, backend tetap memeriksa status ticket dan status session JIT sebelum fitur sensitif dieksekusi.

Secara keseluruhan, hasil pengujian *Just-In-Time Access* menunjukkan bahwa fitur sensitif tidak aktif secara permanen pada role Customer Support. Akses terhadap fitur sensitif hanya diberikan melalui session JIT yang valid, sesuai CS, sesuai ticket, sesuai feature, masih aktif, belum digunakan, dan belum melewati batas waktu akses. Dengan demikian, JIT berfungsi sebagai pembatas durasi dan kondisi akses terhadap fitur sensitif pada backend *Prototype*.

4.6 Analisis Hasil Pengujian

Berdasarkan hasil pengujian yang telah dilakukan, mekanisme kontrol akses pada *Prototype* berjalan sesuai dengan rancangan penelitian. Pengujian dilakukan melalui akses langsung menggunakan Postman/API client untuk memastikan bahwa pembatasan akses benar-benar ditegakkan pada backend *Access control layer*, bukan hanya melalui pembatasan tampilan pada frontend. Hasil pengujian pada matriks menunjukkan bahwa setiap skenario menghasilkan actual result yang sesuai dengan expected result, baik pada pengujian RBAC, *Least Privilege*, maupun *Just-In-Time Access*.

Hasil pengujian *Role-Based Access Control* menunjukkan bahwa RBAC berfungsi sebagai batas awal otorisasi berdasarkan role pengguna. User, Customer Support, dan Administrator hanya dapat mengakses area yang sesuai dengan kewenangannya. Akses yang sesuai role diterima oleh backend, sedangkan akses lintas role ditolak. Namun, RBAC pada penelitian ini tidak diposisikan sebagai mekanisme tunggal, karena RBAC dasar hanya membedakan akses berdasarkan peran dan belum cukup untuk membatasi ruang lingkup serta durasi akses pengguna internal terhadap data dan fitur sensitif.

Hasil pengujian *Least Privilege* menunjukkan bahwa akses Customer Support tidak hanya ditentukan oleh role, tetapi juga oleh konteks assignment ticket. Customer Support tidak memiliki akses global terhadap seluruh ticket atau seluruh data pengguna. CS hanya dapat melakukan claim terhadap ticket yang tersedia, membuka ticket yang ditugaskan kepadanya, dan ditolak ketika mencoba mengambil atau membuka ticket yang sudah ditugaskan kepada CS lain. Hal ini menunjukkan bahwa *Least Privilege* memperkuat RBAC dasar dengan membatasi ruang lingkup akses pengguna internal berdasarkan konteks tugas yang sedang dijalankan.

Hasil pengujian *Just-In-Time Access* menunjukkan bahwa fitur sensitif tidak aktif secara permanen pada role Customer Support. Fitur sensitif seperti VIEW_KYC, RESET_PASSWORD, CHANGE_EMAIL, RESET_PIN, dan UNBLOCK_ACCOUNT dikendalikan melalui session JIT. Pada pengujian, fitur representatif CHANGE_EMAIL hanya dapat dijalankan ketika session JIT valid, sesuai CS, sesuai ticket, sesuai feature, masih aktif, belum digunakan, dan belum melewati batas waktu akses. Request ditolak ketika session tidak tersedia, sudah digunakan, expired, feature tidak sesuai, atau konteks ticket sudah tidak valid. Dengan demikian, *Just-In-Time Access* memperkuat RBAC dasar dengan membatasi durasi dan kondisi penggunaan fitur sensitif melalui session sementara.

Sintesis hasil evaluasi mekanisme kontrol akses berlapis dapat dilihat pada Tabel 4.2.

Tabel 4. 2 Sintesis Hasil Evaluasi Mekanisme Kontrol Akses Berlapis

Aspek Evaluasi	Bentuk Pembatasan	Hasil Pengujian	Peran terhadap RBAC Dasar
Role-Based Access Control	Membatasi akses awal berdasarkan role pengguna	Akses sesuai role diterima, sedangkan akses lintas role ditolak oleh backend	Menjadi batas awal otorisasi berdasarkan role
Least Privilege	Membatasi ruang lingkup akses CS berdasarkan assignment ticket	CS dapat claim dan membuka ticket miliknya, tetapi tidak dapat membuka ticket milik CS lain	Memperkuat RBAC dengan membatasi scope akses berdasarkan konteks ticket
Just-In-Time Access	Membatasi fitur sensitif melalui session sementara	Fitur sensitif hanya dapat dijalankan ketika session JIT valid; request ditolak ketika session tidak aktif atau konteks tidak sesuai	Memperkuat RBAC dengan membatasi durasi dan kondisi penggunaan fitur sensitif
Backend Enforcement	Memastikan validasi akses dilakukan di sisi backend	Request tidak sah yang dikirim langsung melalui Postman/API client tetap ditolak	Membuktikan bahwa pembatasan akses tidak hanya bergantung pada frontend

Berdasarkan sintesis tersebut, hasil pengujian memverifikasi perbandingan konseptual yang telah dijelaskan pada Tabel 2.2. Pada model RBAC dasar, akses cenderung ditentukan oleh role. Setelah diperkuat dengan *Least Privilege* dan *Just-In-Time Access*, akses tidak hanya bergantung pada role, tetapi juga dibatasi oleh assignment ticket, status ticket, feature, session sementara, dan waktu kedaluwarsa. Dengan demikian, kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* menunjukkan pembatasan akses yang lebih kontekstual dibandingkan RBAC dasar pada *Prototype* ini.

Hasil penelitian ini juga menunjukkan perbedaan fokus dengan beberapa penelitian terdahulu yang banyak membahas *Least Privilege* dan *Just-In-Time Access* pada konteks IAM, PAM, cloud, atau infrastruktur enterprise. Penelitian ini tidak mengembangkan mekanisme baru di luar konsep tersebut, tetapi memodelkan

penerapannya pada backend application *layer* melalui konteks Customer Support, ticket, fitur sensitif, dan session JIT. Dengan demikian, kontribusi penelitian ini berada pada bentuk implementasi dan verifikasi mekanisme kontrol akses berlapis pada level aplikasi.

Penerapan mekanisme berlapis ini memiliki beberapa konsekuensi. Dari sisi keamanan akses, kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* membuat akses internal menjadi lebih terkendali karena role CS tidak langsung memberikan akses global terhadap ticket maupun fitur sensitif. Namun, dari sisi implementasi, mekanisme ini menambah proses validasi pada backend. Setiap request tertentu perlu melewati pemeriksaan role, assignment ticket, status ticket, session JIT, feature, dan expired_at. Selain itu, CS harus melakukan request JIT terlebih dahulu sebelum menggunakan fitur sensitif, sehingga terdapat tambahan langkah operasional dibandingkan akses langsung berbasis role saja. Hasil penelitian ini tidak dimaksudkan untuk menyatakan bahwa *Prototype* telah aman secara absolut atau siap digunakan sebagai sistem produksi berskala industri. *Prototype* ini digunakan sebagai media akademik untuk memodelkan dan memverifikasi penerapan kontrol akses internal pada *layer* aplikasi. Pengujian juga masih dilakukan pada lingkungan *Prototype* dengan skenario terbatas, sehingga belum mencakup pengujian performa, load testing, penetration testing, keamanan infrastruktur, tamper-resistant audit log, maupun integrasi finansial nyata.

Dengan batasan tersebut, fokus evaluasi berada pada pembuktian bahwa kombinasi RBAC, *Least Privilege*, dan *Just-In-Time Access* berjalan sesuai rancangan dalam membatasi akses pengguna internal berdasarkan role, konteks ticket, dan session sementara. Hasil pengujian menunjukkan bahwa mekanisme kontrol akses berlapis pada backend *Prototype* mampu membatasi lingkup dan durasi akses pengguna internal sebagai penguatan terhadap model RBAC dasar.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, diperoleh beberapa kesimpulan sebagai berikut:

1. Kebutuhan kontrol akses pengguna internal berhasil diidentifikasi melalui tiga aktor, yaitu User, Customer Support, dan Administrator, dengan Customer Support sebagai aktor utama. Data sensitif dibatasi berdasarkan konteks *ticket*, sedangkan fitur sensitif dikendalikan melalui *Just-In-Time Access* agar tidak tersedia secara permanen.
2. *Least Privilege* dan *Just-In-Time Access* berhasil dirancang dan diimplementasikan pada backend. *Least Privilege* membatasi Customer Support agar hanya dapat mengakses *ticket* yang menjadi *assignment*-nya, sedangkan *Just-In-Time Access* membatasi fitur sensitif melalui *session* sementara yang hanya diberikan pada konteks *ticket* yang valid, berlaku selama 15 menit, dan hanya dapat digunakan satu kali. Seluruh validasi akses ditegakkan pada sisi backend.
3. Evaluasi melalui 19 skenario pengujian menunjukkan seluruh hasil aktual sesuai dengan hasil yang diharapkan. RBAC membatasi akses berdasarkan *role*, *Least Privilege* membatasi lingkup akses berdasarkan *assignment ticket*, dan *Just-In-Time Access* membatasi durasi serta kondisi penggunaan fitur sensitif. Kombinasi tersebut menghasilkan kontrol akses yang lebih kontekstual dibandingkan RBAC dasar dalam ruang lingkup *Prototype* yang dikembangkan.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran untuk pengembangan selanjutnya, yaitu:

1. Penelitian selanjutnya dapat mengeksplorasi pendekatan kontrol akses hibrida dengan mengombinasikan *Role-Based Access Control* (RBAC), *Least Privilege*, dan *Just-In-Time Access* dengan model kontrol akses lain, seperti Attribute-Based

Access Control (ABAC). Dengan pendekatan tersebut, keputusan akses tidak hanya didasarkan pada role, tetapi juga dapat mempertimbangkan atribut pengguna, objek, tindakan, maupun konteks operasional sehingga mekanisme kontrol akses menjadi lebih adaptif pada sistem yang lebih kompleks.

2. Penelitian selanjutnya dapat menerapkan mekanisme kontrol akses pada lingkungan yang lebih mendekati kondisi operasional nyata, seperti arsitektur microservices, sistem terdistribusi, atau integrasi dengan layanan eksternal. Pengembangan tersebut dapat memberikan gambaran mengenai penerapan mekanisme kontrol akses pada lingkungan sistem yang lebih kompleks.
3. Penelitian selanjutnya juga dapat menguji mekanisme kontrol akses ini menggunakan pendekatan keamanan yang lebih mendalam, seperti threat modeling, penetration testing, atau pengujian penyalahgunaan akses internal yang lebih variatif.

DAFTAR PUSTAKA

Atlam, H. F., & Yang, Y. (2025). Enhancing Healthcare Security: A Unified RBAC and ABAC Risk-Aware *Access* Control Approach. *Future Internet*, *17*(6), 262. https://doi.org/10.3390/fi17060262

Bates, C., Joseph, C., Tate, Z., Walsh, M., Leonard, P., & Oluwabukunmi Okunola, C. (2025). *Advanced Security Controls for Protecting Sensitive Data in Database as a Service Systems Literature review 1. Introduction & scope*.

Behringer, F., & Baumann, P. (2025). Integrating identity and *Access* management and privileged *Access* management for enhanced identity security in financial institutions: A zero-trust approach. *Cyber Security: A Peer-Reviewed Journal*, *9*(2), 114. https://doi.org/10.69554/YRMM1823

Deliu, D., & Moldovan, A. (2025). *Fintech 5.0: From Shadows to Clarity—The Future of Transparency in Financial Organizations* (hlm. 285–347). https://doi.org/10.1007/978-3-032-03523-3_12

Ebad, S. A., & Amara, M. (2026). The Principle of *Least Privilege* in Microservices: A Systematic Mapping Study. *Applied Sciences*, *16*(3), 1495. https://doi.org/10.3390/app16031495

Farhadighalati, N., Estrada-Jimenez, L. A., Nikghadam-Hojjati, S., & Barata, J. (2025). A Systematic Review of *Access* Control Models: Background, Existing Research, and Challenges. *IEEE Access*, *13*, 17777–17806. https://doi.org/10.1109/ACCESS.2025.3533145

Gupta, A., Dwivedi, D. N., & Shah, J. (2023). *Overview of Technology Solutions* (hlm. 25–39). https://doi.org/10.1007/978-981-99-2571-1_3

Haber, M. J., & Rolls, D. (2020). *Just-In-Time Access* Management. Dalam *Identity Attack Vectors* (hlm. 151–155). Apress. https://doi.org/10.1007/978-1-4842-5165-2_14

Kraiwanit, T., Limna, P., & Wattanasin, P. (2024). Digital wallet dynamics: Perspectives on potential Worldcoin adoption factors in a developing country's

FinTech Sector. *Journal of Open Innovation: Technology, Market, and Complexity*, *10*(2), 100287. https://doi.org/10.1016/j.joitmc.2024.100287

Kumar, P. (2025). Next-generation secure authentication and *Access* control architectures: advanced techniques for securing distributed systems in modern enterprises. *International Journal of Computational and Experimental Science and Engineering*, *11*(3). https://doi.org/10.22399/ijcesen.3294

Mohamed, I. A., Abdulla, H., Mohabilasha, M. M., Ahmed, F., Chandre, P., & Mahalle, P. (2026). *reKYC for Fintech Security: A Privacy-Preserving Approach to Digital Wallet Verification* (hlm. 420–434). https://doi.org/10.1007/978-3-032-13203-1_40

Olaide, A., Kalilu, J., & Ajani, B. (2025). *Least Privilege Access in Practice: Minimising Insider Risk While Maintaining Operational Efficiency*.

Olorunlana, T. J. (2025). *Least Privilege* and *Access* Control Principles in Enterprise Network Security: A Comparative Analysis of the United States and Global Practices. *International Journal of Science Architecture Technology and Environment*, 1. https://doi.org/10.63680/ijsate1225001.001

Omotunde, H., & Ahmed, M. (2023). A Comprehensive Review of Security Measures in Database Systems: Assessing Authentication, *Access* Control, and Beyond. *Mesopotamian Journal of CyberSecurity*, *2023*, 115–133. https://doi.org/10.58496/MJCSC/2023/016

Park, J.-H., Park, S.-C., & Youm, H.-Y. (2025). A Proposal for a Zero-Trust-Based Multi-Level Security Model and Its Security Controls. *Applied Sciences*, *15*(2), 785. https://doi.org/10.3390/app15020785

Penta Security. (2025). *FinWise Data Breach: Encryption Becomes Your Last Line of Defense*. https://www.pentasecurity.com/press-releases/finwise-data-breach-encryption/

Plachkinova, M., & Knapp, K. (2023). *Least Privilege* across People, Process, and Technology: Endpoint Security Framework. *Journal of Computer Information Systems*, *63*(5), 1153–1165. https://doi.org/10.1080/08874417.2022.2128937

Prajapati, V. (2025). Role of Identity and *Access* Management in Zero Trust Architecture for Cloud Security: Challenges and Solutions. *International Journal of Advanced Research in Science, Communication and Technology*, 6–18. https://doi.org/10.48175/IJARSCT-23902

Ragothaman, K., Wang, Y., Rimal, B., & Lawrence, M. (2023). *Access* Control for IoT: A Survey of Existing Research, Dynamic Policies and Future Directions. *Sensors*, *23*(4), 1805. https://doi.org/10.3390/s23041805

Raju, A. B. A. (2021). Zero Trust Architecture for Financial Institutions. *International Journal For Multidisciplinary Research*, *3*(5). https://doi.org/10.36948/ijfmr.2021.v03i05.27000

Rao, S. A. G. (2025). Strategic Value of *Just-In-Time Access* Control: Enhancing Security While Driving Workforce Efficiency in Large-Scale Organizations. *Hampton Global Business Review*. https://doi.org/10.70924/uv4px7jt/qhs81bg2

Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). *Zero Trust Architecture*. https://doi.org/10.6028/NIST.SP.800-207

Subramanyam, S. P. (2025). Zero Trust in Practice: Enhancing Privileged *Access* Security with *Just-In-Time* (JIT) and Self-Service Models. *International Research Journal of Innovations in Engineering and Technology*, *09*(12), 13–18. https://doi.org/10.47001/IRJIET/2025.912003

The Straits Times. (2022). *Former DBS officer jailed for illegally accessing and leaking customer details | The Straits Times*. https://www.straitstimes.com/singapore/courts-crime/former-dbs-officer-jailed-for-illegally-accessing-and-leaking-customer-details

Usman, U. I., Ibrahim, M. D., Aliyu, I. A., Isah, U., & Jamal, F. (2025). The Implications of Insider Threats in Financial Institutions: A Review of Challenges and Mitigation Strategies. *International Journal of Innovative Science and Research Technology*, 186. https://doi.org/10.38124/ijisrt/25dec240

Velur, N. (2025). The Evolution of Enterprise Security Systems in Cloud-native Architectures: From Perimeter-Based to Automated, Granular Models. *Journal of Information Systems Engineering and Management*, *10*(63s), 1457–1464. https://doi.org/10.52783/jisem.v10i63s.14113

Verizon. (2024). *2024 Data Breach Investigations Report*. https://www.verizon.com/business/resources/reports/2024-dbir-data-breach-investigations-report.pdf

Waheed, U., Khan, S. A., Masud, M., Jamshed, H., Jumani, T. A., & Malik, N. U. R. (2025). Blockchain-Based, Dynamic Attribute-Based *Access* Control for Smart Home Energy Systems. *Energies*, *18*(8), 1973. https://doi.org/10.3390/en18081973

DAFTAR ISTILAH

Berikut adalah daftar istilah teknis beserta penjelasannya yang digunakan dalam penelitian ini.

***Access* Control**

Mekanisme untuk mengatur dan membatasi siapa saja yang dapat mengakses data, fitur, atau sumber daya tertentu dalam sistem.

Administrator

Pengguna internal yang berperan dalam pengelolaan sistem dan pemantauan aktivitas pada *Prototype*.

API

Singkatan dari Application Programming Interface, yaitu antarmuka yang digunakan agar client dapat berkomunikasi dengan backend melalui request dan response.

API Client

Aplikasi yang digunakan untuk mengirim request langsung ke backend, misalnya Postman.

Assignment Ticket

Hubungan penugasan antara ticket tertentu dengan Customer Support tertentu. Dalam penelitian ini, assignment ticket menjadi dasar pembatasan akses *Least Privilege*.

Autentikasi

Proses verifikasi identitas pengguna sebelum pengguna dapat mengakses sistem.

Backend

Bagian sistem yang berjalan di sisi server dan bertugas memproses request, menjalankan logika aplikasi, mengakses database, serta mengembalikan response.

Backend *Access control layer*

Lapisan kontrol akses pada backend yang memvalidasi role, assignment ticket, session JIT, feature, dan status akses sebelum request diproses.

Backend Enforcement

Penegakan aturan akses pada sisi backend, sehingga request yang tidak sah tetap ditolak meskipun endpoint dipanggil langsung melalui API client.

Black-Box Testing

Metode pengujian yang menilai hasil keluaran sistem berdasarkan input atau skenario tertentu tanpa menilai struktur kode internal secara langsung.

Claim Ticket

Proses ketika Customer Support mengambil ticket yang tersedia sehingga ticket tersebut ditugaskan kepada CS yang bersangkutan.

Contextual Auto-Approval

Mekanisme pemberian session JIT secara otomatis oleh backend berdasarkan validasi konteks, tanpa persetujuan manual Administrator.

Customer Support

Pengguna internal yang bertugas menangani ticket bantuan, berinteraksi dengan data pengguna, dan dapat mengajukan akses sementara untuk fitur sensitif melalui JIT.

Data KYC

Data identitas pengguna yang digunakan dalam proses Know Your Customer dan dikategorikan sebagai data sensitif.

Endpoint

Alamat API tertentu yang digunakan client untuk mengakses fungsi atau data pada backend.

Insider Threat

Risiko keamanan yang berasal dari pengguna internal yang memiliki akses sah tetapi menyalahgunakan hak akses tersebut.

JWT

Singkatan dari JSON Web Token, yaitu token yang digunakan untuk membawa informasi login, identitas pengguna, dan role pada request backend.

***Just-In-Time Access* (JIT)**

Mekanisme pemberian akses sementara untuk fitur atau data sensitif hanya ketika diperlukan dan dalam durasi tertentu. Setelah durasi berakhir atau akses digunakan, hak akses dinonaktifkan kembali.

***Least Privilege* (LP)**

Prinsip pemberian hak akses minimum kepada pengguna sesuai kebutuhan tugasnya. Akses yang tidak relevan dengan tugas pengguna tidak diberikan.

Layer

Lapisan fungsional dalam arsitektur aplikasi yang memiliki tanggung jawab tertentu. Dalam penelitian ini, *layer* digunakan untuk menjelaskan posisi penerapan kontrol akses pada sisi backend sebelum *request* diteruskan ke logika aplikasi.

Middleware

Komponen backend yang memproses request sebelum diteruskan ke fungsi utama, misalnya untuk validasi JWT, RBAC, *Least Privilege*, dan JIT.

Postman

Aplikasi API client yang digunakan untuk menguji endpoint backend secara langsung, termasuk skenario akses valid dan akses ditolak.

Privilege

Hak akses yang dimiliki pengguna untuk melakukan tindakan tertentu dalam sistem.

Prototype

Bentuk sistem percobaan yang dibangun untuk memodelkan, mengimplementasikan, dan menguji mekanisme kontrol akses dalam penelitian.

RBAC Dasar

Model RBAC yang hanya membedakan akses berdasarkan role, tetapi belum membatasi akses berdasarkan konteks ticket atau durasi penggunaan fitur sensitif.

Scenario-Based Testing

Metode pengujian yang menggunakan skenario tertentu untuk memeriksa apakah sistem berjalan sesuai rancangan.

Standing Privilege

Kondisi ketika hak akses melekat secara permanen pada pengguna atau role, meskipun akses tersebut tidak selalu dibutuhkan.

Status Ticket

Kondisi ticket dalam alur penanganan, seperti OPEN, CLAIMED, IN_PROGRESS, RESOLVED, atau CLOSED.

403 Forbidden

Kode status HTTP yang menunjukkan bahwa request ditolak karena pengguna tidak memiliki hak akses yang sesuai.

201 Created

Kode status HTTP yang menunjukkan bahwa resource baru berhasil dibuat, misalnya session JIT.

200 OK

Kode status HTTP yang menunjukkan bahwa request berhasil diproses oleh backend.

409 Conflict

Kode status HTTP yang menunjukkan bahwa request tidak dapat diproses karena terjadi konflik dengan kondisi resource saat ini, misalnya ticket telah ditugaskan kepada Customer Support lain.

Lampiran 1 Repositori Source Code

Source code *Prototype* sistem yang digunakan dalam penelitian ini tersedia pada repositori GitHub berikut:

Nama Repositori : backend-skripsi
Platform : GitHub
Alamat Repositori : https://github.com/SyukurGit/backend-skripsi
Branch Utama : main

Repositori tersebut memuat source code implementasi *Prototype* sistem dompet digital yang digunakan dalam penelitian, khususnya implementasi kontrol akses pengguna internal melalui *Role-Based Access Control* (RBAC), prinsip *Least Privilege*, dan mekanisme *Just-In-Time* Access. Repositori juga memuat komponen pendukung implementasi seperti pengelolaan ticket, session JIT, database, middleware, audit log, serta konfigurasi aplikasi.