PENGEMBANGAN RESTFUL API PADA SISTEM REMUNERASI DOSEN UNIVERSITAS LAMPUNG BIDANG PENDIDIKAN DAN PENELITIAN DENGAN AUTENTIKASI JSON WEB TOKEN (JWT)

(Skripsi)

Oleh

SHAFA AULIYA
NPM 2117051042

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

PENGEMBANGAN RESTFUL API PADA SISTEM REMUNERASI DOSEN UNIVERSITAS LAMPUNG BIDANG PENDIDIKAN DAN PENELITIAN DENGAN AUTENTIKASI JSON WEB TOKEN (JWT)

Oleh

SHAFA AULIYA

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

PENGEMBANGAN RESTFUL API PADA SISTEM REMUNERASI DOSEN UNIVERSITAS LAMPUNG BIDANG PENDIDIKAN DAN PENELITIAN DENGAN AUTENTIKASI JSON WEB TOKEN (JWT)

Oleh

SHAFA AULIYA

Sistem remunerasi dosen Universitas Lampung berperan penting dalam mengelola kinerja dosen, khususnya pada bidang pendidikan dan penelitian. Namun, keterbatasan integrasi data pada sistem yang ada menyebabkan pemanfaatan informasi kinerja dosen menjadi kurang optimal, terutama untuk mendukung proses akreditasi, evaluasi kinerja, dan pelaporan institusional. Penelitian ini bertujuan untuk mengembangkan RESTful API yang terintegrasi pada sistem remunerasi dosen agar data kinerja dapat dikelola dan diakses secara lebih efektif. API dirancang dengan menerapkan autentikasi JSON Web Token (JWT) untuk menjaga keamanan data dan memastikan hanya pengguna berwenang yang dapat mengakses informasi. Metode pengembangan menggunakan pendekatan Node.js dan basis data PostgreSQL, dengan pengujian dilakukan melalui metode *white-box* untuk memastikan keakuratan logika dan keandalan sistem. Hasil pengujian menunjukkan bahwa API yang dikembangkan mampu mengelola data secara konsisten dan memberikan respons yang cepat terhadap permintaan pengguna. Pengembangan ini diharapkan dapat meningkatkan efisiensi pengelolaan data kinerja dosen, memperkuat akurasi informasi, serta mendukung kebutuhan analisis dan pengambilan keputusan di Universitas Lampung.

Kata kunci: Sistem Remunerasi Dosen, RESTful API, JSON Web Token, Integrasi Data, Pendidikan, Penelitian, Universitas Lampung.

ABSTRACT

RESTFUL API DEVELOPMENT IN THE LECTURER REMUNERATION SYSTEM OF THE UNIVERSITY OF LAMPUNG IN THE FIELD OF EDUCATION AND RESEARCH WITH JSON WEB TOKEN (JWT) AUTHENTICATION

BY

SHAFA AULIYA

The lecturer remuneration system at the University of Lampung plays an important role in managing lecturer performance, particularly in the fields of education and research. However, the limited data integration in the existing system has resulted in suboptimal utilization of lecturer performance information, especially in supporting accreditation processes, performance evaluations, and institutional reporting. This study aims to develop an integrated RESTful API for the lecturer remuneration system to enable more effective data management and access. The API is designed by implementing JSON Web Token (JWT) authentication to ensure data security and restrict access to authorized users only. The development method uses a Node.js approach with a PostgreSQL database, and testing is carried out using the white-box method to ensure logical accuracy and system reliability. The test results show that the developed API can manage data consistently and provide fast responses to user requests. This development is expected to improve the efficiency of lecturer performance data management, enhance information accuracy, and support analysis and decision-making needs at the University of Lampung.

Keywords: Lecturer Remuneration System, RESTful API, JSON Web Token, Data Integration, Education, Research, University of Lampung.

Judul Skripsi	:	**PENGEMBANGAN RESTFUL API PADA SISTEM REMUNERASI DOSEN UNIVERSITAS LAMPUNG BIDANG PENDIDIKAN DAN PENELITIAN DENGAN AUTENTIKASI JSON WEB TOKEN (JWT)**
Nama Mahasiswa	:	Shafa Auliya
Nomor Pokok Mahasiswa	:	2117051042
Program Studi	:	Ilmu Komputer (S-1)
Fakultas	:	Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing

M. Iqbal Parabi, S.SI., M.T.	**Igit Sabda Ilman, M.Kom**
NIP. 199011302015041002	NIK. 232111960101101

MENGETAHUI

2. Ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung	3. Ketua Program Studi Ilmu Komputer FMIPA Universitas Lampung
Dwi Sakethi, S.Si., M.Kom.	**Tristiyanto, S.Kom., M.I.S., Ph.D.**
NIP. 196806111998021001	NIP. 198104142005011001

MENGESAHKAN

1. Tim Penguji

   Ketua : **M. Iqbal Parabi, S.SI., M.T.**

   Sekertaris : **Igit Sabda Ilman, M.Kom.**

   Penguji Utama : **Tristiyanto, S.Kom., M.I.S., Ph.D.**

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.
NIP. 197110012005011002

Tanggal Lulus Ujian Skripsi : **11 Agustus 2025**

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Shafa Auliya

NPM : 2117051042

Menyatakan bahwa skripsi saya yang berjudul **"Pengembangan RESTful API pada Sistem Remunerasi Dosen Universitas Lampung Bidang Pendidikan dan Penelitian dengan Autentikasi JSON Web Token (JWT)"** merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang di skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti skripsi saya merupakan hasil penjiplakan atau dibuat orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar yang telah saya terima.

Bandar Lampung, 28 Agustus 2025

Penulis,

Shafa Auliya

NPM. 2117051042

RIWAYAT HIDUP

Penulis dilahirkan di Ganjar Agung pada tanggal 10 Agustus 2003 sebagai anak pertama dari dua bersaudara dari Bapak Supriyanto dan Ibu Khotimah. Riwayat pendidikan penulis dimulai dari jenjang pendidikan dasar di SD Negeri 6 Metro pada Tahun 2015, kemudian melanjutkan pendidikan menengah pertama di SMP Negeri 2 Metro pada Tahun 2018. Selanjutnya pendidikan menengah atas di SMA Negeri 3 Metro pada Tahun 2021. Penulis kemudian melanjutkan pendidikan tinggi dengan terdaftar sebagai mahasiswa Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN) pada Tahun 2021.

Selama menjadi mahasiswa, penulis aktif mengikuti berbagai kegiatan antara lain:

1. Menjadi Anggota Muda Ilmu Komputer (ADAPTER) Himpunan Mahasiswa Jurusan Ilmu Komputer pada Tahun 2021.
2. Menjadi Anggota Bidang Keilmuan Himpunan Mahasiswa Jurusan Ilmu Komputer pada Tahun 2022.
3. Menjadi Asisten Dosen mata kuliah Logika di Jurusan Ilmu Komputer pada Tahun 2022.

4. Menjadi Bendahara Bidang Keilmuan Himpunan Mahasiswa Jurusan Ilmu Komputer pada Tahun 2023.
5. Menjadi Panitia Acara Pekan Raya Jurusan Ilmu Komputer Divisi Esai Nasional pada Tahun 2023.
6. Melaksanakan Kerja Praktik di PT Jasaraharja Putera Cabang Bandar Lampung Divisi *Claim Servis* pada Tahun 2023-2024.
7. Mengikuti Studi Independen MSIB di Infinite Learning Bidang *Web Development* pada Tahun 2024.
8. Menjadi Asisten Dosen mata kuliah Pemrosesan Data Terdistribusi di Jurusan Ilmu Komputer pada Tahun 2024.
9. Mengikuti Kuliah Kerja Nyata (KKN) Universitas Lampung Periode 2 di Desa Sumur Kucing, Kecamatan Pasir Sakti, Kabupaten Lampung Timur pada Tahun 2024.

MOTTO

''Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya''
(Q.S. Al-Baqarah Ayat 286)

''Ambillah satu langkah menuju Aku, maka Aku akan mengambil sepuluh langkah ke arahmu. Berjalanlah ke arah-Ku, maka Aku akan berlari ke arahmu''
(Firman Allah dalam Hadits Qudsi)

''Kamu akan menemukan banyak aspek yang akan membuat kamu berpaling daripada ilmu, disaat kamu sudah memutuskan untuk berazam menuntut ilmu''
(Maulana Syaikh Husam Ramadhan)

''Tujuan pendidikan itu untuk mempertajam kecerdasan, memperkeruh kemauan serta memperhalus perasaan''
(Tan Malaka)

''Tidak ada mimpi yang terlalu tinggi. Tak ada mimpi yang patut diremehkan. Lambungkan setinggi yang kau inginkan dan gapailah dengan selayaknya yang kau harapkan''
(Maudy Ayunda)

PERSEMBAHAN

Alhamdulillahirobbilalamin

Puji Syukur kehadirat Allah Subhanahu Wa Ta'ala atas segala Rahmat dan karunia-Nya sehingga skripsi ini dapat diselesaikan dengan sebaik-baiknya. Shalawat serta salam selalu tercurahkan kepada junjungan Nabi Agung Muhammad Shallallahu 'Alaihi Wassalam.

Kupersembahkan karya ini kepada:

Kedua Orang Tuaku Tercinta

Dengan segenap cinta dan penuh hormat, kuhaturkan terima kasih yang tak berujung atas setiap doa, dukungan dan kasih sayang yang tak pernah mengenal lelah. Setiap pengorbanan dan perjuanganmu dalam mendidik serta membesarkanku adalah cahaya yang menuntunku melangkah, menjadi kekuatan terbesar dalam setiap perjalanan hidupku. Terima kasih pula, karena dalam setiap helaan napasmu, selalu kau langitkan doa-doa terbaikmu, memohon keberkahan, kebaikan dan kebahagiaanku di setiap langkahku.

Seluruh Keluarga Besar Ilmu Komputer 2021

Atas kebersamaan, dukungan dan bantuan yang senantiasa tersimpan dalam setiap langkah, terima kasih telah menjadi bagian dari perjalanan ini.

Almamater Tercinta, Universitas Lampung dan Jurusan Ilmu Komputer

Tempat menimba ilmu, menanam mimpi dan menumbuhkan harapan yang kelak menjadi bekal berharga dalam setiap perjalanan kehidup

SANWACANA

Puji Syukur penulis panjatkan ke hadirat Allah Subhanahu Wa Ta'ala, Tuhan Yang Maha Esa, atas segala rahmat, karunia, dan hidayah-Nya. Shalawat serta salam senantiasa tercurahkan kepada junjungan Nabi Muhammad Shallallahu 'Alaihi Wassalam, sehingga penulis diberikan kekuatan, kesempatan, dan kelancaran dalam menyelesaikan skripsi yang berjudul "**Pengembangan RESTful API pada Sistem Remunerasi Dosen Universitas Lampung Bidang Pendidikan dan Penelitian dengan Autentikasi JSON Web Token (JWT)"**. Keberhasilan penyusunan skripsi ini tidak terlepas dari bimbingan, bantuan, dan dukungan dari berbagai pihak yang senantiasa menjadi sumber motivasi dan inspirasi bagi penulis. Setiap perhatian, doa, dan dorongan yang diberikan menjadi cahaya penuntun yang menguatkan langkah penulis hingga skripsi ini dapat diselesaikan dengan baik.

Pada kesempatan ini penulis mengucapkan terima kasih sebesar-besarnya kepada:

1. Kedua orang tua tercinta Bapak Supriyanto dan Ibu Khotimah. Bapak adalah cinta pertamaku yang selalu memelukku dengan kasih sayang dan keteguhan, sementara Ibu adalah pintu surgaku yang senantiasa menebarkan doa dan bimbingan tanpa lelah. Meski tak pernah menapaki bangku kuliah, pengorbanan, cinta, dan kesabarannya menjadi fondasi yang menguatkan setiap langkahku, menuntunku meraih ilmu dan impian dengan keyakinan dan harapan.
2. Adik tersayang Muhammad Faqih Al-Ghifari dan Almeera Azzahra Alfatunnisa. Terima kasih atas canda, tawa, dan semangat yang selalu kau bagi. Kehadiranmu membuat setiap langkahku lebih ringan dan perjalanan hidupku dipenuhi warna serta keceriaan yang tak tergantikan.

3. Bapak Dr. Eng. Heri Satria, S.Si., M.Si. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.
4. Bapak Dwi Sakethi, S.Si., M.Kom. selaku Ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.
5. Ibu Yunda Heningtyas, M.Kom. selaku Sekertaris Jurusan Ilmu Komputer FMIPA Universitas Lampung.
6. Bapak Tristiyanto, S.Kom., M.I.S., Ph.D. selaku Ketua Program Studi Ilmu Komputer FMIPA Universitas Lampung dan Dosen Pembahas. Kritik dan saran menjadi yang membangun menjadi pedoman bagi penulis dalam memperbaiki dan menyempurnakan penelitian ini.
7. Ibu Ossy Dwi Endah Wulansari, M.T. selaku Pembimbing Akademik. Saran dan dukungan senantiasa membimbing penulis dalam menjalani proses akademik.
8. Bapak M. Iqbal Parabi, S.SI., M.T. selaku Dosen Pembimbing Utama. Kritik dan saran yang membangun menjadi pedoman bagi penulis dalam menyusun penelitian ini, membantu menajamkan pemikiran serta memperkuat penelitian.
9. Bapak Igit Sabda Ilman, M.Kom. selaku Dosen Pembimbing Kedua. Kritik dan saran yang membangun membimbing penulis dalam menyempurnakan penelitian ini, dan menambah perspektif yang memperkuat penelitian.
10. Bapak dan Ibu Dosen Jurusan Ilmu Komputer FMIPA Universitas Lampung. Ilmu, pengetahuan, dan pengalaman yang telah di bagikan selama menempuh pendidikan di jurusan ini menjadi bekal berharga bagi pengembangan kemampuan akademik penulis.
11. Seluruh staf Jurusan Ilmu Komputer yang telah membantu dalam urusan administrasi selama penulis menempuh pendidikan sangat berperan dalam kelancaran proses akademik.
12. Kakak-kakak tersayang Bunga Govia Puti, Agita Viola Putri, Sausan Nabillah dan Ismiwati Intan Soraya atas dukungan, motivasi, serta pembelajaran yang telah diberikan menjadi sumber inspirasi dan memberikan tempat yang nyaman untuk berdiskusi, sehingga mempermudah penulis dalam menjalani perjalanan perkuliahan.

iv

13. Fathimah Abiyyi Khairunnisa, Amalia Nurul Rahmawati dan Chandra Prasetya Putra rekan penelitian penulis. Kerja sama dan kolaborasi yang terjalin selama penelitian ini menghasilkan diskusi dan ide-ide konstruktif yang membantu memperkuat setiap tahap hingga penelitian ini terselesaikan.
14. Teman-teman terkasih ''Temannya Cahya'' yaitu Vidya Sinta Bilkis, Fathimah Abiyyi Khairunnisa, Nabillah Aisyah, Siti Ayuni, Jihan Haya Mufialdo dan Cindy Loria. Kebersamaan selama perjalanan perkuliahan, dalam suka maupun duka, belajar dan berbagi pengalaman, telah menjadi bagian berharga yang menemani setiap langkah penulis hingga mencapai titik ini.
15. Teman-teman bayangan, Nailah Fauzah, Hendra Prasetya, Ajeng Fuji Rahayu dan Muhammad Jafar. Meski kebersamaan sebagian besar berlangsung secara daring melalui *Google Meeting* dan *Zoom* dalam pembelajaran Studi Independen, dukungan dan kolaborasi mereka menghadirkan semangat serta inspirasi penulis.
16. Teman-teman Ilmu Komputer Angkatan 2021 yang telah menjadi bagian dari perjalanan ini, terima kasih atas kebersamaan dan dukungan yang diberikan selama perjalanan kuliah di Jurusan Ilmu Komputer Universitas Lampung.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan dan belum sepenuhnya sempurna. Meski demikian, penulis berharap karya ini dapat memberikan manfaat serta menjadi konstribusi yang berarti bagi perkembangan ilmu pengetahuan, khususnya civitas akademika Jurusan Ilmu Komputer Universitas Lampung.

Bandar Lampung, 28 Agustus 2025

Shafa Auliya
NPM. 2117051042

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR KODE

I. PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi telah membawa perubahan besar dalam berbagai sektor, termasuk pendidikan tinggi. Di era digital saat ini, perguruan tinggi dituntut untuk menerapkan sistem informasi yang mampu meningkatkan efisiensi dan efektivitas dalam pengelolaan data akademik dan administratif (Alfarizi dkk., 2023). Sistem informasi yang terstruktur dan terintegrasi berperan penting dalam mendukung pengambilan keputusan berbasis data, serta dalam meningkatkan kualitas layanan pendidikan secara menyeluruh.

Salah satu penerapan teknologi informasi di lingkungan perguruan tinggi adalah sistem remunerasi dosen. Sistem ini digunakan untuk menilai kinerja dosen berdasarkan capaian Tri Dharma Perguruan Tinggi, yang meliputi pendidikan, penelitian, pengabdian kepada masyarakat, serta unsur penunjang (Sari dkk., 2024). Penilaian ini menjadi dasar pemberian insentif, sehingga keakuratan, konsistensi, dan keterpaduan data menjadi faktor penting dalam menjamin transparansi dalam proses tersebut.

Universitas Lampung telah menerapkan sistem remunerasi berbasis laporan Beban Kerja Dosen (BKD) yang disusun dan diverifikasi setiap semester sebagai dasar perhitungan insentif (Nugraha & Rosmeida, 2021). Namun, dalam praktiknya, pemanfaatan data kinerja dosen, khususnya pada bidang pendidikan dan penelitian, masih belum optimal. Proses pengumpulan data masih dilakukan secara manual dan tidak terintegrasi, sehingga menimbulkan sejumlah kendala, seperti keterlambatan, ketidaktepatan, serta risiko kesalahan pencatatan.

Ketiadaan integrasi antar sistem di lingkungan Universitas Lampung menyebabkan proses pengolahan data kinerja dosen menjadi tidak efisien. Hal ini tidak hanya menghambat penyusunan laporan internal, tetapi juga berdampak pada kelengkapan data untuk proses akreditasi program studi. Padahal, proses akreditasi membutuhkan data kinerja dosen yang lengkap, akurat, dan dapat diakses dengan cepat, khususnya pada bidang pendidikan dan penelitian yang menjadi indikator utama dalam penilaian.

Untuk mengatasi permasalahan tersebut, penelitian ini mengembangkan sebuah *Application Programming Interface* (API) yang berfungsi untuk mengintegrasikan data kinerja dosen dari sistem remunerasi. API ini dirancang menggunakan pendekatan RESTful untuk mendukung pertukaran data antar sistem secara efisien dan fleksibel (Choirudin & Adil, 2019). Dengan API ini, data kinerja dosen dalam bidang pendidikan dan penelitian dapat diakses oleh sistem lain secara otomatis, tepat waktu, dan terstruktur.

Selain efisiensi, aspek keamanan juga menjadi fokus utama dalam integrasi sistem. Informasi kinerja dosen merupakan data sensitif yang harus dijaga kerahasiaannya dan hanya boleh diakses oleh pihak yang berwenang. Untuk itu, dalam penelitian ini juga diterapkan mekanisme autentikasi menggunakan JSON *Web* Token (JWT). Berdasarkan penelitian sebelumnya, JWT mampu memberikan perlindungan yang baik terhadap data dengan memastikan bahwa hanya pengguna yang memiliki hak akses yang dapat mengakses data tertentu secara aman (Adam dkk., 2020a).

Berdasarkan permasalahan tersebut, penelitian ini difokuskan pada pengembangan RESTful API dengan penerapan autentikasi JSON *Web* Token (JWT) pada sistem remunerasi dosen Universitas Lampung. Pengembangan ini bertujuan untuk mengintegrasikan data kinerja dosen secara aman dan efisien, khususnya pada bidang pendidikan dan penelitian. Dengan demikian, diharapkan sistem ini dapat meningkatkan efektivitas pengelolaan data, mempercepat proses pelaporan, serta mendukung kelancaran akreditasi program studi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana mengintegrasikan data kinerja dosen pada Sistem Remunerasi Dosen Universitas Lampung dalam bidang pendidikan dan penelitian?
2. Bagaimana menerapkan mekanisme keamanan dalam pertukaran data pada Sistem Remunerasi Dosen Universitas Lampung?
3. Bagaimana mengembangkan RESTful API yang dapat menyediakan data kinerja dosen untuk mendukung proses akreditasi?

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penelitian difokuskan pada pengembangan RESTful API untuk menyajikan data kinerja dosen dalam bidang pendidikan dan penelitian pada Sistem Remunerasi Dosen Universitas Lampung.
2. Sistem yang dikembangkan hanya menyediakan fitur untuk mengakses dan menampilkan data kinerja dosen pada bidang pendidikan dan penelitian, tanpa mencakup fungsi pengelolaan atau modifikasi data secara langsung.
3. Data kinerja dosen yang digunakan dalam sistem ini masih bersifat terbatas.
4. Data yang digunakan dalam penelitian ini secara khusus berasal dari bagian beban lebih pada Sistem Remunerasi Dosen Universitas Lampung.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diidentifikasi, penelitian ini bertujuan untuk:

1. Mengembangkan RESTful API yang menyajikan data kinerja dosen pada bidang pendidikan dan penelitian dalam Sistem Remunerasi Dosen Universitas Lampung.
2. Merancang serta mengimplementasikan mekanisme autentikasi dan otorisasi antara *client-server* menggunakan JSON *Web* Token (JWT) untuk menjaga keamanan pertukaran data Sistem Remunerasi Universitas Lampung dengan sistem informasi lainnya.
3. Menyediakan data kinerja dosen yang terstruktur untuk mendukung kebutuhan proses akreditasi.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat, diantaranya:

1. Mengoptimalkan pemanfaatan data kinerja pada bidang pendidikan dan penelitian untuk mendukung proses akreditasi.
2. Meningkatkan keamanan akses data melalui penerapan autentikasi dan otorisasi berbasis JSON *Web* Token (JWT).
3. Meningkatkan efisiensi integrasi data antara Sistem Remunerasi Dosen Universitas Lampung dan sistem informasi lainnya.

II. TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai acuan dalam penelitian ini untuk memperkuat kerangka penyusunan dan memperluas pemahaman terhadap konsep yang relevan. Selain itu, penelitian terdahulu juga berfungsi sebagai referensi untuk mendalami literatur sesuai dengan penelitian yang dilakukan. Adapun referensi yang digunakan dalam penelitian ini disajikan pada Tabel 1.

Tabel 1. Penelitian Terdahulu

No	Judul Penelitian	Teknologi yang Digunakan	Hasil
1.	*RESTful Web Service Implementation on Unklab Information System Using JSON Web Token (JWT)* (Adam dkk., 2020b).	JavaScript, MySQL, dan JSON *Web* Token (JWT).	Penelitian ini berhasil mengimplementasikan layanan *web* RESTful pada Sistem Informasi Universitas Klabat dengan menggunakan JSON *Web* Token (JWT) untuk otorisasi akses data mahasiswa. Layanan *web* tersebut mampu meningkatkan aksebilitas data bagi pengguna yang telah terotoriasi, serta menjamin keamanan dan integritas data yang tersimpan dalam *database*. Hasil penelitian menunjukkan bahwa sistem yang dikembangkan mampu memenuhi kebutuhan pengguna dalam pengelolaan data mahasiswa secara efisien.

No	Judul Penelitian	Teknologi yang Digunakan	Hasil
2.	Implementasi API RESTful dengan JSON *Web* Token pada Aplikasi *E-commerce Thrifty Shop* Untuk Otentikasi dan Otorisasi Pengguna (Nashikhuddin dkk., 2023).	JSON *Web* Token (JWT) dan API RESTful.	Implementasi API RESTful dengan JSON *Web* Token (JWT) pada aplikasi *e-commerce Thrifty Shop* mampu meningkatkan keamanan dan performa sistem secara keseluruhan. JWT digunakan sebagai mekanisme autentikasi dan otorisasi yang efektif dalam melindungi data sensitif serta memastikan hak akses pengguna sesuai dengan perannya. Hasil pengujian menunjukkan waktu rata-rata proses autentikasi sebesar 101,792 milidetik, yang mencerminkan efisiensi dalam proses *login* pengguna. Meskipun demikian, pengelolaan kontrol akses yang baik tetap diperlukan untuk mengurangi risiko keamanan.
3.	*Implementation of One Data-based Lecturer Profile Information System for Key Performance Indicator Monitoring* (Septama dkk., 2022).	*Web Service*, Metode *Extreme Programming*, dan *Framework* PHP Laravel	Penelitian ini berhasil mengimplementasikan sistem informasi profil dosen berbasis satu data yang terintegrasi dengan aplikasi SISTER untuk mendukung pemantauan Indikator Kinerja Utama (IKU). Sistem ini dirancang agar dosen cukup sekali memasukkan data, sehingga dapat mengurangi redundansi dan meningkatkan efisiensi pengelolaan informasi. Hasil uji pengalaman pengguna (*User Experience Questionnaire*) menunjukkan bahwa impresi pengguna terhadap sistem sangat baik, yang menunjukkan bahwa sistem tersebut dapat digunakan secara efektif dalam mendukung pengelolaan data dosen.

2.2 Beban Kerja Dosen (BKD)

Beban Kerja Dosen (BKD) merupakan kewajiban yang harus dipenuhi dosen dalam melaksanakan tugas sebagai pendidik profesional dan ilmuwan dalam jangka waktu tertentu. Kewajiban ini mencakup pelaksanaan Tridharma Perguruan Tinggi, yang meliputi pendidikan, penelitian, dan pengabdian kepada masyarakat, serta dapat disertai tugas tambahan dan penunjang. Laporan kinerja BKD disampaikan setiap semester di perguruan tinggi tempat dosen bertugas, dengan ketentuan beban kerja minimal 12 SKS dan maksimal 16 SKS.

Ketentuan ini diatur dalam Pasal 72 Undang-Undang Guru dan Dosen, yang menjelaskan bahwa BKD mencakup perencanaan dan pelaksanaan pembelajaran, evaluasi, bimbingan, penelitian, tugas tambahan, serta pengabdian kepada masyarakat. Meskipun batas kepatutan beban kerja telah ditentukan, pada praktiknya banyak dosen melaksanakan tugas melebihi 16 SKS dalam satu semester. Dalam kondisi tersebut, pimpinan perguruan tinggi dapat memberikan penghargaan berupa insentif atau remunerasi sesuai kemampuan lembaga, sebagai bentuk apresiasi atas beban lebih yang dijalankan dosen (Direktur Jenderal Pendidikan Tinggi Kementerian Pendidikan dan Kebudayaan, 2021).

2.3 Remunerasi

Remunerasi berasal dari bahasa Inggris *remunerate*, yang berarti memberi imbalan atas jasa atau pekerjaan yang telah dilakukan (Putra dkk., 2019). Menurut Kamus Besar Bahasa Indonesia (KBBI), remunerasi diartikan sebagai penghargaan atau imbalan atas jasa seseorang (Khairina & Afrizai, 2024). Secara umum, remunerasi mencakup seluruh bentuk kompensasi yang diterima pegawai atas kontribusinya terhadap organisasi, baik berupa gaji pokok, tunjangan, maupun insentif lainnya, yang dapat diberikan secara langsung maupun tidak langsung (Darmawan dkk., 2021).

Remunerasi merupakan bentuk penghargaan atas pencapaian kinerja individu yang mendukung tercapaianya tujuan organisasi. Besarannya dipengaruhi oleh tanggung jawab pekerjaan, tingkat pengalaman, dan hasil kerja yang dicapai. Sistem remunerasi yang dirancang secara tepat, dengan mempertimbangkan beban kerja dan insentif yang sesuai, terbukti dapat meningkatkan motivasi, produktivitas, serta efisiensi kerja (Dzikria dkk., 2023).

Dalam sektor pemerintahan, termasuk Pegawai Negeri Sipil (PNS), remunerasi diberikan melalui kebijakan tunjangan kinerja. Berdasarkan Peraturan Menteri Perhubungan Republik Indonesia Nomor PM 107 Tahun 2013, tunjangan ini dihitung berdasarkan capaian kinerja individu dan kemajuan pelaksanaan reformasi birokrasi yang mendukung pencapaian target organisasi (Rachmawaty & Pandoyo, 2020).

Di perguruan tinggi seperti Universitas Lampung, sistem remunerasi dosen disusun berdasarkan kontribusi terhadap pelaksanaan tridharma perguruan tinggi, yang meliputi pendidikan, penelitian, pengabdian kepada masyarakat, dan kegiatan penunjang lainnya. Penilaian dilakukan secara objektif dengan memperhatikan beban kerja dan hasil kerja dosen, untuk memastikan keseimbangan antara tugas yang dilaksanakan dan kompensasi yang diterima, serta untuk mendorong peningkatan kualitas institusi secara berkelanjutan.

Di Unila, remunerasi diberikan dalam bentuk insentif kinerja (*pay for performance*) berdasarkan jumlah poin kinerja yang dicapai dalam periode tertentu. Poin ini berasal dari kontrak kinerja dan/atau perhitungan yang merujuk pada Beban Kerja Dosen (BKD) dan pedoman penilaian remunerasi. Penilaian mencakup dosen, pejabat pengelola, dan tenaga kependidikan, dengan mempertimbangkan kinerja individu serta hasil kerja unit kerja masing-masing, seperti program studi, fakultas, dan rektorat. Untuk dosen, poin kinerja dihitung dari pencapaian pribadi dan akumulasi kinerja program studi yang dinaunginya (Universitas Lampung, 2025).

2.4 Bidang

Penelitian ini berfokus pada kinerja dosen bidang pendidikan dan penelitian, khususnya dalam mendukung proses akreditasi ataupun pelaporan institusional lainnya. Pada bidang pendidikan, kinerja dosen mencakup pengelolaan pembelajaran yang efektif serta pencapaian hasil pendidikan yang optimal. Sementara itu, bidang penelitian, kinerja dosen berkaitan dengan kualitas dan kuantitas penelitian yang dihasilkan. Penjelasan masing-masing bidang sebagai berikut.

2.4.1 Bidang Pendidikan

Pendidikan dan pengajaran merupakan salah satu pilar utama dalam Tri Dharma Perguruan Tinggi, yang menegaskan tanggung jawab perguruan tinggi dalam menyelenggarakan proses pembelajaran untuk menghasilkan lulusan yang kompeten dan sesuai dengan kebutuhan masyarakat serta dunia industri (Regita, 2023). Pendidikan mencerminkan perpaduan antara pembentukan karakter, penguasaan ilmu pengetahuan, dan pengembangan keterampilan profesional, sehingga menjadi bagian yang tidak terpisahkan dari upaya mencerdaskan kehidupan bangsa.

Sebagai bentuk tanggung jawab tersebut, perguruan tinggi menyelenggarakan program pendidikan seperti sarjana, magister, dan doktor untuk memberikan layanan pendidikan tinggi yang berkualitas. Proses pembelajaran dirancang secara aplikatif dan inovatif untuk membekali mahasiswa dengan kompetensi akademik dan profesional. Selain itu, kurikulum terus dikembangkan agar tetap relevan dengan perkembangan ilmu pengetahuan serta kebutuhan masyarakat (Amalia dkk., 2024).

2.4.2 Bidang Penelitian

Penelitian merupakan pilar penting dalam Tri Dharma Perguruan Tinggi yang mewajibkan perguruan tinggi untuk menghasilkan karya ilmiah yang berkontribusi terhadap pengembangan ilmu pengetahuan, teknologi, dan seni (Regita, 2023). Kegiatan penelitian bertujuan menggali pengetahuan baru dan mengungkapkan kebenaran berdasarkan kaidah ilmiah.

Melalui penelitian, diharapkan lahir inovasi yang berdampak positif bagi kemajuan ilmu pengetahuan dan teknologi. Penelitian juga menjadi saranabagi dosen dan mahasiswa untuk aktif mencari solusi atas permasalahan nyata di masyarakat. Hasil penelitian tersebut kemudian diintegrasikan ke dalam proses pembelajaran, sehingga mahasiswa dapat memperoleh manfaat dari temuan-temuan terbaru dan memperluas wawasan akademiknya (Amalia dkk., 2024).

2.5 *Unified Modelling Language* (UML)

Unified Modeling Language (UML) merupakan bahasa visual berbasis diagram yang digunakan untuk menggambarkan, menentukan, dan mendokumentasikan pengembangan sistem berbasis objek (*object-oriented*) (Narulita dkk., 2024a). UML berfungsi sebagai bahasa pemodelan standar yang mendukung proses analisis dan perancangan sistem dengan menyederhanakan permasalahan kompleks agar lebih mudah dipahami dan dievaluasi (Widyastuti dkk., 2022).

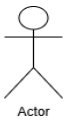
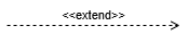
Melaui pemodelan UML, proses pengembangan sistem dapat mengikuti kerangka kerja terstruktur yang menyerupai *blueprint*, mencakup pemodelan proses bisnis, perancangan kelas yang dapat diimplementasikan dalam bahasa pemrograman, desain basis data, serta identifikasi komponen utama dalam sistem (Narulita dkk., 2024a). Oleh karena itu, UML memiliki peran penting dalam tahap analisis dan perancangan sistem informasi.

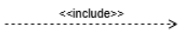
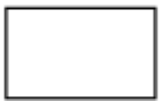
2.5.1 *Use Case Diagram*

Use Case Diagram merupakan salah satu jenis diagram UML yang digunakan untuk memvisualisasikan interaksi antara pengguna dan sistem dalam suatu konteks tertentu (Pranoto dkk., 2024). Diagram ini menggambarkan hubungan antara satu atau lebih aktor dengan sistem informasi yang dirancang, serta menunjukkan bagaimana aktor-aktor tersebut berinteraksi dengan sistem untuk mencapai tujuan tertentu (Hafsari dkk., 2023).

Use Case Diagram bekerja dengan merepresentasikan skenario penggunaan sistem melalui interaksi antara aktor, seperti admin atau pengguna akhir, dan sistem itu sendiri (Ramdany dkk., 2024). sistem tersebut digunakan (Ramdany dkk., 2024). Dengan demikian, diagram ini dapat membantu analis dalam mengidentifikasi kebutuhan fungsional sistem (*requirement*), menjelaskan rancangan sistem kepada pengguna, serta merancang fitur-fitur utama yang akan diimplementasikan (Narulita dkk., 2024b). Berikut merupakan simbol-simbol utama yang digunakan dalam *Use Case Diagram* (Dias & Muhallim, 2022).

Tabel 2. Simbol *Use Case Diagram*.

No	Simbol	Nama	Keterangan
1.		Aktor	Entitas di luar sistem yang berinteraksi langsung, seperti pengguna atau sistem eksternal.
2.		*Use Case*	Menyatakan fungsi atau layanan yang disediakan sistem untuk aktor demi mencapai tujuan tertentu.
3.		Asosiasi	Menunjukkan hubungan antara aktor dan *use case*, yaitu bentuk interaksi atau komunikasi di antaranya.
4.		*Extend*	Menyatakan perluasan dari suatu *use case*. *Use*

No	Simbol	Nama	Keterangan
			case tambahan ini bersifat opsional.
5.		*Include*	Menunjukkan bahwa suatu *use case* menyertakan *use case* lain sebagai bagian dari prosesnya. Wajib dijalankan.
6.		*System*	Menunjukkan batas sistem yang menjadi ruang lingkup seluruh interaksi dalam diagram.
7.		*Generalization*	Komunikasi antara satu fungsi umum dengan fungsi lain.

2.5.2 *Use Case Description*

Use Case Description merupakan dokumen yang menggambarkan secara rinci bagaimana aktor (baik pengguna maupun sistem lain) berinteraksi dengan sistem untuk mencapai tujuan tertentu. Dokumen ini memuat informasi seperti tujuan interaksi, langkah-langkah yang dilakukan, kondisi sebelum dan sesudah, hasil yang diharapkan, serta kemungkinan variasi atau kegagalan skenario. Tujuan utama dari deskripsi ini adalah untuk memberikan pemahaman menyeluruh mengenai fungsionalitas sistem.

Penyusunan *Use Case Description* membutuhkan komponen-komponen yang dapat menggambarkan secara rinci alur interaksi antara aktor dan sistem. Berikut merupakan komponen utama yang digunakan dalam penyusunannya.

Tabel 3. Komponen *Use Case Description*

No	Nama	Keterangan
1.	*Use Case Name*	Nama skenario atau aktivitas yang dijelaskan dalam use case, yang menggambarkan tujuan utama proses.
2.	ID	Kode unik untuk membedakan satu *use case* dengan yang lain, sehingga memudahkan pengelolaan dokumen.
3.	*Priority*	Tingkat kepentingan atau urgensi dari use case, biasanya ditentukan berdasarkan kebutuhan sistem atau bisnis.
4.	*Actor*	Pengguna atau sistem yang berinteraksi langsung dengan sistem dalam skenario tersebut.
5.	*Description*	Penjelasan singkat mengenai aktivitas yang dilakukan serta tujuan aktor dalam berinteraksi dengan sistem.
6.	*Trigger*	Peristiwa atau aksi yang memulai skenario *use case*.
7.	*Precondition*	Kondisi yang harus terpenuhi sebelum *use case* dapat dijalankan.
8.	*Normal Course*	Alur utama atau urutan langkah-langkah normal yang dilakukan oleh aktor dan sistem hingga tujuan tercapai.
9.	*Postcondition*	Keadaan sistem setelah skenario selesai dilaksanakan, menggambarkan hasil akhir yang diharapkan.
10.	*Sub Flows*	Langkah tambahan dalam alur normal, biasanya berupa proses opsional, pengulangan, atau detail lanjutan dari suatu langkah.
11.	*Alternate/Exceptional Flows*	Alur alternatif atau pengecualian yang terjadi bila terdapat kesalahan, kondisi khusus, atau penyimpangan dari alur normal.

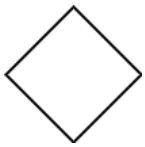
2.5.3 *Activity Diagram*

Activity diagram merupakan salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk memodelkan aliran aktivitas atau proses bisnis dalam suatu sistem (Pranoto dkk., 2024).

Diagram ini menyajikan urutan langkah-langkah yang menggambarkan bagaimana proses dimulai, berlangsung, bercabang, dan berakhir.

Activity Diagram mempermudah pemahaman terhadap logika alur kerja sistem, termasuk aktivitas yang terjadi secara berurutan maupun paralel (Narulita dkk., 2024b). Diagram ini juga menampilkan tindakan, keputusan, dan jalur alternatif dalam proses, sehingga sangat berguna dalam analisis dan perancangan sistem informasi. Berikut merupakan simbol-simbol utama yang digunakan dalam *Activity Diagram* (Alfarizi dkk., 2023).

Tabel 4. Simbol *Activity Diagram*.

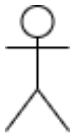
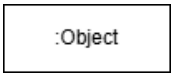
No	Simbol	Nama	Keterangan
1.		*Initial State*	Menandakan titik awal dimulainya alur aktivitas.
2.		*Actifity*	Menunjukkan rangkaian aktivitas atau proses dalam sistem untuk mencapai tujuan tertentu.
3.		*Transition*	Menyatakan perpindahan dari satu aktivitas ke aktivitas lain dalam alur proses.
4.		*Decision*	Menunjukkan titik pengambilan keputusan, di mana alur bercabang berdasarkan kondisi tertentu.
5.		*Fork/Join*	*Fork*: memisahkan satu alur menjadi beberapa alur paralel. *Join*: menyatukan kembali alur paralel menjadi satu jalur.
6.		*Final State*	Menunjukkan berakhirnya seluruh proses dalam satu jalur aktivitas.

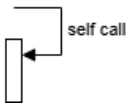
2.5.4 *Sequence Diagram*

Sequence diagram merupakan salah satu diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk memodelkan interaksi antar objek berdasarkan urutan waktu. Diagram ini menampilkan bagaimana objek-objek dalam sistem saling berkomunikasi melalui pertukaran pesan yang terjadi secara kronologis sebagai respons terhadap suatu peristiwa.

Diagram ini memiliki dua sumbu utama yaitu sumbu *vertikal* menggambarkan alur waktu dari atas ke bawah, sedangkan sumbu *horizontal* menunjukkan objek-objek yang terlibat dalam interaksi. Setiap objek direpresentasikan oleh *lifeline* yang menunjukkan durasi keberadaannya selama interaksi berlangsung. Diagram ini berfungsi untuk membantu memahami alur komunikasi sistem, dimulai dari pemicu suatu proses, pengiriman pesan antar objek, aktivitas yang berlangsung di masing-masing objek, hingga hasil akhir yang dihasilkan (Londjo, 2021). Berikut merupakan simbol-simbol utama yang digunakan dalam *Sequence Diagram*.

Tabel 5. Simbol *Sequence Diagram*

No	Simbol	Nama	Keterangan
1.		*Actor*	Entitas eksternal yang berinteraksi dengan sistem untuk memulai alur proses.
2.		*Lifeline*	Garis vertikal yang menunjukkan keberadaan dan waktu hidup objek selama proses berlangsung.
3.		*Object*	Objek di dalam sistem yang menjalankan fungsi tertentu atau menerima pesan dalam interaksi.

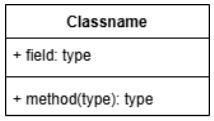
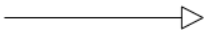
No	Simbol	Nama	Keterangan
4.		*Action Bar*	Menunjukkan periode saat objek/aktor sedang menjalankan suatu aktivitas.
5.		*Message*	Menggambarkan pengiriman pesan atau pemanggilan fungsi antar objek.
6.		*Replay Message*	Menunjukkan balasan atau hasil dari pesan yang telah dikirim sebelumnya..
7.		*Self Message*	Menunjukkan bahwa objek memanggil metode internal atau melanjutkan proses sendiri.

2.5.5 *Class Diagram*

Class Diagram merupakan jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan struktur statis sistem dengan memodelkan kelas-kelas, atribut, metode, serta relasi antar kelas dalam sistem. *Class* merupakan representasi dari sekumpulan objek yang memiliki struktur, perilaku, dan hubungan yang serupa (Malius & Dani, 2021). Berikut merupakan simbol-simbol yang digunakan dalam *Class Diagram* (Ramdany dkk., 2024).

Tabel 6. Simbol Class Diagram.

No	Simbol	Nama	Keterangan
1.		*Dependency*	Menunjukkan hubungan ketergantungan antara satu kelas dengan kelas lainnya. Artinya, suatu kelas menggunakan atau bergantung pada kelas lain untuk menjalankan fungsinya.

No	Simbol	Nama	Keterangan
2.		*Class*	Elemen dasar dalam pemrograman berorientasi objek. Digambarkan sebagai kotak yang terbagi menjadi tiga bagian yaitu nama kelas (atas), atribut (tengah), dan operasi/metode (bawah).
3.		*Association*	Hubungan struktural umum antara dua kelas.
4.		*Generalization*	Relasi pewarisan dari kelas umum ke kelas yang lebih khusus.
5.		*Directed Association*	Relasi yang menunjukkan satu kelas menggunakan kelas lain secara spesifik.
6.		*Aggregation*	Relasi seluruh-bagian, di mana bagian masih bisa berdiri sendiri.

Class Diagram terdiri dari tiga bagian utama dalam representasi kotak kelas, yaitu bagian atas, tengah, dan bawah. Berikut penjelasan masing-masing bagian: (Ramdany dkk., 2024).

a. Komponen Atas (Nama Kelas)

   Bagian atas menampilkan nama kelas, yang merepresentasikan entitas utama dalam sistem. Ini mewakili komponen tingkat tertinggi dalam sistem.

b. Komponen Tengah (Atribut)

   Bagian tengah berisi atribut, yaitu variabel atau properti yang dimiliki oleh objek dari kelas tersebut. Atribut menunjukkan data atau karakteristik dari suatu objek.

c. Komponen Bawah (Operasi/Metode)

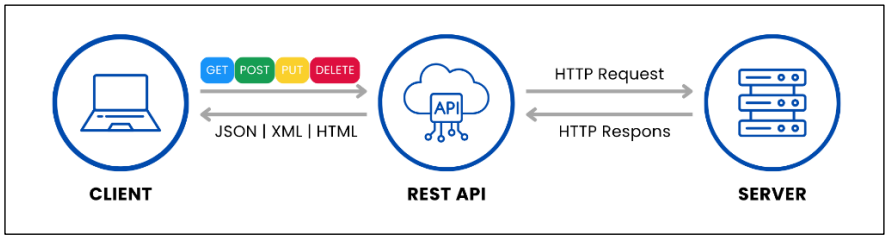
   Bagian bawah menampilkan operasi atau metode, yaitu fungsi atau prosedur yang dapat dilakukan oleh objek dari kelas tersebut. Metode menunjukkan perilaku dari objek dan bagaimana objek tersebut berinteraksi dengan objek lain.

2.6 *Web Service*

Web service merupakan perangkat lunak yang memungkinkan sistem berbeda saling berinteraksi melalui jaringan komputer (Pariddudin & Fadhli, 2022). Data dalam *web service* umumnya disimpan dan ditransmisikan dalam format XML, sehingga dapat diakses lintas *platform*, sistem operasi, maupun bahasa pemrograman. *Web service* dirancang untuk mendukung komunikasi antar mesin (*machine-to-machine*) melalui jaringan (Junardi dkk., 2020).

Tujuan utamanya adalah menyediakan informasi atau layanan yang dapat diakses oleh sistem lain melalui protokol standar, seperti HTTP, sehingga sistem eksternal dapat berinteraksi dengan layanan yang disediakan (Pariddudin & Fadhli, 2022). Kemampuan ini menjadikan *web service* sebagai solusi efektif untuk mengintegrasikan berbagai sistem yang terpisah dalam suatu jaringan.

2.7 *RESTful API*



Gambar 1. Model REST API

Representional State Transfer (REST) merupakan standar arsitektur komunikasi yang umum digunakan dalam pengembangan situs *web* dan layanan berbasis aplikasi (Rivando, 2023). REST memungkinkan pertukaran data antara klien dan server melalui protokol HTTP. Dengan pendekatan ini, klien dapat mengirimkan permintaan menggunakan *Uniform Resource Identifier* (URI), dan server merespons permintaan tersebut secara efisien (Perkasa & Setiawan, 2018). Mekanisme ini serupa dengan prinsip kerja aplikasi *web* pada umumnya, di mana

interaksi antara klien dan server terjadi secara langsung melalui protokol HTTP (Hasanuddin dkk., 2022c).

Sementara itu, *Application Programming Interface* (API) merupakan antarmuka yang memungkinkan suatu program berinteraksi dengan aplikasi atau layanan lain. Dalam hal ini, API digunakan untuk memanggil fungsi tertentu menggunakan protokol HTTP dan memberikan respons dalam format yang mudah diolah, seperti XML atau JSON (Hasanuddin dkk., 2022a). Cara kerja API melibatkan interaksi klien dan server melalui metode REST, menggunakan metode HTTP seperti GET, POST, PUT, dan DELETE (Cahyono dkk., 2022).

REST API terdiri dari beberapa komponen utama yang mendukung proses komunikasi antara klien dan server secara terstruktur, yaitu (Perdana, 2018):

1. URL Desain

   REST API diakses melalui HTTP, sehingga struktur dan penamaan URL harus dirancang secara konsisten dan mudah dipahami. URL atau *endpoint* menjadi titik masuk utama dalam pemanggilan API.

2. HTTP *Verbs*

   HTTP *verbs* atau metode HTTP digunakan klien untuk menentukan jenis operasi yang ingin dilakukan terhadap data. Umumnya digunakan:

   a. GET: Mengambil data.
   b. POST: Menambahkan data.
   c. PUT: Memperbarui data.
   d. DELET: Menghapus data.

3. Kode Respons HTTP

   Merupakan kode status yang dikembalikan server sebagai respons atas permintaan klien. Beberapa kategori utama meliputi:

   a. 2XX: Menunjukkan bahwa permintaan berhasil diproses (misalnya 200 OK).

b. 4XX: Menunjukkan adanya kesalahan pada sisi klien, seperti kesalahan dalam permintaan (misalnya 404 *Not Found*).
c. 5XX: Menunjukkan adanya kesalahan pada sisi server saat memproses permintaan (misalnya 500 Internal Server Error).

4. Format Respons

   Data hasil respons biasanya dikirim dalam format JSON atau XML. Format ini memungkinkan klien untuk mengurai (*parsing*) dan memproses data sesuai kebutuhan.

Selain komponen dasar, REST (*Representational State Transfer*) memiliki enam prinsip utama yang menjadi dasar dalam pengembangan layanan berbasis *web*. Prinsip-prinsip tersebut dijelaskan sebagai berikut (Farhandika dkk., 2024):

1. *Uniform Interface*

Prinsip ini menyederhanakan arsitektur dengan memisahkan tanggung jawab antara *client* dan server. REST menggunakan metode HTTP seperti GET, POST, PUT, dan DELETE untuk melakukan permintaan (*request*). URI (*Uniform Resource Identifier*) digunakan untuk mengidentifikasi sumber daya, sedangkan HTTP *response* berfungsi untuk mengembalikan data dari server kepada *client*. Prinsip ini merupakan fondasi utama dari desain RESTful.

2. *Client-Server*

Prinsip ini menekankan pemisahan antara pengembangan sisi *client* dan server, sehingga keduanya dapat dikembangkan secara independen. Dengan pemisahan ini, sistem menjadi lebih fleksibel, skalabel, dan mudah dikembangkan pada berbagai *platform*.

3. *Stateless*

Setiap permintaan dari *client* ke server harus memuat semua informasi yang dibutuhkan untuk memproses permintaan tersebut. Server tidak menyimpan

informasi status dari client sebelumnya. Oleh karena itu, semua state atau sesi (*session*) disimpan di sisi *client*.

4. *Layered System*

Prinsip ini mengharuskan arsitektur disusun dalam bentuk lapisan (layer) hierarkis. Setiap lapisan memiliki tanggung jawab tersendiri dan tidak mengetahui detail dari lapisan lainnya. Hal ini meningkatkan skalabilitas dan keamanan sistem.

5. *Code on Demand*

REST memungkinkan server untuk mengirimkan kode (seperti skrip JavaScript) ke *client* secara dinamis. *Client* dapat mengeksekusi kode tersebut untuk menambah atau menyesuaikan fungsionalitas sementara tanpa perlu pembaruan manual.

6. *Cacheable*

Response dari server harus secara eksplisit atau implisit menyertakan informasi apakah data dapat disimpan dalam *cache*. Jika dapat disimpan, *client* diperbolehkan menggunakan data tersebut untuk permintaan (*request*) yang sama di masa mendatang, sehingga dapat meningkatkan efisiensi dan mengurangi beban server.

Untuk menerapkan REST API secara tepat, pengembang perlu memahami sejumlah komponen yang umum digunakan dalam struktur dan komunikasi API. Berikut merupakan komponen Spesifikasi API (Farhandika dkk., 2024):

Tabel 7. Komponen Spesifikasi API

No	Nama	Deskripsi
1.	*Method*	Metode HTTP yang digunakan untuk mengakses, seperti GET, POST, PUT, dan DELETE.
2.	*Path*	Bagian dari URL yang menunjukkan lokasi *resource*, yang digunakan untuk mengakses data pengguna tertentu.

No	Nama	Deskripsi
3.	*Description*	Penjelasan mengenai fungsi dari endpoint API untuk membantu pengguna memahami tujuan penggunaannya.
4.	Parameter	Data yang dikirim dalam permintaan untuk mengatur hasil respons, seperti *path* parameter, *query*, *header*, atau *body*.
5.	*Request*	Permintaan dari klien kepada server yang terdiri atas *method*, *path*, *parameter*, dan *body* sesuai kebutuhan operasional.
6.	*Response*	Balasan dari server terhadap permintaan klien, yang berupa kode status dan data dalam format JSON atau XML.

2.8 JSON *Web* Token (JWT)

JSON *Web* Token (JWT) merupakan token dalam bentuk string yang digunakan untuk proses autentikasi dan pertukaran data antar sistem (Setiawan & Purnamasari, 2017). Token ini berisi informasi yang dapat diverifikasi dan dipercaya, tanpa memerlukan penyimpanan status sesi di sisi server. Ketika pengguna berhasil melakukan proses *login*, server akan menghasilkan token yang berfungsi sebagai bukti autentikasi. Token ini kemudian disimpan oleh pengguna, umumnya pada *local storage* atau *cookies*, dan dikirim kembali ke server setiap kali pengguna mengakses sumber daya tertentu (Edy dkk., 2017).

JWT menggunakan format JSON yang memudahkan server dalam memvalidasi identitas pengguna melalui token yang dikirimkan. Jika token valid, maka akses diberikan sesuai hak pengguna. JWT terdiri atas tiga bagian utama yang dipisahkan oleh tanda titik (“.”), yaitu *header*, *payload*, dan *signature* (Astowo & Sujarwo, 2023).

Salah satu keunggulan utama JWT adalah bersifat *stateless*, yang berarti server tidak perlu menyimpan data sesi pengguna (Tanaem dkk., 2016). Seluruh informasi autentikasi sudah tercakup di dalam token, sehingga sistem menjadi lebih ringan dan efisien, terutama dalam skala besar atau pada aplikasi berbasis RESTful API. JWT terdiri atas tiga komponen utama, yaitu (Okta, 2024):

1. ***Header***

   Header memuat informasi tipe token yaitu JWT dan algoritma yang digunakan untuk proses penandatanganan, seperti HMAC SHA256 atau RSA. Contoh struktur *header*:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Kode 1. Contoh *Header* JWT

Bagian ini kemudian dikodekan menggunakan algoritma Base64Url dan menjadi bagian pertama dari JWT.

2. ***Payload***

   Payload berisi klaim, yaitu pernyataan mengenai entitas (misalnya pengguna) dan data tambahan yang diperlukan. Terdapat tiga jenis klaim:

   a. Klaim Terdaftar

      Klaim ini merupakan klaim standar yang direkomendasikan untuk interoperabilitas, seperti:

      a) iss (*issuer*): pihak yang menerbitkan token
      b) exp (*expiration*): waktu kedaluwarsa token
      c) sub (*subject*): subjek token
      d) aud (*audience*): target pengguna token

   b. Klaim Publik

      Klaim publik, yaitu klaim yang dapat digunakan secara bebas, namun disarankan untuk didaftarkan di JSON *Web* Token *Registry* atau ditetapkan sebagai URI dengan *namespace* unik agar tidak terjadi konflik.

   c. Klaim Privat

      Klaim privat, yaitu klaim yang ditentukan secara khusus dan hanya digunakan oleh pihak-pihak tertentu sesuai kesepakatan bersama.

Contoh struktur *playload*:

```
{
  "nidn": "xxxxxxxxxx",
  "role": "admin",
  "iat": 1747820840,
  "exp": 1748425640
}
```

Kode 2. Contoh *Playload* JWT

Payload ini juga dikodekan menggunakan Base64Url untuk membentuk bagian kedua dari JSON *Web* Token.

3. *Signature*

Signature digunakan untuk memverifikasi bahwa token tidak mengalami perubahan dan berasal dari sumber yang sah. *Signature* dibuat dengan menggabungkan *header* dan *payload* yang telah dikodekan, lalu ditandatangani menggunakan *secret key* dan algoritma yang ditentukan pada *header*.

Sebagai contoh, jika menggunakan HMAC SHA256, maka *signature* dibentuk sebagai berikut:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
)
```

Kode 3. Contoh *Signature* JWT

Hasilnya adalah tiga string URL dalam format Base64 yang dipisahkan oleh titik. Berikut ini adalah contoh JWT, di mana *header* dan *payload* telah dikodekan.

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuaWRuIjoiMTExMTExMTExMSIsInJv
bGUiOiJhZG1pbiIsImlhdCI6MTc0NzgyMDg0MCwiZXhwIjoxNzQ4NDI1NjQwfQ.Ee6YMy
0snctEqGxjLb8NbM3NQQydsQX_Ek_G3CHVvV0
```

Kode 4. Contoh JWT

2.9 Node JS

Node.js merupakan *runtime environment* berbasis JavaScript yang berjalan di sisi server, bersifat *open source*, dan mendukung berbagai *platform* (Sauda & Barokah, 2022). Platform ini mengimplementasikan arsitektur *event-driven* dengan model *non-blocking* I/O, yang memungkinkan pemrosesan banyak permintaan secara bersamaan tanpa menghambat jalannya proses lain. Karakteristik tersebut menjadikan Node.js ringan, efisien, dan sesuai untuk pengembangan aplikasi *real-time* yang dapat dijalankan pada berbagai perangkat.

Berbeda dari JavaScript yang umumnya dijalankan pada sisi klien pada peramban, Node.js memungkinkan eksekusi kode JavaScript di sisi server dengan dukungan V8 *JavaScript Engine* yang dikembangkan oleh Google. Node.js juga menyediakan modul-modul bawaan seperti http, fs (*filesystem*), dan *crypto* (keamanan) yang dapat digunakan untuk membangun berbagai fungsi server (Aji dkk., 2022). Selain itu, Node.js memiliki pustaka server HTTP internal yang memungkinkan pengembangan server *web* secara langsung tanpa ketergantungan pada perangkat lunak tambahan seperti Apache atau Nginx (Hasanuddin dkk., 2022b).

Dukungan arsitektur dan komponen bawaan tersebut menjadi landasan berbagai keunggulan Node.js. Pemrosesan I/O asinkron memungkinkan aplikasi tetap responsif dan berkinerja tinggi meskipun melayani banyak koneksi secara simultan. Mekanisme *event loop* mendukung skalabilitas sistem, sehingga mampu menangani ribuan koneksi tanpa menambah beban signifikan pada sumber daya. Selain itu, keberadaan *Node Package Manager* (NPM) sebagai sistem manajemen paket resmi menyediakan ribuan modul siap pakai yang memudahkan pengembang dalam

menambahkan fungsionalitas baru, mengelola pustaka JavaScript, serta menangani dependensi proyek secara efisien. Ketersediaan modul dan pustaka yang luas ini mempercepat pengembangan aplikasi secara modular dan menambah fungsionalitas sistem tanpa harus membangun seluruh komponen dari awal (Chang dkk., 2019).

2.10 *Database*

Database merupakan kumpulan data yang berasal dari istilah "basis" yang berarti tempat penyimpanan, serta "data" yang merujuk pada catatan informasi dunia nyata yang merepresentasikan berbagai objek seperti manusia, barang, binatang, konsep, insiden, serta dapat diwujudkan dalam alfabet, angka, simbol, gambar, teks, suara, dan kombinasi lainnya (Prio dkk., 2022). *Database* mengelola data dengan aturan tertentu dan menghubungkannya secara terstruktur, sehingga memudahkan pengelolaan informasi. Dengan kontrol yang baik, *database* mampu menyimpan, mengorganisasikan, dan menghapus data secara efisien (Rezeki & Nasution, 2023). *Database* mengintegrasikan banyak catatan yang sebelumnya tersimpan dalam file terpisah, menciptakan kumpulan data yang saling terhubung dan memenuhi kebutuhan informasi suatu organisasi. Oleh karena itu, database berfungsi sebagai tempat penyimpanan yang luas dan dapat diakses oleh banyak pengguna untuk mendukung berbagai aktivitas pengolahan data secara optimal (Sudarso, 2022).

Pada penelitian ini *database* yang digunakan adalah PostgreSQL. PostgreSQL merupakan salah satu sistem manajemen basis data relasional objek (ORDBMS) yang bersifat *open source*, sehingga kode sumbernya dapat diakses dan digunakan secara bebas (Praba & Safitri, 2020). Sistem manajemen *database* ini mampu mengelola data dalam tabel yang saling terhubung dan dapat digunakan tanpa biaya serta bebas untuk dimodifikasi. Fungsi utama PostgreSQL adalah sebagai tempat untuk menyimpan dan mengelola data melalui perintah atau *query* SQL (Stiawan dkk., 2022). Dengan kemampuannya untuk mengolah data yang lebih kompleks, PostgreSQL juga dilengkapi dengan fitur-fitur yang sangat lengkap (Nurhayati &

Nasution, 2023). Ada tiga *tools* utama yang tersedia pada PostgreSQL sebagai berikut:

a PSQL

PSQL merupakan tampilan pada *command-line* yang digunakan untuk menuliskan perintah (*query*) pada PostgreSQL.

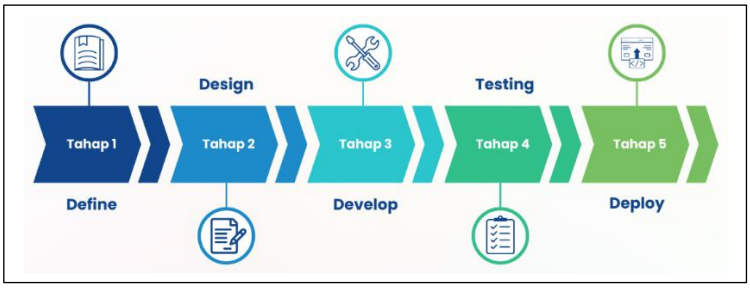
b PgAdmin

PgAdmin merupakan aplikasi berbasis grafis yang digunakan untuk mengelola PostgreSQL. Aplikasi desktop ini dapat terhubung ke berbagai server PostgreSQL, meskipun dijalankan pada sistem operasi yang berbeda.

c PHPPgAdmin

PHPPgAdmin merupakan alat berbasis *web* untuk mengelola PostgreSQL, serupa dengan PHPMyAdmin yang digunakan untuk MySQL.

2.11 Metode *API-First Development*



Gambar 2. Metode API-*First Development* (Postman, 2025)

Pendekatan API-*First* merupakan metode pengembangan perangkat lunak yang memprioritaskan perancangan dan pendefinisian API sebelum proses penulisan kode aplikasi lainnya. Dalam pendekatan ini, API diperlakukan sebagai kontrak terstandarisasi antara tim pengembang dan pengguna akhir, sehingga dapat meminimalkan potensi kesalahpahaman dan memastikan pemahaman yang jelas mengenai kemampuan serta batasan API. Pendekatan ini mendorong kolaborasi yang lebih baik antara tim *frontend* dan *backend*, sekaligus memfasilitasi

pengembangan sistem yang lebih terintegrasi, aman, dan dapat diskalakan (Chandrachood, 2021).

API berperan sebagai pondasi utama yang memungkinkan aplikasi mengakses layanan internal maupun eksternal secara efisien. Peran ini semakin signifikan di era teknologi saat ini, ketika pengguna mengharapkan ketersediaan data dan layanan yang dapat diakses secara instan melalui berbagai *platform* (Postman, 2025). Dengan demikian, API-First tidak hanya mendukung efisiensi pengembangan, tetapi juga memberikan fleksibilitas tinggi dalam integrasi lintas sistem.

Berikut adalah tahapan dalam pendekatan API-first.

1. ***Define***

   Pada tahap awal, pengembang mengidentifikasi kebutuhan API melalui analisis terhadap kebutuhan pengguna atau pemangku kepentingan. Proses ini mencakup penentuan fungsi utama API, skema data yang dihasilkan, serta persyaratan operasional, bisnis, dan keamanan. Tujuannya adalah memastikan pengembangan API selaras dengan kebutuhan dan spesifikasi teknis yang telah ditetapkan.

2. ***Design***

   Menentukan bagaimana API akan menyajikan data kepada pengguna dengan mengikuti standar spesifikasi, seperti OpenAPI. Desain mencakup perancangan endpoint, skema data, metode permintaan, serta bentuk respons. Dokumentasi desain ini menjadi panduan utama bagi seluruh proses pengembangan.

3. ***Develop***

   Setelah desain disetujui, pengembang mengimplementasikan fungsionalitas sesuai spesifikasi yang telah direncanakan, kemudian mulai menyusun dokumentasi teknis untuk memudahkan pemanfaatan API.

4. ***Testing***

   Melakukan pengujian untuk memastikan bahwa fungsionalitas API berjalan sesuai yang diharapkan. Pengujian dilakukan secara manual maupun otomatis untuk memverifikasi performa dan keamanan, sehingga API dapat memenuhi standar yang telah ditetapkan.

5. ***Deploy***

   Setelah pengujian selesai, API *dideploy* ke berbagai lingkungan, mulai dari pengembangan hingga produksi. Proses ini bertujuan untuk memastikan bahwa API yang telah diuji dapat diimplementasikan dengan benar sebelum digunakan oleh pengguna.

Pendekatan API-*First* memastikan pengembangan API dilakukan secara terstruktur, konsisten, dan aman. Strategi ini tidak hanya meningkatkan efisiensi pengembangan aplikasi, tetapi juga memberikan fleksibilitas tinggi dalam integrasi antar sistem.

2.12 Pengujian

Pengujian perangkat lunak merupakan tahap penting dalam pengembangan sistem yang bertujuan untuk memastikan bahwa aplikasi berjalan sesuai kebutuhan dan spesifikasi yang ditetapkan. Melalui pengujian, kesalahan (*bug*) dapat ditemukan lebih awal, serta keandalan, kinerja, dan keamanan sistem dapat dievaluasi sebelum digunakan secara luas (Abdillah dkk., 2025).

2.11.1 *White Box Testing*

Pengujian *white box* merupakan metode pengujian perangkat lunak yang berfokus pada desain dan kode program untuk memastikan bahwa fungsi

masukan dan keluaran yang dihasilkan sesuai dengan spesifikasi yang dibutuhkan (Khamaeni, 2023). Untuk melakukan pengujian ini secara efektif, penguji perlu memahami secara mendalam kode sumber yang akan diuji, sehingga kesalahan implementasi dapat terdeteksi dengan lebih baik. Pengujian *white box* dapat diterapkan pada berbagai tingkatan, seperti integrasi, unit, dan sistem, untuk memastikan kualitas dan ketepatan aplikasi yang diuji (Londjo, 2021).

2.11.2 JMeter

Apache JMeter adalah alat pengujian kinerja (*performance testing*) *open source* yang dibuat dengan bahasa Java (Ismail dkk., 2023). Alat ini digunakan untuk mengukur dan menganalisis performa aplikasi, khususnya aplikasi berbasis *web*. JMeter bekerja dengan mensimulasikan ribuan pengguna virtual yang mengakses aplikasi secara bersamaan. Simulasi ini membantu mengukur waktu respons, stabilitas, dan kemampuan sistem dalam menangani beban tinggi (Indrianto, 2023).

Pengujian dilakukan dengan membuat *test plan* yang mereplikasi aktivitas pengguna nyata, seperti *login* atau penggunaan fitur aplikasi. *Test plan* ini mengatur jumlah pengguna virtual (*thread*) dan skenario interaksinya. Pengujian bisa dilakukan secara manual atau otomatis menggunakan skrip. Penggunaan JMeter secara otomatis membantu meningkatkan efisiensi, akurasi, dan membantu mengidentifikasi kelemahan sistem untuk diperbaiki (Ismail dkk., 2023).

Untuk mendukung proses pengujian, JMeter menyediakan beberapa pengaturan teknis penting. Pengaturan ini digunakan untuk mengatur beban, skenario, dan pengukuran hasil pengujian sistem. Berikut adalah beberapa parameter utama yang sering digunakan dalam pengujian performa dengan JMeter (Ismail dkk., 2023):

Tabel 8. Parameter JMeter

No	Parameter	Deskripsi
1.	*Number of Threads* (*users*)	Jumlah pengguna virtual yang disimulasikan untuk mengakses aplikasi secara bersamaan. Semakin banyak *thread*, semakin besar beban yang diberikan kepada sistem.
2.	*Ramp-up Period* (*seconds*)	Waktu (dalam detik) yang dibutuhkan untuk menaikkan seluruh *thread* secara bertahap. Pengaturan ini mencegah beban datang secara tiba-tiba di awal pengujian.
3.	*Thread Spawn Rate* (*seconds*)	Kecepatan atau jeda kemunculan antar-*thread* selama *periode ramp-up*. Parameter ini memungkinkan kontrol terhadap laju pertambahan pengguna virtual.
4.	*Loop Count*	Jumlah pengulangan setiap *thread* dalam menjalankan skenario uji. Nilai ini menentukan durasi dan intensitas pengujian.
5.	*Min* (ms)	Waktu respons tercepat dalam satuan milidetik selama pengujian. Nilai ini mencerminkan seberapa cepat sistem dapat merespons permintaan dalam kondisi terbaik.
6.	*Max* (ms)	Waktu respons terlama dalam milidetik yang tercatat selama pengujian. Menunjukkan waktu terburuk sistem dalam merespons permintaan.
7.	*Average* (ms)	Rata-rata waktu respons dari seluruh permintaan. Nilai ini menggambarkan performa umum sistem selama pengujian berlangsung.
8.	*Error* %	Persentase permintaan yang gagal selama pengujian, seperti *timeout* atau *respons* tidak valid. Semakin rendah nilainya, semakin andal sistem yang diuji.
9.	*Throughput*	Jumlah permintaan yang berhasil diproses oleh sistem per satuan waktu (detik). Mencerminkan kapasitas sistem dalam menangani beban tinggi secara simultan.

2.11.3 Postman

Postman merupakan platform yang menyediakan serangkaian alat komprehensif untuk mempercepat *lifecycle* API. Postman mencakup tahap desain, pengujian, dokumentasi, simulasi, hingga kolaborasi dan kemudahan penemuan API (Postman, 2025). Selain itu, Postman juga digunakan untuk mengotomatisasi pengujian API, memantau performa API secara *real-time*, mengelola *environment* variabel untuk berbagai skenario pengujian, serta mendukung integrasi dengan berbagai layanan.

2.11.4 OWASP ZAP (*Zed Attack Proxy*)

OWASP ZAP (*Zed Attack Proxy*) merupakan perangkat lunak sumber terbuka (*open-source*) yang berfungsi untuk menguji keamanan aplikasi *web* secara otomatis maupun manual. Perangkat ini dikembangkan oleh komunitas global di bawah naungan *Open Web Application Security Project* (OWASP) dengan tujuan membantu pengembang, penguji, dan peneliti keamanan dalam mengidentifikasi serta menganalisis kerentanan pada aplikasi *web* sebelum dimanfaatkan oleh pihak yang tidak bertanggung jawab (Putra dkk., 2024).

ZAP memiliki arsitektur modular yang mengintegrasikan fitur pemindaian otomatis (*automated scanning*) dan penjelajahan aplikasi *web* (*crawling*) untuk memetakan struktur serta alur kerja aplikasi. Selain itu, ZAP berfungsi sebagai *proxy intercepting* yang memungkinkan pengguna memantau, merekam, dan memodifikasi lalu lintas data HTTP/HTTPS secara *real-time* sehingga dapat menguji respons aplikasi terhadap berbagai *input* dan manipulasi data (Mu'min dkk., 2024) .

Dengan kemampuan tersebut, ZAP mampu mendeteksi berbagai jenis kerentanan keamanan, seperti injeksi SQL (SQL *Injection), Cross-Site*

Scripting (XSS), dan kesalahan konfigurasi, yang sangat penting dalam menjaga keamanan aplikasi *web* secara menyeluruh (Putra dkk., 2024).

Dalam OWASP ZAP, warna *alert* digunakan untuk mengkategorikan tingkat risiko dari kerentanan yang ditemukan pada aplikasi *web*. Klasifikasi ini digunakan untuk memprioritaskan penanganan. Berikut adalah penjelasan masing-masing warna *alert* beserta tingkat risikonya (Ariyadi dkk., 2025).

Tabel 9. Tingkat Risiko Berdasarkan Warna *Alert*

No	Warna Alert	Tingkat Risiko	Deskripsi
1.	Merah	*High Risk*	Menandakan kerentanan yang sangat serius yang berpotensi mengekspos data sensitif atau menyebabkan kerusakan besar pada sistem. Kerentanan ini harus segera diperbaiki untuk mencegah dampak yang merugikan.
2.	Oranye	*Medium Risk*	Menunjukkan masalah keamanan yang cukup serius dan berpotensi menyebabkan akses tidak sah atau kebocoran data.
3.	Kuning	*Low Risk*	Mengindikasikan risiko yang relatif rendah, biasanya terkait dengan konfigurasi sistem yang kurang optimal atau potensi risiko tidak langsung membahayakan sistem secara signifikan.
4.	Biru	*Informational*/Info	Memberikan informasi tambahan atau peringatan yang tidak langsung berisiko tinggi, tetapi berguna sebagai panduan untuk analisis lebih lanjut dalam proses pengujian keamanan.

Dalam proses pengujian keamanan menggunakan OWASP ZAP, berbagai jenis serangan disimulasikan untuk mengidentifikasi potensi kerentanan pada sistem. Setiap jenis serangan tersebut diwakili oleh plugin khusus yang bertugas menguji

celah keamanan terkait. Tabel 10 berikut menyajikan ringkasan jenis serangan yang diuji.

Tabel 10. Jenis-Jenis Serangan

No	Jenis Serangan	Deskripsi
1.	*Path Traversal*	Upaya mengakses file atau direktori di luar batas yang diizinkan pada server, mengancam data sensitif.
2.	*Remote File Inclusion*	Memasukkan file berbahaya dari sumber eksternal untuk dijalankan di server, berpotensi mengendalikan server.
3.	*Cross-Site Scripting (XSS)*	Penyisipan skrip berbahaya ke dalam halaman *web* yang dapat dijalankan oleh pengguna lain.
4.	*SQL Injection*	Menyisipkan perintah SQL berbahaya ke dalam input untuk memanipulasi *database* secara tidak sah.
5.	*Server Side Code Injection*	Menyisipkan kode berbahaya yang dijalankan di sisi server, mengancam integritas sistem.
6.	*Remote OS Command Injection*	Menjalankan perintah sistem operasi secara tidak sah melalui input yang dimanipulasi.
7.	*XML External Entity (XXE) Attack*	Memanfaatkan kelemahan pemrosesan XML untuk membaca file lokal atau menyerang server.
8.	*Directory Browsing*	Pengungkapan daftar isi direktori server yang dapat mengungkap data atau struktur file penting.
9.	*Buffer Overflow*	Memasukkan data yang melebihi kapasitas buffer untuk menyebabkan crash atau eksekusi kode berbahaya.
10.	*Parameter Tampering*	Manipulasi parameter URL atau form untuk mengubah logika aplikasi atau mendapatkan akses tidak sah.
11.	*User Agent Fuzzer*	Mengirim variasi header User-Agent untuk mendeteksi kebocoran informasi atau perilaku tidak normal.

III. METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini dimulai pada semester ganjil tahun ajaran akademik 2024/2025 dan dilaksanakan di Laboratorium Rekayasa Perangkat Lunak, Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung. Rincian waktu pelaksanaan penelitian dapat dilihat pada Tabel 11 berikut.

Tabel 11. Waktu Penelitian.

Aktivitas	2024	2025				
	Desember	Januari	Februari	Maret	April	Mei
Plan						
Define						
Design						
Develop						
Testing						
Deploy						

3.2 Perangkat Penelitian

Perangkat keras dan lunak yang digunakan dalam penelitian ini memiliki spesifikasi yang mendukung proses pengembangan dan penerapan sistem. Spesifikasi tersebut dijelaskan sebagai berikut.

3.2.1 Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini berupa sebuah laptop dengan spesifikasi sebagai berikut.

a. *System Manufacturer* : ASUS ek COMPUTER INC
b. *System Model* : Vivobook ASUS X1404VA A1404VA
c. *Processor* : 13th Gen Intel® Core™ i5-1335U
d. GPU : Intel(R) Iris(R) Xe Graphics
e. RAM : 16.00 GB
f. *System Type* : 64 bit
g. *Storage* : SSD 512 GB

3.2.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini dapat dilihat sebagai berikut.

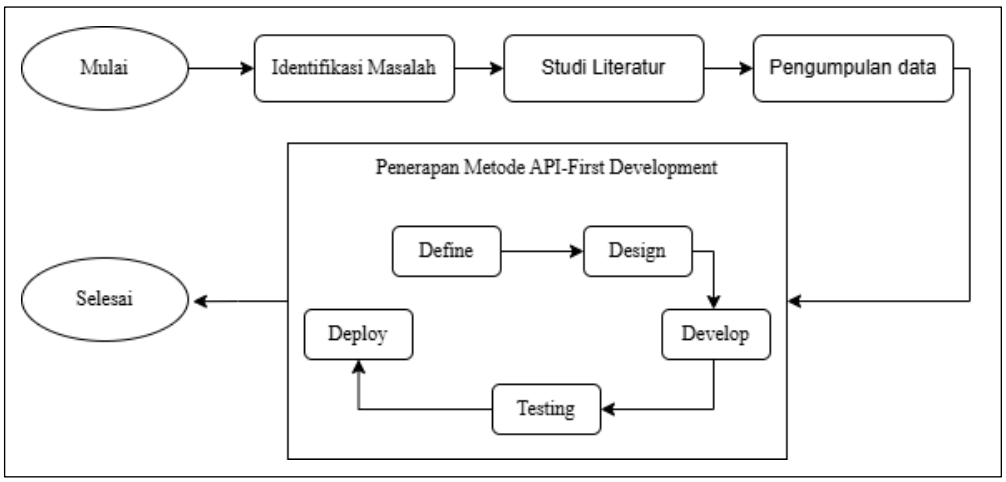
Tabel 12. Perangkat Lunak yang digunakan

No	Nama Perangkat Lunak	Deskripsi
1.	*Windows 11 Home Operating System*	Sistem operasi yang digunakan sebagai *platform* utama untuk menjalankan berbagai perangkat lunak dalam proses pengembangan.
2.	*Visual Studio Code*	*Visual Studio Code* merupakan editor kode ringan namun kaya fitur yang digunakan untuk menulis, mengedit, dan melakukan *debugging* kode program.
3.	*PostgreSQL versi 17*	*PostgreSQL* merupakan sistem manajemen basis data relasional *open source* yang andal dan digunakan untuk menyimpan serta mengelola data.

No	Nama Perangkat Lunak	Deskripsi
4.	*Google Chrome*	*Google Chrome* merupakan peramban *web* yang digunakan untuk menguji aplikasi berbasis *web*.
5.	*Postman Agent*	*Postman* merupakan alat untuk menguji dan mengelola *Application Programming Interface* (API), termasuk membuat, mengirim, dan memantau permintaan API.
6.	*Apache JMeter*	*Apache JMeter* merupakan perangkat lunak destop berbasis Java yang digunakan untuk menguji fungsional dan kinerja aplikasi berbasis klien/server.
7.	*Navicat Premium Lite 17*	*Navicat* merupakan alat manajemen basis data yang mempermudah pengelolaan basis data seperti *PostgreSQL*.
8.	*Github*	*Github* merupakan *platform* pengelolaan versi dan kolaborasi proyek berbasis Git. *Platform* ini digunakan untuk menyimpan kode, melacak perubahan, dan bekerja sama tim.
9.	*Publish or Perish 8*	*Publish or Perish* merupakan alat pencarian referensi akademik yang digunakan untuk menganalisis data sitasi dan membantu menemukan sumber yang relevan.
10.	*Mendeley Reference Manager*	*Mendeley Reference Manager* merupakan alat pengelola referensi yang digunakan untuk menyimpan, mengatur, dan menyisipkan sitasi secara otomatis dalam dokumen.
11.	*Microsoft Word*	*Microsoft Word* merupakan perangkat lunak pengolah kata yang digunakan untuk menyusun dan membuat dokumen skripsi.
12.	*Draw.io*	*Draw.io* adalah aplikasi diagram berbasis *web* yang digunakan untuk membuat dan mengedit berbagai jenis diagram, seperti diagram alur, organisasi, dan UML.

3.3 Tahapan Penelitian

Proses penelitian ini terdiri dari beberapa tahapan yang digambarkan secara rinci pada Gambar 3 dalam bentuk diagram alir. Secara keseluruhan, terdapat empat tahap utama dalam penelitian ini, yaitu identifikasi masalah, studi literatur, pengumpulan data, dan penerapan metode API-*First Development*. Setiap tahapan dilakukan secara sistematis dan berurutan untuk memastikan kelancaran proses serta konsistensi pelaksanaan penelitian dari awal hingga akhir.



Gambar 3. Diagram Alir Tahapan Penelitian

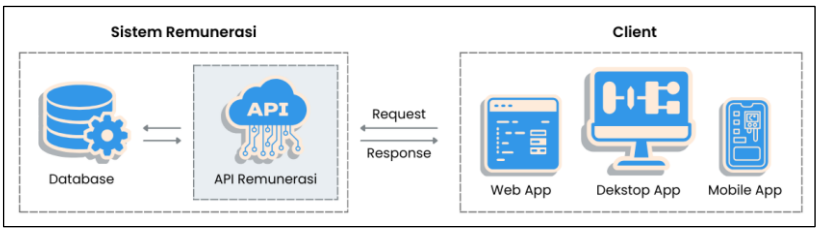
3.3.1 Identifikasi Masalah

Salah satu permasalahan utama dalam sistem remunerasi dosen di Universitas Lampung adalah keterbatasan akses terhadap data kinerja dosen bidang pendidikan dan penelitian. Saat ini, data tersebut hanya dapat diakses oleh dosen yang bersangkutan, sehingga dekan dan ketua program studi tidak memiliki akses langsung terhadap informasi yang dibutuhkan. Padahal, data kinerja memegang peranan penting dalam proses evaluasi dan akreditasi program studi, khususnya dalam menilai kontribusi dosen dalam bidang pendidikan dan penelitian. Ketidaktersediaan informasi yang cepat

dan akurat dapat menghambat proses evaluasi serta pengambilan keputusan yang dibutuhkan untuk menjaga dan meningkatkan mutu program studi.

Keterbatasan akses ini juga berdampak pada efektivitas sistem remunerasi yang seharusnya mampu memberikan penghargaan secara objektif berdasarkan capaian kinerja. Selain itu, dekan dan ketua program studi mengalami kesulitan dalam melakukan pemantauan serta perencanaan akademik berbasis data valid. Dalam konteks akreditasi, ketersediaan data kinerja dosen yang mudah diakses menjadi sangat penting untuk memenuhi standar mutu yang ditetapkan oleh lembaga akreditasi. Tanpa dukungan sistem yang memadai, proses evaluasi kinerja dosen menjadi kurang optimal, sehingga dapat memengaruhi kualitas kebijakan akademik dan mutu pendidikan secara keseluruhan.

Oleh karena itu, untuk mengatasi permasalahan tersebut, diperlukan solusi yang mampu menyediakan akses informasi secara lebih luas, tepat waktu, dan berbasis data terintegrasi. Salah satu upaya yang dapat dilakukan adalah melalui pengembangan sistem integrasi data kinerja dosen berbasis *Application Programming Interface* (API), yang bertujuan untuk mendukung proses akreditasi program studi secara efektif dan efisien. Rancangan proses integrasi API remunerasi ini ditampilkan pada Gambar 4.



Gambar 4. Arsitektur Integrasi API Remunerasi

Proses pengembangan API pada sistem remunerasi dosen di Universitas Lampung diawali dengan pelaksanaan kegiatan tridharma perguruan tinggi oleh dosen, yang mencakup bidang pendidikan, penelitian, pengabdian,

serta penunjang. Seluruh aktivitas tersebut dicatat dan diinput oleh dosen ke dalam Sistem Informasi Sumber Daya Terintegrasi (SISTER). Setelah data aktivitas terdokumentasi secara lengkap, dosen mengunduh portofolio Beban Kerja Dosen (BKD) dari sistem SISTER, yang berisi rekapitulasi kegiatan yang telah dilaporkan. Laporan BKD tersebut kemudian diunggah ke dalam sistem remunerasi untuk diproses menjadi poin capaian kinerja berdasarkan ketentuan yang berlaku. Data kinerja dosen yang telah diproses selanjutnya disimpan dalam basis data sistem remunerasi dan digunakan sebagai dasar perhitungan insentif serta bahan evaluasi kinerja dosen.

Untuk memperluas pemanfaatan data kinerja secara lebih luas dan menyeluruh, sistem remunerasi menyediakan API yang memungkinkan integrasi dengan sistem lain yang membutuhkan akses terhadap data kinerja dosen. Melalui API ini, sistem ekternal, seperti sistem evaluasi kinerja dan akreditasi program studi, dapat mengakses data secara langsung dan *real-time*. Integrasi tersebut tidak hanya meningkatkan efisiensi dalam pengambilan keputusan berbasis data, tetapi juga mendorong transparansi dalam pengelolaan informasi kinerja dosen di lingkungan Universitas Lampung.

3.3.2 Studi Literatur

Studi literatur memiliki peranan penting dalam pengembangan sistem karena memberikan landasan teoretis yang kuat dan relevan. Melalui kajian terhadap berbagai sumber referensi, baik berupa jurnal ilmiah, maupun dokumentasi teknis, pengembangan sistem dapat memperoleh pemahaman mendalam mengenai konsep, teknologi, serta metode yang mendukung keberhasilan implementasi. Selain itu, studi literatur membantu mengidentifikasi solusi yang telah terbukti efektif, sehingga proses pengembangan dapat diarahkan sesuai dengan kebutuhan dan standar yang berlaku.

Penelitian ini, memfokuskan kajian literatur pada pemahaman arsitektur *Representational State Transfer* (RESTful API), prinsip kerja Node.js, dan mekanisme autentikasi menggunakan JSON *Web* Token (JWT). Kajian ini juga mencakup integrasi sistem berbasis API dengan basis data. Pemahaman terhadap aspek-aspek tersebut diharapkan dapat mendukung terciptanya sistem yang aman, efisien, dan terintegrasi.

3.3.3 Pengumpulan Data

Dalam penelitian ini, proses pengumpulan data dilakukan untuk memperoleh dua jenis data, yaitu data primer dan data sekunder, yang digunakan sebagai dasar dalam analisis kebutuhan sistem.

a. Data Primer

Data primer dalam penelitian ini diperoleh melalui wawancara dengan pihak-pihak yang secara langsung menggunakan sistem remunerasi dosen Universitas Lampung. Teknik pengumpulan data dilakukan melalui dua pendekatan, yaitu wawancara langsung dengan Ketua Program Studi S1 Ilmu Komputer dan wawancara tidak langsung melalui kuesioner daring oleh Ketua Program Studi D3 Manajemen Informatika. Pengumpulan data ini bertujuan untuk memperoleh gambaran mengenai penggunaan sistem serta kendala yang dihadapi dalam mengakses data kinerja dosen di tingkat program studi.

Berdasarkan hasil wawancara, diketahui bahwa Ketua Program Studi tidak memiliki akses langsung terhadap data kinerja dosen melalui sistem. Akses hanya diberikan kepada dosen yang bersangkutan, sementara pihak program studi harus mengajukan permohonan resmi kepada pengelola sistem untuk mendapatkan data tersebut. Prosedur ini dinilai tidak efisien, terutama ketika data dibutuhkan secara cepat pada saat akreditasi program studi. Kondisi tersebut menggambarkan

hambatan yang dialami oleh pengguna sistem dalam mengakses informasi yang dibutuhkan.

b. Data Sekunder

Data sekunder yang digunakan dalam penelitian ini adalah data kinerja dosen bidang pendidikan dan penelitian. Data tersebut diperoleh dari sistem yang telah tersedia sebelumnya dan telah melalui proses pengumpulan serta pengolahan. Pemanfaatan data ini bertujuan untuk memberikan informasi yang relevan dan mendalam dalam proses analisis serta pengembangan sistem, khususnya dalam hal integrasi data kinerja dosen agar dapat dimanfaatkann secara optimal untuk kebutuhan akreditasi.

3.3.4 Penerapan Metode API-*First Development*

Pada penelitian ini digunakan pendekatan API-*First Development*, yaitu metode yang mengutamakan perancangan *Application Programming Interface* (API) sejak tahap awal pengembangan perangkat lunak. API dirancang sebagai komponen utama agar integrasi dan konsistensi antar sistem dapat terjaga sejak awal proses pengembangan. Tahapan-tahapan dalam metode ini dijelaskan sebagai berikut.

1. *Define*

Pada tahap *Define*, dilakukan identifikasi terhadap kebutuhan pengguna dan sistem untuk menghasilkan gambaran fungsionalitas yang akan dikembangkan. Tahapan ini bertujuan untuk memperoleh pemahaman mengenai kebutuhan fungsional dan nonfungsional sistem, serta menyusun spesifikasi teknis yang mendukung pengembangan sesuai dengan tujuan. Hasil dari identifikasi ini kemudian divisualisasikan ke dalam beberapa diagram perancangan. *Use case diagram* digunakan

untuk menggambarkan hubungan dan interaksi antara pengguna dengan sistem, sedangkan *activity diagram* digunakan untuk memetakan alur aktivitas dalam setiap proses yang terjadi. Selain itu, *sequence diagram* digunakan untuk memperjelas urutan interaksi antar objek dalam sistem. Penyusunan diagram-diagram tersebut tidak hanya membantu pengembang dalam memahami sistem, tetapi juga dapat meminimalkan risiko kesalahpahaman dan meningkatkan efisiensi selama proses perancangan API.

a. Mengidentifikasi Kebutuhan Pengguna dan Sistem

Kebutuhan fungsional merupakan gambaran mengenai fitur-fitur utama yang harus tersedia dalam sistem untuk mendukung pencapaian tujuan serta kelancaran proses kerja yang telah dirancang. Kebutuhan ini menjadi dasar dalam menentukan fungsi-fungsi yang harus diimplementasikan pada sistem. Adapun beberapa kebutuhan fungsional tersebut antara lain:

a) Sistem mampu melakukan proses *login* dan *logout*
Sistem menyediakan mekanisme autentikasi yang aman dengan menggunakan JSON *Web* Token (JWT) untuk mendukung proses *login* dan *logout*. Saat pengguna berhasil melakukan *login*, sistem akan mengarahkan sesuai dengan peran masing-masing, seperti admin, dekan, kaprodi, atau dosen. Ketika pengguna melakukan *logout*, token yang telah diterbitkan akan dinonaktifkan agar sesi berakhir dan tidak dapat digunakan kembali tanpa autentikasi yang sah.

b) Sistem mampu melakukan pengelolaan data pengguna
Sistem memberikan hak akses kepada admin untuk melakukan pengelolaan data pengguna, yang meliputi penambahan, pembaruan, dan penghapusan data dosen. Setiap akun dosen yang diinputkan ke dalam sistem harus dilengkapi dengan

informasi yang lengkap dan valid untuk menjamin keakuratan data yang tersimpan dalam basis data.

c) Sistem mampu menampilkan data kinerja bidang pendidikan dan penelitian

Sistem dirancang untuk menampilkan laporan kinerja dosen dalam bidang pendidikan dan penelitian dalam bentuk tabel yang informatif dan mudah dipahami. Laporan ini dapat diakses oleh pihak yang memiliki wewenang seperti, admin, dekan, kaprodi, maupun dosen, sesuai dengan peran dan hak akses yang telah ditentukan dalam sistem.

Selanjutnya, kebutuhan nonfungsional sistem berfokus pada aspek kualitas dan kinerja dalam pengembangan sistem. Kebutuhan ini bertujuan untuk memastikan sistem dapat berjalan secara optimal, aman, dan mampu beradaptasi dengan perkembangan pengguna. Adapun kebutuhan nonfungsional dijelaskan sebagai berikut:

a) Performa

Sistem harus mampu memberikan waktu respons yang cepat dalam memproses data maupun menampilkan informasi. Hal ini, agar pengguna dapat mengakses serta mengelola data secara efisien tanpa mengalami keterlambatan yang mengganggu kinerja.

b) Skalabilitas

Sistem yang dikembangkan mampu menangani peningkatan jumlah pengguna seiring waktu tanpa menyebabkan penurunan performa. Selain itu, sistem juga harus dapat menyimpan data historis kinerja dosen dalam jangka panjang tanpa mengganggu efisiensi dalam pengolahan dan penyajian data.

c) Keamanan

Keamanan merupakan aspek penting dalam pengembangan sistem, khususnya untuk melindungi data dari akses yang tidak sah. Salah satu cara yang diterapkan untuk menjaga keamanan adalah dengan melakukan enkripsi terhadap data, sehingga hanya pihak yang memiliki otorisasi yang dapat mengakses informasi tersebut. Selain itu, untuk keamanan juga digunakan mekanisme autentikasi berbasis JSON *Web* Token (JWT).

Tabel 13. Para *Stakeholder* yang terlibat dalam penelitian

No	Nama Anggota	Peran
1.	M. Iqbal Parabi	*Product Owner*
2.	Shafa Auliya	*Developer Team*
3.	Fathimah Abiyyi Khairunnisa	*Developer Team*
4.	Amalia Nurul Rahmawati	*Developer Team*
5.	Chandra Prasetya Putra	*Developer Team*
6.	UPT TIK Unuversitas Lampung	*Business Stakeholder*

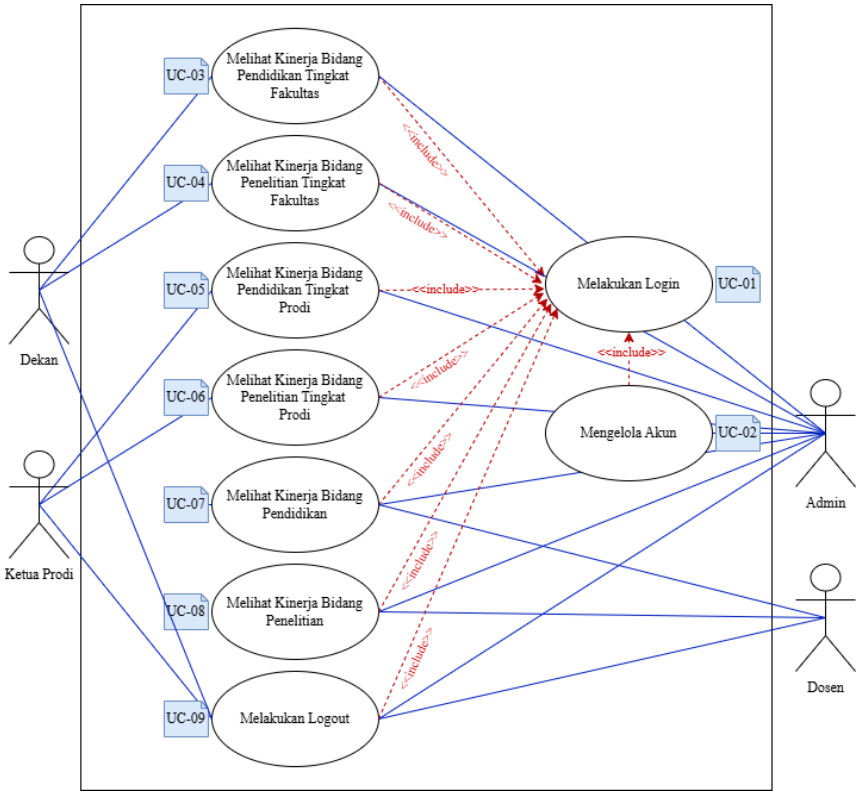
b. *Use Case Diagram*

Pada *use case diagram* sistem remunerasi dosen Universitas Lampung bidang pendidikan dan penelitian, terdapat empat aktor utama yang memiliki peran dalam mengakses sistem, yaitu admin, dekan, ketua program studi (kaprodi), dan dosen. Masing-masing aktor diberikan hak akses yang berbeda sesuai dengan tugas dan tanggung jawabnya. Pembagian hak akses ini bertujuan untuk menjamin bahwa setiap pengguna hanya dapat mengakses informasi yang sesuai dengan perannya, sehingga integrasi dan keamanan data tetap terjaga.

Admin memiliki peran utama dalam pengelolaan akun pengguna, yang mencakup penambahan, pembaruan, dan penghapusan data pengguna sesuai kebutuhan. Selain itu, admin juga memiliki

memiliki hak akses untuk melihat data kinerja dosen berdasarkan fakultas, program studi, serta bidang kinerja yang meliputi bidang pendidikan dan penelitian. Dengan tanggung jawab tersebut, admin berperan penting dalam memastikan validitas dan keakuratan data yang terdapat dalam sistem.

Sementara itu, dekan diberikan hak akses untuk melihat data kinerja seluruh dosen dari berbagai program studi dalam lingkup fakultas. Ketua program studi memiliki akses terbatas hanya pada data kinerja dosen yang berada dalam program studinya masing-masing. Akses ini berguna untuk mendukung proses pemantauan kinerja dosen serta keperluan akreditasi. Sedangkan, dosen hanya diberikan akses terhadap data kinerja pribadi. Rancangan *use case* diagram dari sistem ini dapat dilihat pada Gambar 5.



Gambar 5. *Use Case Diagram* Sistem Remunerasi Bidang Pendidikan dan Penelitian

Aturan penomoran dan penamaan *use case diagram* sesuai dengan Tabel 14 berikut.

Tabel 14. Penomoran *Use Case Diagram*

No	*Use Case Diagram*	Penomoran
1.	Melakukan *Login*	UC-01
2.	Mengelola Akun	UC-02
3.	Melihat Kinerja Bidang Pendidikan Tingkat Fakultas	UC-03
4.	Melihat Kinerja Bidang Penelitian Tingkat Fakultas	UC-04
5.	Melihat Kinerja Bidang Pendidikan Tingkat Prodi	UC-05
6.	Melihat Kinerja Bidang Penelitian Tingkat Prodi	UC-06
7.	Melihat Kinerja Bidang Pendidikan	UC-07
8.	Melihat Kinerja Bidang Penelitian	UC-08
9.	Melakukan *Logout*	UC-09

Adapun *use case description* digunakan untuk menjelaskan secara rinci interaksi antara aktor dengan sistem dalam menjalankan suatu proses tertentu. Deskripsi ini mencakup alur kejadian, kondisi awal, kondisi akhir, serta kemungkinan pengecualian yang dapat terjadi selama sistem dijalankan. Informasi tersebut penting untuk menggambarkan bagaimana sistem merespons tindakan pengguna dalam berbagai skenario penggunaan yang mungkin terjadi.

Berikut ini merupakan *use case description* untuk sistem remunerasi dosen.

a) *Use Case Description* Melakukan *Login* (UC-01)

Use case Melakukan *Login* merupakan proses autentikasi yang bertujuan untuk memastikan bahwa hanya pengguna yang terdaftar dan memiliki hak akses sesuai peran yang dapat masuk ke dalam sistem. Proses ini diawali saat aktor (Admin,

Dekan, Kaprodi, atau Dosen) mengakses halaman *login* dan memasukkan NIDN beserta *password*. Selanjutnya, sistem akan mengirimkan permintaan verifikasi ke basis data untuk mencocokkan kredensial yang dimasukkan dengan data yang tersimpan. Apabila kredensial yang dimasukkan dinyatakan valid, maka sistem akan mengeluarkan token otorisasi, menyimpannya, dan mengarahkan aktor ke halaman utama sesuai dengan peran masing-masing. Rincian *use case description* ini dapat dilihat pada Tabel 15.

Tabel 15. *Use Case Description* Melakukan *Login*

Use Case Name	Melakukan *Login*
ID	UC-01
Priority	*High*
Actor	Admin, Dekan, Kaprodi, Dosen
Description	Aktor melakukan *login* ke sistem menggunakan NIDN dan *password* yang sudah terdaftar pada *database*.
Triger	Aktor melakukan *login*.
Preconditions	Aktor sudah terdaftar di sistem.
Normal Course	1. Aktor mengakses halaman *login*. 2. Aktor memasukkan NIDN dan *password*. 3. Sistem mengirim permintaan ke *database*. 4. *Database* memvalidasi kredensial akun. 5. Jika valid, sistem mengirim token *authorization*. 6. *Database* menyimpan token. 7. Sistem memvalidasi token. 8. Jika valid, sistem akan menampilkan halaman *user*.
Postconditions	Aktor berhasil *login* dan dapat mengakses fitur-fitur sistem sesuai dengan perannya.
Sub Flows	Actor lupa *password.*
Alternate/ Exceptional Flow	A1: Jika NIDN atau *password* salah, maka sistem akan menampilkan pesan *error* dan meminta aktor untuk mencoba lagi. A2: Jika aktor lupa *password*, sistem akan menampilkan pesan untuk meminta aktor menghubungi admin.

	A3: Jika aktor telah melewati batas waktu sesi tertentu, maka secara otomatis token akan berubah menjadi tidak aktif dan sesi akan berakhir.

b) *Use Case Description* Mengelola Akun Pengguna

Use case Mengelola Akun Pengguna menggambarkan proses administratif yang dilakukan oleh aktor dengan peran sebagai admin dalam mengelola data akun pengguna pada sistem. Aktivitas ini meliputi penambahan, pengubahan, dan penghapusan akun, serta hanya dapat dilakukan setelah admin berhasil melakukan *login* dan memperoleh sesi aktif dalam sistem. Setelah berhasil masuk, admin akan ditampilkan daftar akun pengguna yang telah tersimpan. Melalui antarmuka tersebut, admin dapat memilih aksi yang diinginkan, seperti menambah akun baru, mengubah data akun yang sudah ada, maupun menghapus akun tertentu.

Setiap aksi yang dilakukan akan memicu interaksi antara sistem dan basis data untuk memproses dan menyimpan perubahan yang dilakukan. Hasil dari proses tersebut akan ditampilkan dalam bentuk pembaruan daftar akun. Apabila terjadi kesalahan dalam pengisian data, sistem akan memberikan pesan kesalahan. Rincian *use case description* ini dapat dilihat pada Tabel 16.

Tabel 16. *Use Case Description* Mengelola Akun Pengguna

Use Case Name	Mengelola Akun Pengguna
ID	UC-02
Priority	*Medium*
Actor	Admin
Description	Aktor dapat melakukan kelola akun pengguna.
Triger	Aktor memilih salah satu aksi kelola akun.

Preconditions	Aktor sudah *login* sebagai Admin dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan daftar berisi data akun pengguna. 3. Aktor memilih aksi menambah akun. 3.1 Sistem menampilkan formulir tambah akun 3.2 Aktor mengisi *form* berisi data yang dibutuhkan untuk menambahkan akun. 3.3 Aktor melakukan *submit form*. 3.4 Sistem mengirimkan *request* berisi data akun baru yang dimasukkan Aktor. 3.5 *Database* menyimpan dan mengembalikan *response* daftar akun yang baru. 4. Aktor memilih aksi mengubah data akun. 4.1 Sistem menampilkan formulir ubah data akun. 4.2 Aktor mengubah *form* berisi data sebelumnya menjadi data yang ingin diubah. 4.3 Aktor melakukan *submit form.* 4.4 Sistem mengirimkan *request* berisi data akun baru yang diubah *actor*. 4.5 *Database* menyimpan dan mengembalikan *response* data akun yang baru. 5. Aktor memilih aksi menghapus data akun. 5.1 Aktor memvalidasi pilihan hapus akun. 5.2 Sistem mengirimkan *reques*t hapus akun. 5.3 *Database* menghapus akun dan mengembalikan *response*. 6. Sistem menampilkan seluruh daftar akun pengguna.
Postconditions	Aktor dapat menambah, mengubah, dan menghapus akun pengguna.
Sub Flows	Aktor gagal Menambah Akun, Mengedit Data Akun, Menghapus Akun.
Alternate/ Exceptional Flow	A1: Jika data pengguna yang dimasukkan tidak valid, maka sistem akan menampilkan pesan *error* dan meminta aktor untuk memperbaiki data. A2: Jika data pengguna yang ingin dihapus dibatalkan, sistem akan mengembalikan ke halaman daftar pengguna tanpa perubahan.

c) *Use Case Description* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas

Use case Melihat Kinerja Dosen Bidang Pendidikan Tingkat Fakultas menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan dekan dalam mengakses data kinerja dosen bidang pendidikan dalam lingkup fakultas. Akses ini hanya dapat dilakukan setelah aktor berhasil *login* dan memperoleh sesi aktif di dalam sistem. Setelah berhasil masuk, aktor dapat memilih program studi dan nama dosen dari daftar yang tersedia untuk melihat data kinerja. Sistem kemudian akan memperoses permintaan dan menampilkan informasi secara terstruktur berdasarkan hasil pemilihan tersebut. Rincian *use case description* ini dapat dilihat pada Tabel 17.

Tabel 17. *Use Case Description* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas

Use Case Name	Melihat Kinerja Bidang Pendidikan Tingkat Fakultas
ID	UC-03
Priority	*Medium*
Actor	Admin, Dekan
Description	Aktor mengakses fitur melihat kinerja bidang pendidikan tingkat fakultas berdasarkan program studi dan dosen yang dipilih.
Triger	Aktor memilih fitur melihat kinerja bidang pendidikan tingkat fakultas.
Preconditions	Aktor sudah *login* sebagai Dekan dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data fakultas beserta daftar program studi. 3. Aktor memilih program studi. 4. Sistem meminta data program studi. 5. *Database* mengembalikan data program studi. 6. Sistem menampilkan data program studi dan daftar dosen.

	7. Aktor memilih dosen yang ingin dilihat.
Postconditions	Aktor berhasil memilih program studi dan dosen serta mendapatkan data kinerja bidang pendidikan berdasarkan pilihan tersebut.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika data dosen tidak ditemukan, maka sistem akan menampilkan pesan "Data tidak ditemukan". A2: Jika aktor melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan "Data tidak ditemukan".

d) *Use Case Description* Melihat Kinerja Bidang Penelitian Tingkat Fakultas

Use case Melihat Kinerja Dosen Bidang Penelitian Tingkat Fakultas menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan dekan dalam mengakses data kinerja dosen bidang penelitian dalam lingkup fakultas. Proses ini hanya dapat dilakukan setelah aktor berhasil *login* dan memiliki sesi yang aktif di dalam sistem. Setelah berhasil masuk, aktor dapat memilih program studi dan nama dosen dari daftar yang tersedia untuk menampilkan data kinerja yang relevan. Sistem akan memproses permintaan tersebut dengan mengambil data dari basis data dan menyajikannya dalam format yang terstruktur. Rincian *use case description* ini dapat dilihat pada Tabel 18.

Tabel 18. *Use Case Description* Melihat Kinerja Bidang Penelitian Tingkat Fakultas

Use Case Name	Melihat Kinerja Bidang Penelitian Tingkat Fakultas
ID	UC-04
Priority	*Medium*
Actor	Admin, Dekan
Description	Aktor mengakses fitur melihat kinerja bidang penelitian tingkat fakultas

	berdasarkan program studi dan dosen yang dipilih.
Triger	Aktor memilih fitur melihat kinerja bidang penelitian tingkat fakultas.
Preconditions	Aktor sudah *login* sebagai Dekan dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data fakultas beserta daftar program studi. 3. Aktor memilih program studi. 4. Sistem meminta data program studi. 5. *Database* mengembalikan data program studi. 6. Sistem menampilkan data program studi dan daftar dosen. 7. Aktor memilih dosen yang ingin dilihat. 8. Sistem meminta data dosen. 9. *Database* mengembalikan data dosen. 10. Sistem menampilkan data dosen dan bidang kinerja. 11. Aktor memilih kinerja bidang penelitian. 12. Sistem meminta data bidang penelitian. 13. *Database* mengembalikan data bidang penelitian. 14. Sistem menampilkan data bidang penelitian.
Postconditions	Aktor berhasil memilih program studi dan dosen serta mendapatkan data kinerja bidang penelitian berdasarkan pilihan tersebut.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika data dosen tidak ditemukan, maka sistem akan menampilkan pesan “Data tidak ditemukan”. A2: Jika aktor melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan “Data tidak ditemukan”.

e) *Use Case Description* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi

Use case Melihat Kinerja Dosen Bidang Pendidikan Tingkat Program Studi menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan kaprodi dalam

mengakses data kinerja dosen bidang pendidikan pada tingkat program studi. Akses ini hanya dapat dilakukan setelah aktor berhasil *login* ke dalam sistem dan memiliki status sesi yang aktif. Setelah berhasil masuk, sistem akan menampilkan data program studi beserta daftar dosen yang berada di dalamnya. Aktor kemudian dapat memilih salah satu dosen tertentu untuk melihat rincian data kinerja dosen bidang pendidikan. Permintaan data yang dilakukan oleh aktor akan diproses oleh sistem dengan mengambil informasi dari basis data, lalu ditampilkan dalam bentuk yang terstruktur dan informatif. Rincian *use case description* ini dapat dilihat pada Tabel 19.

Tabel 19. *Use Case Description* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi

Use Case Name	Melihat Kinerja Bidang Pendidikan Tingkat Program Studi
ID	UC-05
Priority	*Medium*
Actor	Admin, Kaprodi
Description	Aktor mengakses fitur melihat kinerja bidang pendidikan tingkat program studi berdasarkan dosen yang dipilih.
Triger	Aktor memilih fitur melihat kinerja bidang pendidikan tingkat program studi.
Preconditions	Aktor sudah *login* sebagai Kaprodi dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data program studi beserta daftar dosen. 3. Aktor memilih dosen yang ingin dilihat. 4. Sistem meminta data dosen. 5. *Database* mengembalikan data dosen. 6. Sistem menampilkan data dosen dan bidang kinerja. 7. Aktor memilih kinerja bidang pendidikan. 8. Sistem meminta data bidang pendidikan. 9. *Database* mengembalikan data bidang pendidikan. 10. Sistem menampilkan data bidang pendidikan.

Postconditions	Aktor berhasil memilih dosen serta mendapatkan data kinerja bidang pendidikan berdasarkan pilihan tersebut.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika data dosen tidak ditemukan, maka sistem akan menampilkan pesan "Data tidak ditemukan". A2: Jika aktor melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan "Data tidak ditemukan".

f) *Use Case Description* Melihat Kinerja Bidang Penelitian Tingkat Program Studi

Use case Melihat Kinerja Dosen Bidang Pendidikan Tingkat Program Studi menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan kaprodi dalam mengakses data kinerja dosen bidang penelitian pada tingkat program studi. Akses ini hanya dapat dilakukan setelah aktor berhasil *login* ke dalam sistem dan memiliki status sesi yang aktif. Setelah berhasil masuk, sistem akan menampilkan data program studi beserta daftar dosen yang berada di dalamnya. Aktor kemudian dapat memilih salah satu dosen tertentu untuk melihat rincian data kinerja dosen bidang penelitian. Permintaan data yang dilakukan oleh aktor akan diproses oleh sistem dengan mengambil informasi dari basis data, lalu ditampilkan dalam bentuk yang terstruktur dan informatif. Rincian *use case description* ini dapat dilihat pada Tabel 20.

Tabel 20. *Use Case Description* Melihat Kinerja Bidang Penelitian Tingkat Program Studi

Use Case Name	Melihat Kinerja Bidang Penelitian Tingkat Program Studi
ID	UC-06
Priority	*Medium*

Actor	Admin, Kaprodi
Description	Aktor mengakses fitur melihat kinerja bidang penelitian tingkat program studi berdasarkan dosen yang dipilih.
Triger	Aktor memilih fitur melihat kinerja bidang penelitian tingkat program studi.
Preconditions	Aktor sudah *login* sebagai Kaprodi dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data program studi beserta daftar dosen. 3. Aktor memilih dosen yang ingin dilihat. 4. Sistem meminta data dosen. 5. *Database* mengembalikan data dosen. 6. Sistem menampilkan data dosen dan bidang kinerja. 7. Aktor memilih kinerja bidang penelitian. 8. Sistem meminta data bidang penelitian. 9. *Database* mengembalikan data bidang penelitian. 10. Sistem menampilkan data bidang penelitian.
Postconditions	Aktor berhasil memilih dosen serta mendapatkan data kinerja bidang penelitian berdasarkan pilihan tersebut.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika data dosen tidak ditemukan, maka sistem akan menampilkan pesan “Data tidak ditemukan”. A2: Jika aktor melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan “Data tidak ditemukan”.

g) *Use Case Description* Melihat Kinerja Bidang Pendidikan

Use case Melihat Kinerja Dosen Bidang Pendidikan menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan dosen untuk mengakses data kinerja dosen bidang pendidikan. Akses terhadap data ini dimulai setelah aktor berhasil melakukan *login* ke dalam sistem dan memiliki status sesi yang aktif. Setelah berhasil masuk, sistem akan menampilkan data kinerja milik dirinya sendiri dalam

bidang pendidikan. Rincian *use case description* ini dapat dilihat pada Tabel 21.

Tabel 21. *Use Case Description* Melihat Kinerja Bidang Pendidikan

Use Case Name	Melihat Kinerja Bidang Pendidikan
ID	UC-07
Priority	*High*
Actor	Admin, Dosen
Description	Aktor mengakses fitur melihat kinerja bidang pendidikan yang hanya menampilkan data miliknya sendiri.
Triger	Aktor memilih fitur melihat kinerja bidang pendidikan.
Preconditions	Aktor sudah *login* sebagai Dosen dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data dosen beserta kinerjanya. 3. Aktor memilih kinerja bidang pendidikan. 4. Sistem meminta data bidang pendidikan. 5. *Database* mengembalikan data bidang pendidikan. 6. Sistem menampilkan data bidang pendidikan.
Postconditions	Aktor berhasil mendapatkan data kinerja bidang pendidikan.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika aktor melakukan *filter* data kinerja bidang pendidikan berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan "Data tidak ditemukan".

h) *Use Case Description* Melihat Kinerja Bidang Penelitian

Use case Melihat Kinerja Dosen Bidang Pendidikan menggambarkan proses yang dilakukan oleh aktor dengan peran sebagai admin dan dosen untuk mengakses data kinerja dosen bidang penelitian. Akses terhadap data ini dimulai

setelah aktor berhasil melakukan *login* ke dalam sistem dan memiliki status sesi yang aktif. Setelah berhasil masuk, sistem akan menampilkan data kinerja milik dirinya sendiri dalam bidang penelitian. Rincian *use case description* ini dapat dilihat pada Tabel 22.

Tabel 22. *Use Case Description* Melihat Kinerja Bidang Penelitian

Use Case Name	Melihat Kinerja Bidang Penelitian
ID	UC-08
Priority	*High*
Actor	Dosen
Description	Aktor mengakses fitur melihat kinerja bidang penelitian yang hanya menampilkan data miliknya sendiri.
Triger	Aktor memilih fitur melihat kinerja bidang penelitian.
Preconditions	Aktor sudah *login* sebagai Dosen dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Sistem menampilkan data dosen beserta kinerjanya. 3. Aktor memilih kinerja bidang penelitian. 4. Sistem meminta data bidang penelitian. 5. *Database* mengembalikan data bidang penelitian. 6. Sistem menampilkan data bidang penelitian.
Postconditions	Aktor berhasil mendapatkan data kinerja bidang penelitian.
Sub Flows	Aktor dapat melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun.
Alternate/ Exceptional Flow	A1: Jika aktor melakukan *filter* data kinerja bidang penelitian berdasarkan semester dan tahun, namun data kinerja tidak ada, maka sistem akan menampilkan pesan "Data tidak ditemukan".

i) *Use Case Description* Melakukan *Logout*

Use case Melakukan *Logout* menggambarkan proses yang dilakukan oleh pengguna sistem, yaitu admin, dekan, kaprodi,

maupun dosen, untuk keluar dari sistem dan mengakhiri sesi yang sedang berlangsung. Proses ini dilakukan setelah pengguna berada dalam sistem dan memiliki status sesi yang aktif. Pengguna dapat memilih menu *logout* pada antarmuka sistem, kemudian sistem akan memproses permintaan tersebut dengan memperbarui status sesi menjadi nonaktif. Setelah itu, pengguna secara otomatis akan diarahkan kembali ke halaman *login*. Selain melalui tindakan langsung dari pengguna, sistem juga dirancang untuk mengakhiri sesi secara otomatis apabila waktu sesi telah habis atau melebihi batas waktu yang ditentukan. Rincian *use case description* ini dapat dilihat pada Tabel 23.

Tabel 23. *Use Case Description* Melakukan *Logout*

Use Case Name	Melakukan *Logout*
ID	UC-09
Priority	*Medium*
Actor	Admin, Dekan, Kaprodi, Dosen
Description	Aktor melakukan *logout* untuk keluar dari sistem.
Triger	Aktor melakukan *logout*.
Preconditions	Aktor sudah *login* dan memiliki status sesi yang aktif.
Normal Course	1. Aktor sudah berada dalam sistem dan memiliki status sesi yang aktif. 2. Aktor memilih menu *logout*. 3. Sistem memproses permintaan *logout*. 4. *Database* memperbarui status sesi menjadi nonaktif. 5. Sistem akan mengarahkan *actor* ke halaman *login*.
Postconditions	Aktor berhasil *logout* dan keluar dari sistem.
Sub Flows	Status sesi aktor berubah menjadi nonaktif.
Alternate/ Exceptional Flow	A1: Jika waktu sesi habis, maka sistem akan mengakhiri sesi secara otomatis. A2: Jika aktor *logout* dari satu perangkat tetapi masih *login* di perangkat lain, maka sistem akan secara otomatis mengakhiri sesi di semua perangkat.

c. *Activity Diagram*

Activity diagram merupakan representasi visual dari rangkaian aktivitas yang terjadi dalam suatu sistem. Diagram ini berfungsi untuk menggambarkan alur mulai dari awal hingga akhir, serta menunjukkan keterkaitan antaraktivitas yang berlangsung di dalam sistem. Melalui penyajian ini, pihak yang terlibat dalam pengembangan sistem dapat memahami secara menyeluruh bagaimana proses dijalankan dalam sistem. Hal ini dapat memudahkan dalam mengidentifikasi alur logika dan interaksi antara pengguna dengan sistem. Dengan demikian, proses perancangan dan implementasi sistem menjadi lebih terarah dan minim risiko kesalahan.

a) *Activity Diagram* Melakukan *Login* (UC-01)

Pada *activity diagram* ini menggambarkan urutan aktivitas yang dilakukan oleh pengguna saat akan mengakses sistem. Proses diawali ketika pengguna membuka halaman *login* dan memasukkan informasi identitas berupa NIDN dan *password*. Informasi ini kemudian diproses oleh sistem untuk dilakukan verifikasi.

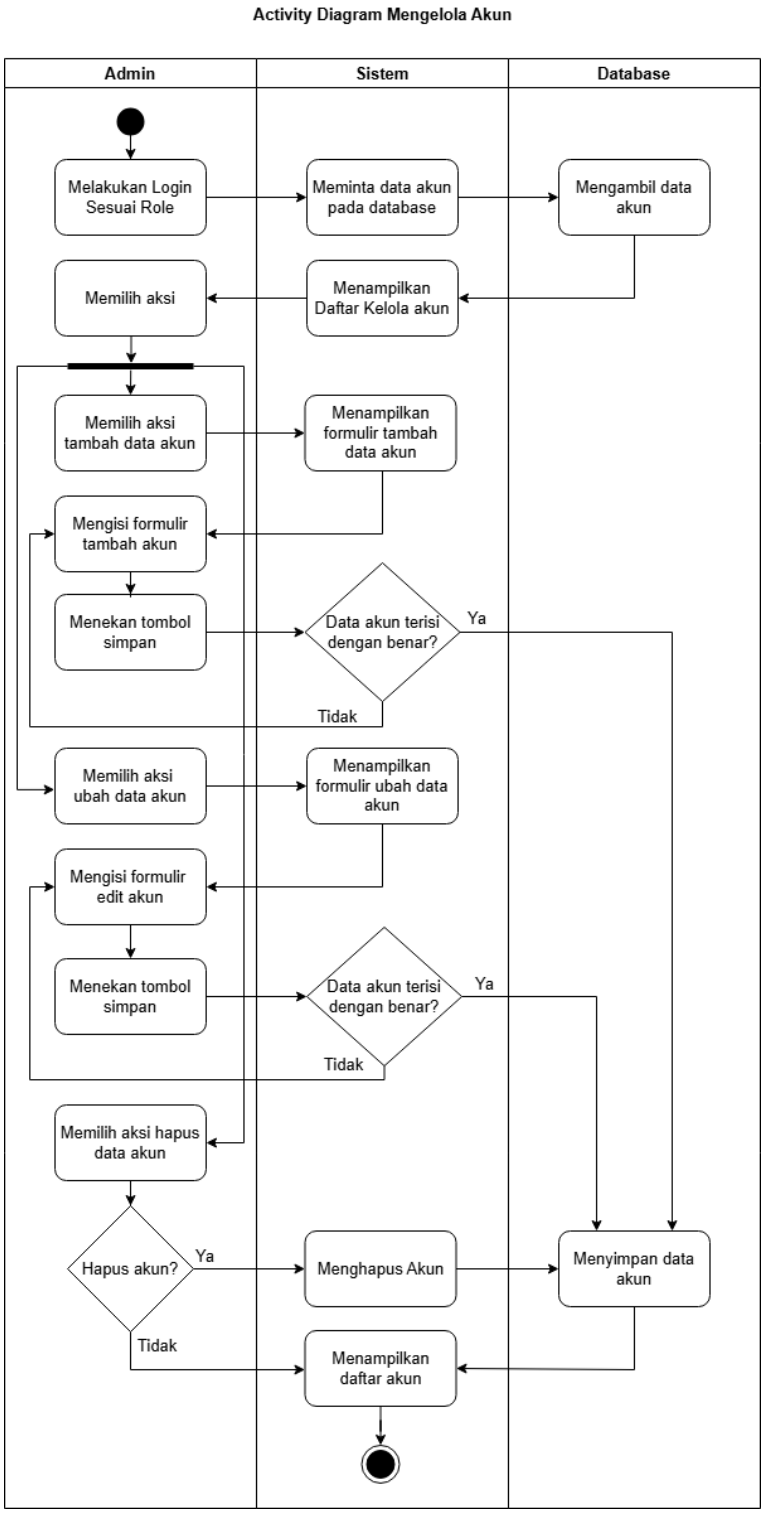
Jika data yang dimasukkan sesuai dengan data yang tersimpan di dalam basis data, maka sistem memberikan akses masuk dan mengarahkan pengguna ke halaman utama sesuai dengan peran yang dimiliki, baik sebagai admin, dekan, kaprodi, maupun dosen. Sebaliknya, apabila data yang dimasukkan tidak valid, sistem akan menampilkan pesan kesalahan dan meminta pengguna untuk mengulangi proses *login*. Rancangan *activity diagram* dari proses *login* dapat dilihat pada Gambar 6.

Gambar 6. *Activity Diagram* Melakukan *Login*

b) *Activity Diagram* Mengelola Akun Pengguna (UC-02)

Activity diagram ini menjelaskan alur aktivitas yang dilakukan oleh admin dalam mengelola akun pengguna pada sistem. Pengelolaan akun diawali dengan proses *login* oleh admin sesuai dengan peran yang telah ditentukan. Setelah berhasil masuk, sistem akan menampilkan daftar akun yang tersedia berdasarkan data yang diambil dari basis data.

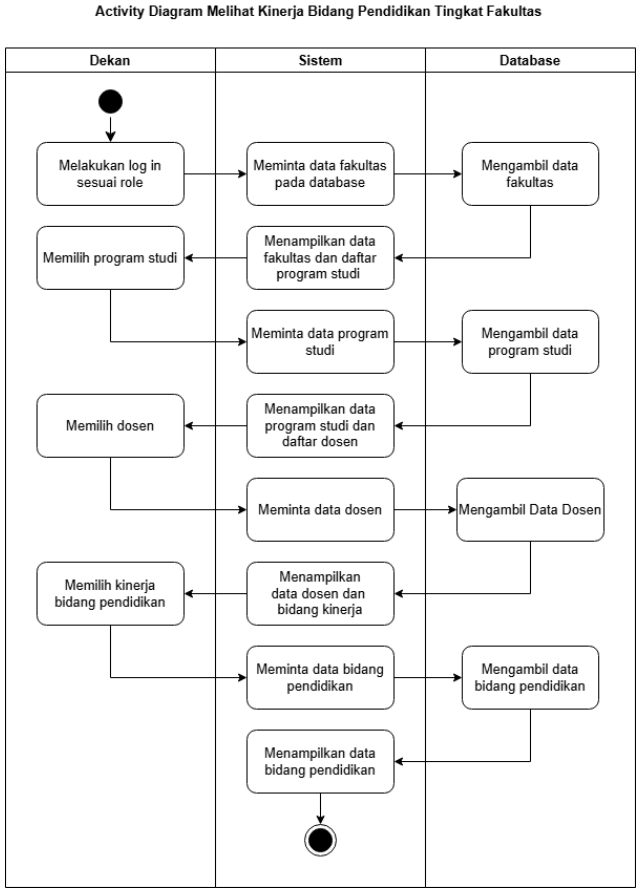
Dalam mengelola akun admin dapat melakukan tindakan, yaitu menambahkan, mengubah, atau menghapus data pengguna. Rancangan *activity diagram* ini dapat dilihat pada Gambar 7.



Gambar 7. *Activity Diagram* Mengelola Akun Pengguna

c) *Activity Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas (UC-03)

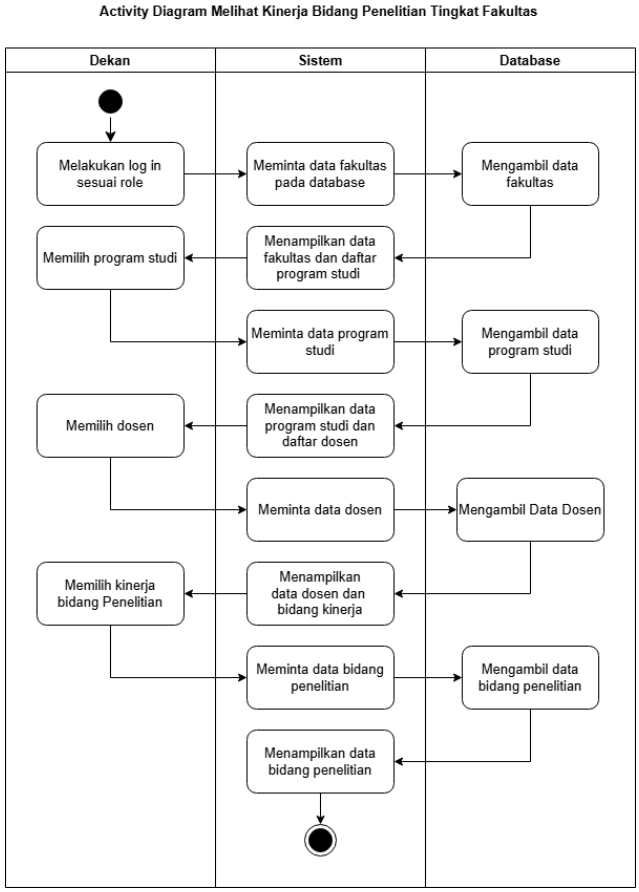
Pada *activity diagram* ini menggambarkan alur aktivitas dekan dalam memantau kinerja dosen bidang pendidikan di tingkat fakultas. Pemantauan ini dilakukan setelah dekan berhasil masuk ke dalam sistem dan memiliki sesi yang aktif. Selanjutnya, dekan memilih program studi dan dosen yang ingin dilihat berdasarkan data yang tersedia. Sistem akan memproses permintaan tersebut dan menampilkan informasi kinerja dosen sesuai dengan data yang tersimpan dalam basis data. Rancangan *activity diagram* ini dapat dilihat pada Gambar 8.



Gambar 8. *Activity Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas

d) *Activity Diagram* Melihat Kinerja Bidang Penelitian Tingkat Fakultas (UC-04)

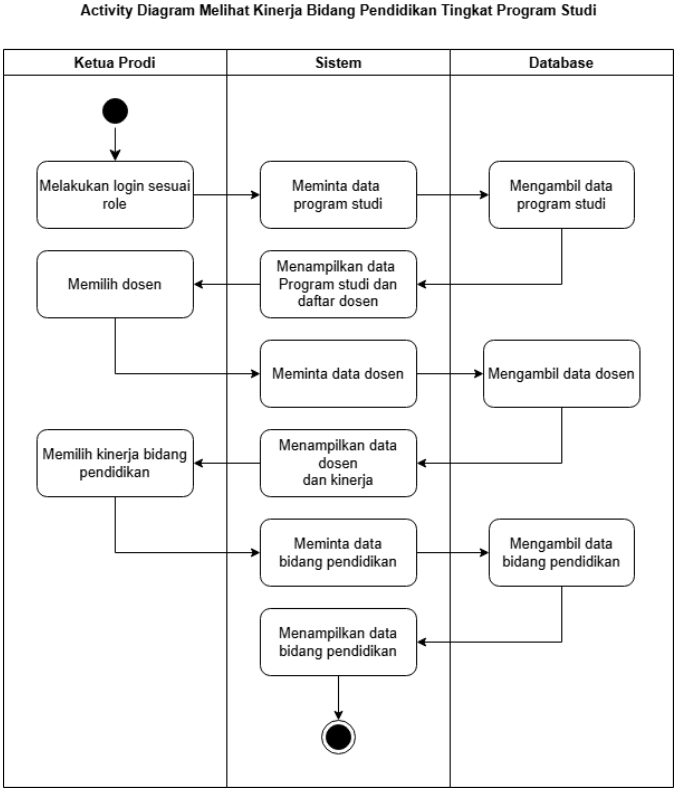
Pada *activity diagram* ini menggambarkan alur aktivitas dekan dalam memantau kinerja dosen bidang penelitian di tingkat fakultas. Pemantauan ini dilakukan setelah dekan berhasil masuk ke dalam sistem dan memiliki sesi yang aktif. Selanjutnya, dekan memilih program studi dan dosen yang ingin dilihat berdasarkan data yang tersedia. Sistem akan memproses permintaan tersebut dan menampilkan informasi kinerja dosen sesuai dengan data yang tersimpan dalam basis data. Rancangan *activity diagram* ini dapat dilihat pada Gambar 9.



Gambar 9. *Activity Diagram* Melihat Kinerja Bidang Penelitian Tingkat Fakultas

e) *Activity Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi (UC-05)

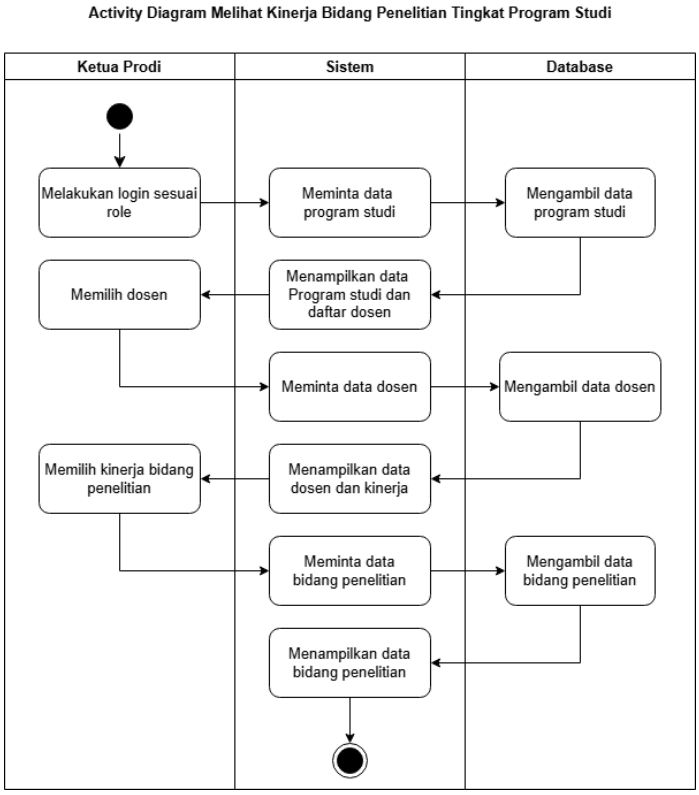
Pada *activity diagram* ini menggambarkan alur aktivitas kaprodi dalam memantau kinerja dosen bidang pendidikan di tingkat program studi. Pemantauan ini dilakukan setelah kaprodi berhasil masuk ke dalam sistem dan memiliki sesi yang aktif. Selanjutnya, kaprodi memilih dosen serta jenis kinerja bidang pendidikan yang ingin dilihat berdasarkan data yang tersedia. Sistem akan memproses permintaan tersebut dan menampilkan informasi kinerja dosen sesuai dengan data yang tersimpan dalam basis data. Rancangan *activity diagram* ini dapat dilihat pada Gambar 10.



Gambar 10. *Activity Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi

f) *Activity Diagram* Melihat Kinerja Bidang Penelitian Tingkat Program Studi (UC-06)

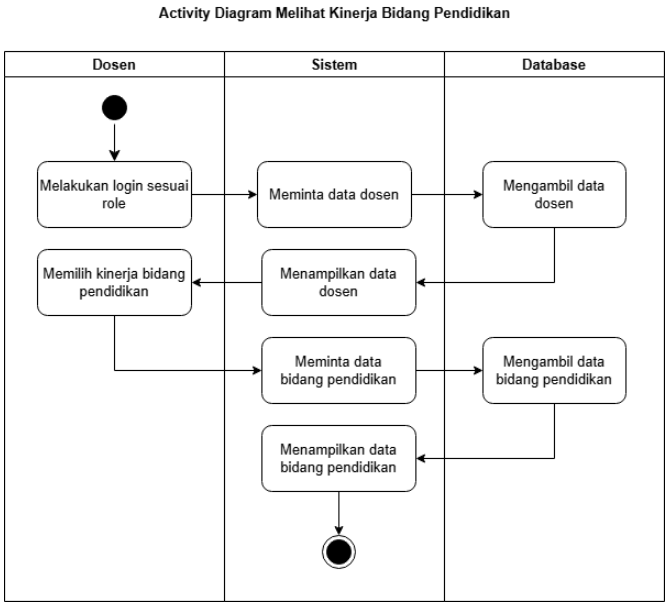
Pada *activity diagram* ini menggambarkan alur aktivitas kaprodi dalam memantau kinerja dosen bidang penelitian di tingkat program studi. Pemantauan ini dilakukan setelah kaprodi berhasil masuk ke dalam sistem dan memiliki sesi yang aktif. Selanjutnya, kaprodi memilih dosen serta jenis kinerja bidang penelitian yang ingin dilihat berdasarkan data yang tersedia. Sistem akan memproses permintaan tersebut dan menampilkan informasi kinerja dosen sesuai dengan data yang tersimpan dalam basis data. Rancangan *activity diagram* ini dapat dilihat pada Gambar 11.



Gambar 11. *Activity Diagram* Melihat Kinerja Bidang Penelitian Tingkat Program Studi

g) *Activity* Diagram Melihat Kinerja Bidang Pendidikan (UC-07)

Pada *activity diagram* ini menggambarkan alur aktivitas dosen dalam memantau kinerjanya pada bidang pendidikan. Proses diawali saat dosen berhasil masuk ke dalam sistem dan memiliki sesi yang aktif. Setelah itu, dosen dapat memilih fitur untuk melihat data kinerja bidang pendidikan yang telah tersedia di dalam basis data. Sistem akan memproses permintaan tersebut dan menampilkan informasi berdasarkan identitas serta data yang berkaitan dengan dosen tersebut. Rancangan *activity diagram* ini dapat dilihat pada Gambar 12.

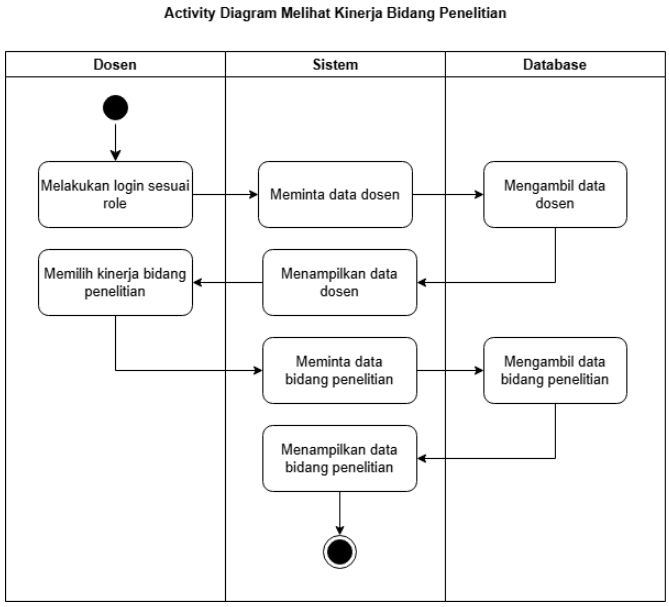


Gambar 12. *Activity Diagram* Melihat Kinerja Bidang Pendidikan

h) *Activity Diagram* Melihat Kinerja Bidang Penelitian (UC-08)

Pada *activity diagram* ini menggambarkan alur aktivitas dosen dalam memantau kinerjanya pada bidang penelitian. Proses diawali saat dosen berhasil masuk ke dalam sistem dan

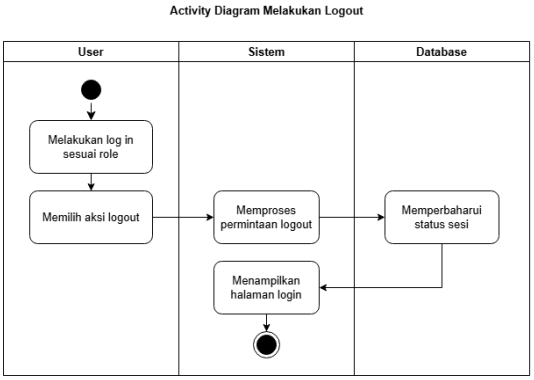
memiliki sesi yang aktif. Setelah itu, dosen dapat memilih fitur untuk melihat data kinerja bidang penelitian yang telah tersedia di dalam basis data. Sistem akan memproses permintaan tersebut dan menampilkan informasi berdasarkan identitas serta data yang berkaitan dengan dosen tersebut. Rancangan *activity diagram* ini dapat dilihat pada Gambar 13.



Gambar 13. *Activity Diagram* Melihat Kinerja Bidang Penelitian

i) *Activity Diagram* Melakukan *Logout* (UC-09)

Pada *activity diagram* ini menggambarkan alur aktivitas pengguna saat melakukan proses keluar dari sistem atau *logout*. Aktivitas dimulai ketika pengguna memilih menu *logout* pada sistem. Selanjutnya, sistem akan memproses permintaan tersebut dengan menghentikan sesi aktif. Setelah sesi dinonaktifkan, sistem secara otomatis mengarahkan pengguna kembali ke halaman *login*. Rancangan *activity diagram* ini dapat dilihat pada Gambar 14.



Gambar 14. *Activity Diagram* Melakukan *Logout*

d. *Sequence Diagram*

Dalam sistem remunerasi dosen, *sequence diagram* digunakan untuk memperjelas alur pertukaran informasi antara pengguna dengan sistem, dimulai dari permintaan hingga penerimaan respon oleh pengguna.

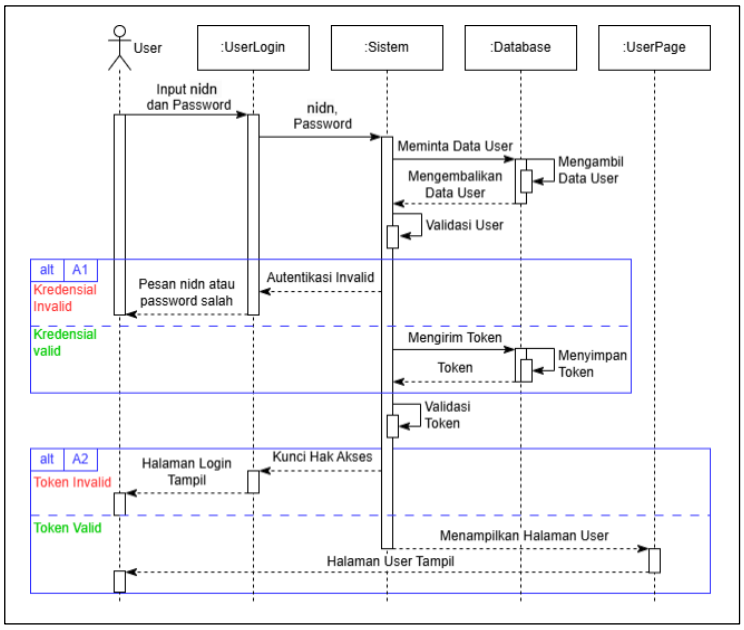
a) *Sequence Diagram* Melakukan *Login* (UC-01)

Pada *sequence diagram* ini menggambarkan tahapan proses autentikasi pengguna dan validasi token pada sistem. Proses dimulai ketika pengguna memasukkan NIDN dan *password* melalui antarmuka *login*. Kredensial tersebut dikirim ke sistem untuk dilakukan pengecekan ke *database*. Setelah *database* mengembalikan informasi yang sesuai, sistem melakukan proses validasi terhadap kredensial tersebut.

Jika hasil validasi menunjukkan kredensial tidak sesuai, sistem mengirimkan pesan kesalahan ke antarmuka *login* sehingga pengguna tetap berada di halaman tersebut. Sebaliknya, apabila kredensial valid, sistem menghasilkan token autentikasi berbasis JWT dan mengirimkannya ke antarmuka

login untuk disimpan di sisi klien sebagai acuan otorisasi pada permintaan berikutnya.

Pada tahap berikutnya, setiap kali pengguna mencoba mengakses halaman yang memerlukan autentikasi, sistem akan memverifikasi token yang dikirimkan. Jika token tidak valid, pengguna diarahkan kembali ke halaman *login*. Namun, apabila token valid, sistem memeriksa hak akses sesuai peran pengguna dan memberikan data yang dibutuhkan untuk ditampilkan pada halaman yang sesuai. Mekanisme ini memastikan bahwa hanya pengguna dengan kredensial sah dan token yang valid yang dapat mengakses. Rancangan *sequence* diagram ini dapat dilihat pada Gambar 15.



Gambar 15. *Sequence Diagram* Melakukan *Login*

b) *Sequence Diagram* Mengelola Akun Pengguna (UC-02)

Pada *sequence diagram* ini menggambarkan alur proses pengguna dalam pengelolaan akun, yang mencakup penambahan akun baru, pembaruan data akun, dan penghapusan akun.

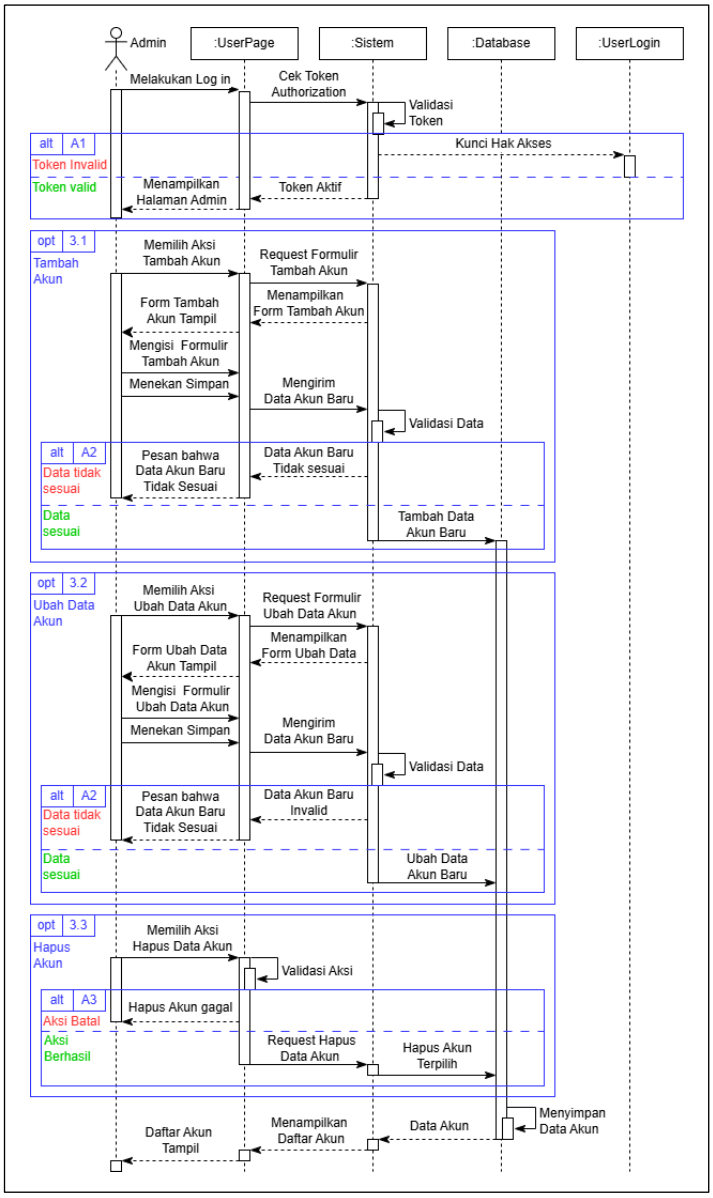
Proses dimulai ketika admin melakukan *login*, kemudian sistem memeriksa token otorisasi yang dikirimkan untuk divalidasi. Apabila token tidak valid, akses ke halaman admin ditolak. Sebaliknya, jika token valid, sistem menampilkan halaman admin yang memuat opsi pengelolaan akun.

Pada proses penambahan akun, admin memilih menu tambah akun, lalu sistem menampilkan formulir *input*. Data yang diisi dikirim ke sistem untuk dilakukan validasi. Jika valid, data disimpan ke *database*, dan daftar akun diperbarui. Jika tidak valid, sistem akan memberikan pesan kesalahan.

Pada proses penambahan akun, admin memilih menu tambah akun, lalu sistem menampilkan formulir *input*. Data yang telah diisi dikirim ke sistem untuk divalidasi. Jika valid, data disimpan ke *database*, dan daftar akun diperbarui. Jika tidak valid, sistem akan memberikan pesan kesalahan.

Pada proses penghapusan akun, admin memilih menu hapus akun. Kemudian, data akun dihapus dari *database* dan daftar akun diperbarui.

Mekanisme ini untuk memastikan seluruh manajemen akun dilakukan secara terkontrol, yang hanya dapat diakses oleh admin dengan token valid, dan melalui proses validasi data sebelum perubahan diterapkan pada database. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 16.



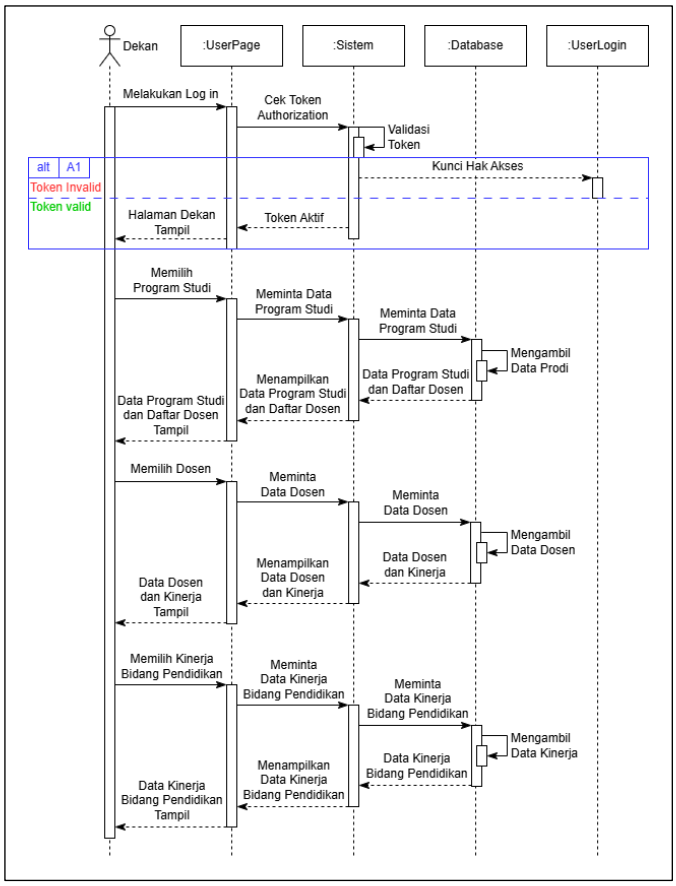
Gambar 16. *Sequence Diagram* Mengelola Akun Pengguna

c) *Sequence Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas (UC-03)

Pada *sequence diagram* ini menggambarkan alur proses data kinerja dosen bidang pendidikan di tingkat fakultas oleh dekan. Proses dimulai ketika dekan melakukan *login* ke dalam sistem. Sistem kemudian memvalidasi token otorisasi untuk menjamin keamanan akses. Apabila token terbukti valid, sistem akan menampilkan halaman utama khusus dekan. Setelah berhasil *login*, dekan dapat memilih program studi

yang diinginkan. Permintaan data program studi beserta daftar dosen terkait dikirimkan ke *database*. Data yang diperoleh dari *database* kemudian ditampilkan pada halaman.

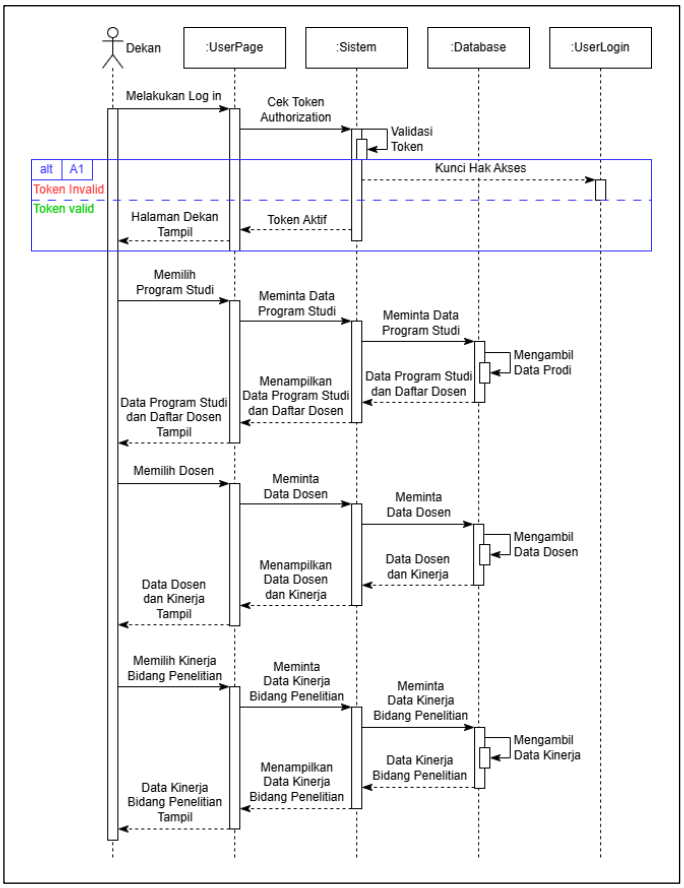
Selanjutnya, Dekan dapat memilih dosen tertentu untuk melihat data rinci dosen tersebut beserta kinerjanya. Sistem akan mengambil data dosen dan data kinerja yang bersangkutan dari *database*, dan kemudian akan menampilkannya. Dekan juga dapat memilih bidang pendidikan tertentu untuk menampilkan data kinerja di bidang tersebut. Sistem akan mengambil data kinerja pada bidang pendidikan yang dipilih dari *database*, kemudian menampilkan informasi tersebut. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 17.



Gambar 17. *Sequence Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Fakultas

d) *Sequence Diagram* Melihat Kinerja Bidang Penelitian Tingkat Fakultas (UC-04)

Pada *sequence diagram* ini menggambarkan alur proses data kinerja dosen bidang penelitian di tingkat fakultas oleh dekan. Alur prosesnya serupa dengan bidang pendidikan tingkat fakultas. Namun, perbedaannya dekan dapat memilih bidang penelitian tertentu untuk menampilkan data kinerja di bidang tersebut. Sistem akan mengambil data kinerja bidang penelitian yang dipilih dari *database*, kemudian menampilkan informasi tersebut. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 18.

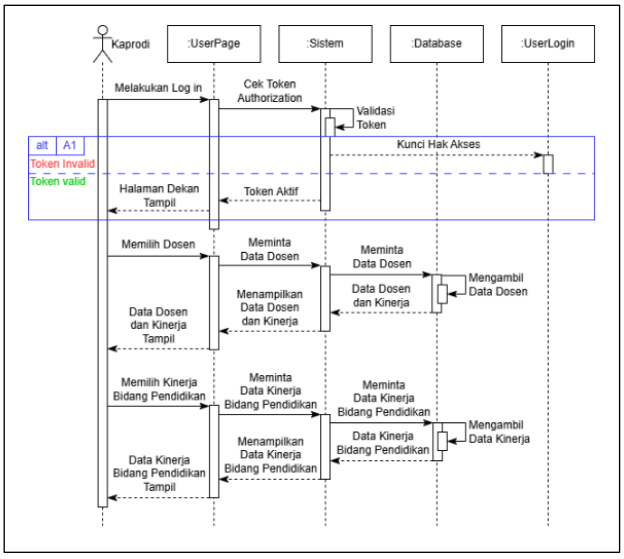


Gambar 18. *Sequence Diagram* Melihat Kinerja Bidang Penelitian Tingkat Fakultas

e) *Sequence Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi (UC-05)

Pada *sequence diagram* ini menggambarkan alur proses data kinerja dosen bidang pendidikan di tingkat program studi oleh kaprodi. Proses dimulai ketika kaprodi melakukan *login* ke dalam sistem. Sistem kemudian memvalidasi token otorisasi untuk menjamin keamanan akses. Apabila token terbukti valid, sistem akan menampilkan halaman utama khusus kaprodi.

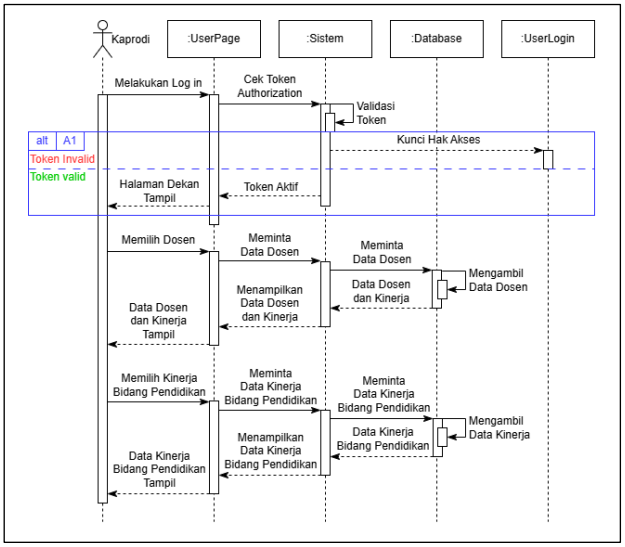
Setelah berhasil *login*, Kaprodi memilih dosen. Sistem mengirimkan permintaan data dosen ke *database*, kemudian *database* mengembalikan data dosen yang mencakup identitas dan riwayat kinerja. Data tersebut kemudian ditampilkan. Selanjutnya, Kaprodi memilih menu kinerja bidang pendidikan. Sistem kembali mengirim permintaan ke *database* untuk mengambil data kinerja bidang pendidikan dosen yang dipilih. Kemudian *database* mengembalikan data tersebut, lalu sistem menampilkannya. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 17.



Gambar 19. *Sequence Diagram* Melihat Kinerja Bidang Pendidikan Tingkat Program Studi

f) *Sequence Diagram* Melihat Kinerja Bidang Penelitian Tingkat Program Studi (UC-06)

Pada *sequence diagram* ini menggambarkan alur proses data kinerja dosen bidang penelitian di tingkat program studi oleh kaprodi. Alur prosesnya serupa dengan bidang pendidikan tingkat program studi. Namun, perbedaanya kaprodi dapat memilih menu kinerja bidang penelitian. Sistem kembali mengirim permintaan ke *database* untuk mengambil data kinerja bidang penelitian dosen yang dipilih. Kemudian *database* mengembalikan data tersebut, lalu sistem menampilkannya. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 18.



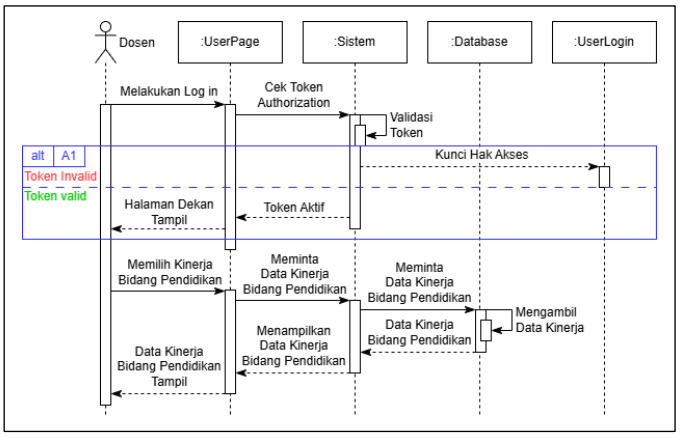
Gambar 20. *Sequence Diagram* Melihat Kinerja Bidang Penelitian Tingkat Program Studi

g) *Sequence Diagram* Melihat Kinerja Bidang Pendidikan (UC-07)

Pada *sequence diagram* ini menggambarkan alur proses ketika dosen mengakses data kinerjanya pada bidang pendidikan. Proses dimulai ketika dosen melakukan *login* ke dalam sistem. Sistem kemudian memvalidasi token otorisasi untuk menjamin

keamanan akses. Apabila token terbukti valid, sistem akan menampilkan halaman utama khusus dosen.

Setelah berhasil *login*, Dosen memilih menu kinerja bidang pendidikan. Sistem mengirimkan permintaan data ke *database* untuk mengambil informasi kinerja bidang pendidikan. Kemudian, *database* memproses permintaan tersebut, mengambil data kinerja yang relevan, dan mengembalikannya ke sistem. Data tersebut kemudian ditampilkan. Rancangan *sequence diagram* dapat dilihat pada Gambar 21.

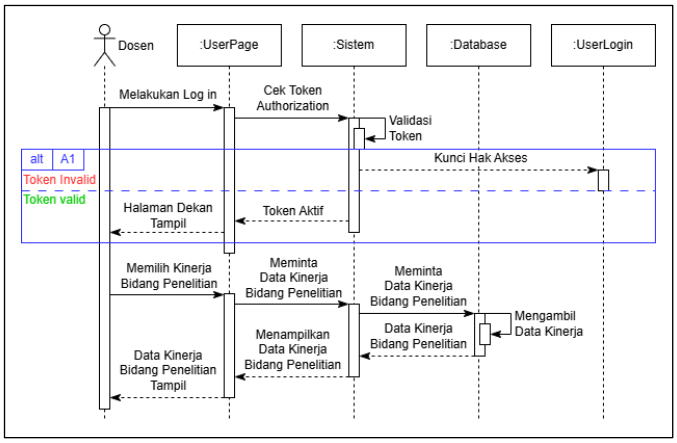


Gambar 21. *Sequence Diagram* Melihat Kinerja Bidang Pendidikan

h) *Sequence Diagram* Melihat Kinerja Bidang Penelitian (UC-08)

Pada *sequence diagram* ini menggambarkan alur proses ketika dosen mengakses data kinerjanya pada bidang penelitian. Alur prosesnya serupa dengan bidang pendidikan. Namun, perbedaanya dosen dapat memilih menu kinerja bidang penelitian. Sistem mengirimkan permintaan data ke *database* untuk mengambil informasi kinerja bidang penelitian. Kemudian, *database* memproses permintaan tersebut, mengambil data kinerja yang relevan, dan mengembalikannya

ke sistem. Data tersebut kemudian ditampilkan. Rancangan *sequence diagram* dapat dilihat pada Gambar 22.

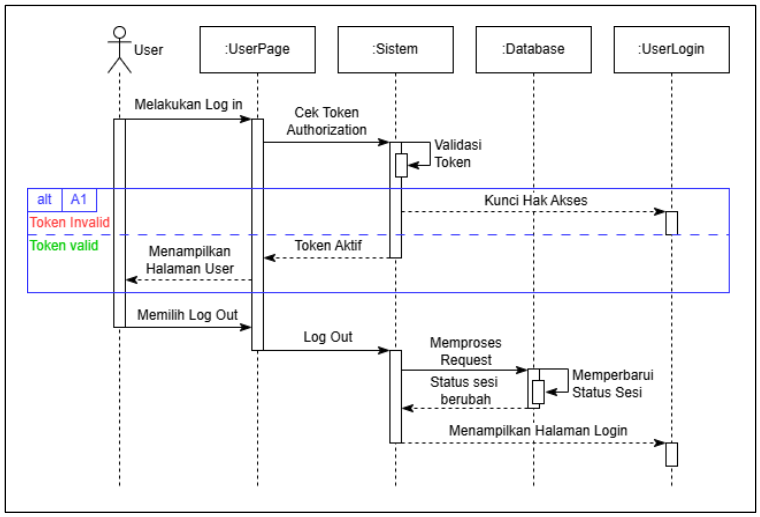


Gambar 22. *Sequence Diagram* Melihat Kinerja Bidang Penelitian

i) *Sequence Diagram* Melakukan *Logout* (UC-09)

Pada *sequence diagram* ini menggambarkan alur proses ketika pengguna mengakhiri sesi melalui fitur *logout*. Proses dimulai ketika pengguna melakukan *login* ke dalam sistem. Sistem kemudian memvalidasi token otorisasi untuk menjamin keamanan akses. Apabila token terbukti valid, sistem akan menampilkan halaman utama khusus pengguna.

Setelah berhasil *login*, pengguna dapat memilih untuk keluar dari sistem (*log out*). Ketika perintah *log out* dipilih, sistem akan memproses permintaan tersebut dan memperbarui status sesi pengguna di *database* menjadi tidak aktif, lalu menampilkan kembali halaman *login*. Rancangan *sequence diagram* ini dapat dilihat pada Gambar 23.



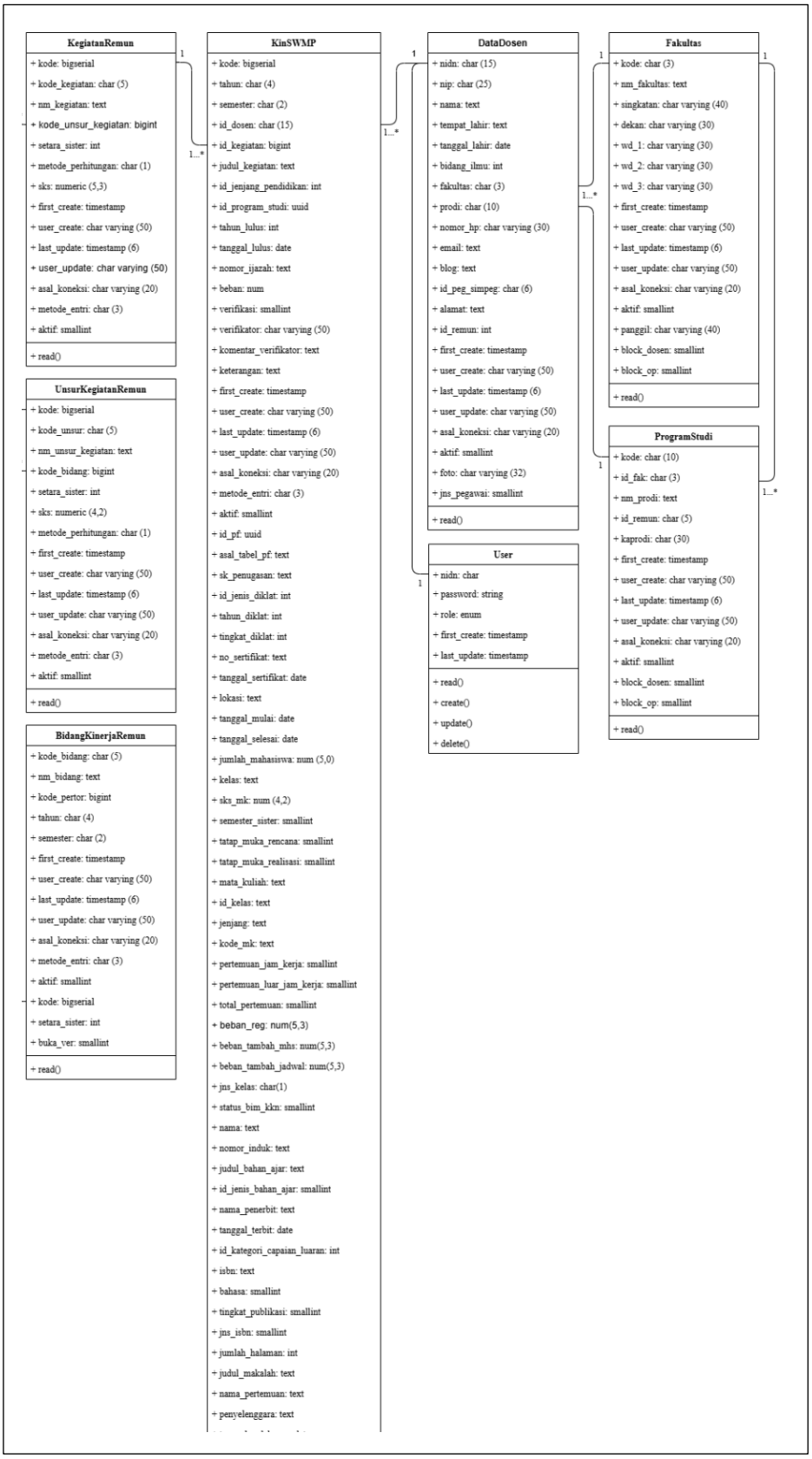
Gambar 23. *Sequence Diagram* Melakukan *Logout*

2. *Design*

Tahap *Design* merupakan proses perancangan difokuskan pada penentuan komponen utama, termasuk *class* diagram dan *endpoint* API. Dengan adanya rancangan yang terdokumentasi dengan baik, proses implementasi di tahap berikutnya dapat berjalan lebih terstruktur.

a. *Class Diagram*

Class diagram pada pengembangan Sistem Remunerasi Dosen Universitas Lampung bidang pendidikan dan penelitian dirancang yang mencakup sejumlah kelas utama, antara lain User, Data Dosen, Fakultas, Program Studi, kinSWMP, Kegiatan Remun, Unsur Kegiatan Remun, dan Bidang Kinerja Remun. Masing-masing kelas memiliki fungsi spesifik yang saling berkaitan dalam membentuk struktur sistem yang terintegrasi. Dengan perancangan ini, sistem mampu mengelola dan memproses data secara lebih efisien. Rancangan *class diagram* dapat dilihat pada Gambar 24.



Gambar 24. *Class Diagram* Sistem Remunerasi Dosen

b. API *Endpoint*

Perancangan API *endpoint* dilakukan untuk mengatur mekanisme komunikasi antara *server* dan *klien* secara terstruktur. Tahapan ini mencakup penetapan *endpoint*, penetuan metode HTTP yang disesuai, serta format data yang digunakan dalam proses *request* dan *response*. Rancangan API *endpoint* pada sistem remunerasi dosen disajikan sebagai berikut.

Tabel 24. Rancangan API *Endpoint* Melakukan *Login*

Method		POST
Path		/api/auth/login
Description		Pengguna masuk dan mendapatkan autentikasi.
Parameter		-
Request	***Header***	-
	Body	nidn, password

Respons	***Success***	satus code: 200 Ok
		message: “Login Successful”
		data: {
		token: string
		}

	Failed	status Code: 401 Unauthorized message: “Invalid NIDN or password”
		status code: 401 Unauthorized message: “NIDN not registered”

Tabel 24 menjelaskan rancangan API *endpoint* yang digunakan dalam proses *login*. Pengguna melakukan permintaan dengan mengirimkan data yang berisi NIDN dan *password* melalui metode POST. Selanjutnya, sistem akan memverifikasi data yang dikirimkan. Jika data sesuai, sistem memberikan *respons* yang berisi token sebagai bukti autentikasi. Token tersebut digunakan untuk mengakses sistem sesuai dengan hak akses pengguna. Proses ini bertujuan untuk memastikan bahwa hanya pengguna yang terverifikasi yang dapat menggunakan layanan dalam sistem.

Tabel 25. Rancangan API *Endpoint* Menampilkan Data Pengguna

Method		GET
Path		/api/users
Description		Admin dapat melihat data pengguna.
Parameter		nidn, role, limit, offset, sort, order
Request	***Header***	authorization: bearer {token}
	Body	-

Respons	***Success***	status code: 200 Ok
		message: “Get All User Successful”
		data: {
		nama: text,
		nidn: char(15),
		role: char(10),
		fakultas: text,
		prodi: text,
		foto: char varying(32)
		}

	Failed	status code: 404 Not Found message: “User Not Found”

Tabel 25 menjelaskan rancangan API *endpoint* yang digunakan untuk menampilkan data pengguna dalam sistem. *Endpoint* ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin. Proses dimulai saat sistem menerima permintaan dari admin untuk menampilkan data seluruh pengguna yang tersimpan di dalam basis data. Permintaan tersebut dapat dilengkapi dengan parameter, seperti nidn, *role, limit*, *offset*, *sort*, dan *order* untuk mempermudah pencarian dan pengurutan data sesuai kebutuhan. Jika data pengguna ditemukan, sistem akan memberikan *respons* berupa daftar pengguna. Sebaliknya, jika data tidak ditemukan, sistem akan memberikan *respons* gagal.

Tabel 26. Rancangan API *Endpoint* Menambah Pengguna

Method		POST
Path		/api/users
Description		Admin dapat menambah pengguna baru.
Parameter		-
Request	***Header***	authorization: bearer {token}
	Body	nidn, password, role
Respons	***Success***	status code: 201 Created

		message: “Add User Successful”
		data: {
		nidn: char (15),
		password: string (128),
		role: char (10)
		}

	Failed	status code 400 Bad Request message: “NIDN was registered”
		status code 400 Bad Request message: “Invalid Data Input”

Tabel 26 menjelaskan rancangan API *endpoint* yang digunakan untuk menambahkan pengguna baru ke dalam sistem. Fungsi ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin. Proses dimulai ketika admin mengisi data pengguna baru, seperti NIDN, *password*, dan, *role*, kemudian mengirimkan data tersebut melalui *request body* ke sistem. Selanjutnya, sistem akan melakukan verifikasi terhadap data yang diterima, termasuk memeriksa apakah NIDN yang dimasukkan sudah terdaftar di dalam basis data. Jika data tersebut valid dan tidak ditemukan duplikasi, sistem akan memproses penambahan data pengguna dan memberikan *respons* dengan status berhasil. Sebaliknya, jika terjadi kesalahan, seperti data sudah terdaftar atau format data tidak sesuai, sistem akan memberikan *respons* gagal yang berisi pesan kesalahan.

Tabel 27. Rancangan API *Endpoint* Mengubah Data Pengguna

Method		PUT
Path		/api/users?nidn={nidn}
Description		Admin dapat mengubah data pengguna.
Parameter		nidn
Request	***Header***	authorization: bearer {token}
	Body	nidn, password, role

Respons	***Success***	status code: 200 Ok
		message: “Update User Successful”
		data: {
		role: char (10),
		password: string (128)
		}

	Failed	status code 400 Bad Request message: “Invalid Data Input”

Tabel 27 menjelaskan rancangan API *endpoint* yang digunakan untuk melakukan perubahan data pengguna pada sistem. Fungsi ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin. Proses perubahan data dimulai dengan pengiriman permintaan melalui *request body*, yang berisi data baru *role* pengguna. Setelah permintaan diterima, sistem akan melakukan proses validasi terhadap data yang dikirimkan untuk memastikan ketidaksesuaian format. Apabila data yang diberikan valid, maka sistem akan memperbarui data pengguna. Selanjutnya, sistem akan memberikan *respons* sukses yang menyatakan bahwa proses pembaruan telah berhasil dilakukan. Namun, jika terjadi kesalahan, seperti format data tidak valid, maka sistem akan memberikan *respons* gagal yang berisi pesan kesalahan.

Tabel 28. Rancangan API *Endppoint* Menghapus Pengguna

Method		DELETE
Path		/api/users?nidn={nidn}
Description		Admin dapat menghapus pengguna.
Parameter		nidn
Request	***Header***	authorization: bearer {token}
	Body	-
Respons	***Success***	status code: 204 No Content message: “User Successfully Removed”
	Failed	status code 404 Not Found message: “User not found”

Tabel 28 menjelaskan rancangan API *endpoint* yang digunakan untuk menghapus data pengguna dari sistem. Fungsi ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin. Proses penghapusan dilakukan dengan mengirimkan permintaan menggunakan metode DELETE yang menyertakan parameter nidn sebagai identitas pengguna yang akan dihapus. Setelah permintaan diterima, sistem akan melakukan verifikasi

terhadap NIDN yang dikirimkan untuk memastikan bahwa data pengguna tersebut terdaftar di dalam basis data. Apabila data ditemukan dan valid, sistem akan menghapus data pengguna dari basis data. Sebaliknya, apabila data tidak ditemukan, sistem akan memberikan *respons* gagal.

Tabel 29. Rancangan API *Endpoint* Menampilkan Daftar Program Studi Tingkat Fakultas

Method		GET
Path		/api/fakultas
Description		Admin dan Dekan dapat melihat data fakultas
Parameter		fakultas, prodi, nidn, limit, offset, sort, order, search
Request	***Header***	authorization: bearer {token}
	Body	-

Respons	***Success***	Status Code: 200 Ok
		Message: "Get All Study Programs Successful"
		Data: {
		kode: char (3),
		nm_fakultas: text,
		singkatan: char varying (40),
		data_prodi: [{
		kode: char (10),
		nm_prodi: text,
		kaprodi: char (30),
		}]
		}

	Failed	Status Code 404 Not Found Message: "No Data Found for the Specified Faculty"

Tabel 29 menjelaskan rancangan API *endpoint* yang digunakan untuk menampilkan daftar program studi berdasarkan fakultas. Fungsi ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin maupun dekan. Permintaan data dilakukan menggunakan metode GET dengan menyertakan parameter seperti fakultas, prodi, NIDN, serta beberapa parameter tambahan seperti *limit*, *offset*, *sor*t, *order*, dan *search* untuk mempermudah pencarian serta pengurutan data. Setelah sistem menerima permintaan, sistem

akan memverifikasi parameter yang diberikan dan mencari data program studi yang sesuai di dalam basis data. Jika data ditemukan, sistem akan memberikan *respons* sukses yang berisi informasi fakultas, termasuk daftar program studi yang berada di bawahnya. Namun, apabila data tidak ditemukan atau format permintaan tidak sesuai, sistem akan mengembalikan *respons* gagal.

Tabel 30. Rancangan API *Endpoint* Menampilkan Daftar Dosen Tingkat Program Studi

Method		GET
Path		/api/programstudi
Description		Admin, Dekan, dan Kaprodi dapat melihat Data Program Studi
Parameter		fakultas, prodi, nidn, limit, offset, sort, order, search
Request	***Header***	authorization: bearer {token}
	Body	-

Respons	***Success***	Status Code: 200 Ok
		Message: "Get All Lecturers Successful"
		Data: {
		kode: char (10),
		nm_prodi: text,
		kaprodi: char (30),
		data_dosen: [{
		nidn: char (15),
		nip: char (25),
		nama: text,
		foto: char varying (32)
		}]
		}

	Failed	Status Code 404 Not Found Message: "No lecturers found for the specified program study"

Tabel 30 menjelaskan rancangan API *endpoint* yang digunakan untuk menampilkan daftar dosen pada tingkat program studi. Fungsi ini hanya dapat diakses oleh pengguna yang memiliki hak akses sebagai admin, dekan, maupun kaprodi. Permintaan data dilakukan menggunakan metode GET dengan menyertakan parameter seperti fakultas, prodi, NIDN, serta beberapa parameter tambahan seperti *limit*, *offset*, *sor*t, *order*, dan *search* untuk

mempermudah pencarian serta pengurutan data. Setelah permintaan diterima, sistem akan melakukan verifikasi parameter yang diberikan dan mencari data dosen yang sesuai di dalam basis data. Jika data ditemukan, sistem akan memberikan *respons* sukses yang berisi informasi terkait program studi dan daftar dosen yang berada di bawah program studi tersebut. Namun, apabila data tidak ditemukan atau format permintaan tidak sesuai, sistem akan mengembalikan *respons* gagal.

Tabel 31. Rancangan API *Endpoint* Melihat Detail Data Dosen

Method		GET
Path		/api/dosen
Description		Menampilkan detail data dosen.
Parameter		fakultas, prodi, nidn, limit, offset, sort, order, search
Request	***Header***	authorization: bearer {token}
	Body	-
Respons	***Success***	Status Code: 200 Ok Message: "Successful" Data: [{ nidn: char (15), nip: char (25), nama: text, tempat_lahir: text, tanggal_lahir: date, bidang_ilmu: int, nm_prodi: text, nm_fakultas: text, nomor_hp: char varying (30), email: text, alamat: text, foto: char varying (32), }]
	Failed	Status Code 404 Not Found Message: "Data for the specified lecturer was not found"

Tabel 31 menjelaskan rancangan API *endpoint* yang digunakan untuk menampilkan detail data dosen. *Endpoint* ini dapat diakses oleh admin, dekan, kaprodi, dan dosen sesuai dengan hak akses yang dimiliki. Dengan API ini dosen dapat melihat data dirinya sendiri. Permintaan dilakukan melalui metode GET dengan

menyertakan parameter, seperti nidn. Setelah permintaan diterima, sistem akan melakukan verifikasi parameter yang diberikan dan mencari data dosen yang sesuai di dalam basis data.

Tabel 32. Rancangan API *Endpoint* Melihat Kinerja Dosen Bidang Pendidikan

Method		GET
Path		/api/pendidikan
Description		Menampilkan data kinerja dosen bidang pendidikan.
Parameter		fakultas, prodi, nidn, tahun, semester, *limit, offset, sort, order, search*.
Request	***Header***	authorization: bearer {token}
	Body	-
Respons	***Success***	Status Code: 200 Ok Message: "Successful" Data: [{ id_dosen: char (15), kode: bigserial, tahun: char (4), semester: char (2) id_kegiatan: bigint, judul_kegiatan: text, id_jenjang_pendidikan: int, tahun_lulus: int, tanggal_lulus: date, nomor_ijazah: text }]
	Failed	Status Code 404 Not Found Message: "Data not Found"

Tabel 32 menjelaskan rancangan API *endpoint* untuk menampilkan data kinerja dosen dalam bidang pendidikan. API ini dapat diakses oleh pengguna yang memiliki hak akses, seperti admin, dekan, kaprodi, dan dosen. Permintaan dilakukan melalui metode GET dengan menyertakan parameter tertentu. Setelah permintaan diterima, sistem akan melakukan proses verifikasi terhadap parameter yang dikirimkan dan melakukan pencarian data pada basis data. Apabila permintaan dinyatakan valid dan data tersedia, sistem akan memberikan *respons* berhasil yang berisi informasi rinci mengenai kinerja dosen dalam bidang pendidikan, sesuai

dengan data Beban Kinerja Dosen (BKD). Namun, jika tidak ditemukan atau format permintaan tidak sesuai, sistem akan memberikan *respon*s gagal.

Tabel 33. Rancangan API *Endpoint* Melihat Kinerja Dosen Bidang Pelaksanaan Pendidikan

Method		GET
Path		/api/pelaksanaan-pendidikan
Description		Menampilkan data kinerja dosen bidang pelaksanaan pendidikan.
Parameter		fakultas, prodi, nidn, tahun, semester, *limit, offset, sort, order, search*.
Request	***Header***	authorization: bearer {token}
	Body	-
Respons	***Success***	Status Code: 200 Ok Message: "Successful" Data: [{ id_dosen: char (15), kode: bigserial, tahun: char (4), semester: char (2), id_kegiatan: bigint, judul_kegiatan: text, id_program_studi: uuid, beban: num, komentar_verifikator: text, keterangan: text, metode_entri: char (3), aktif: smallint, sk_penugasan: text, id_jenis_diklat: int, tahun_diklat: int, tingkat_diklat: int, no_sertifikat: text, tanggal_sertifikat: date, lokasi: text, tanggal_mulai: date, tanggal_selesai: date, jumlah_mahasiswa: num(5,0), kelas: text, sks_mk: num (4,2), semester_sister: smallint, tatap_muka_rencana: smallint, tatap_muka_realisasi: smallint, mata_kuliah: text, id_kelas: text, jenjang: text, kode_mk: text,

		pertemuan_jam_kerja: smallint, pertemuan_luar_jam_kerja: smallint, total_pertemuan: smallint, beban_reg: num (5,3), jns_kelas: char (1), status_bim_kkn: smallint, nama: text. nomor_induk: text, judul_bahan_ajar: text, id_jenis_bahan_ajar: smallint }]
	Failed	Status Code 404 Not Found Message: "Data not Found"

Tabel 33 menjelaskan rancangan API *endpoint* untuk menampilkan data kinerja dosen dalam bidang pelaksanaan pendidikan. API ini dapat diakses oleh pengguna yang memiliki hak akses, seperti admin, dekan, kaprodi, dan dosen. Permintaan dilakukan melalui metode GET dengan menyertakan parameter tertentu. Setelah permintaan diterima, sistem akan melakukan proses verifikasi terhadap parameter yang dikirimkan dan melakukan pencarian data pada basis data. Apabila permintaan dinyatakan valid dan data tersedia, sistem akan memberikan *respons* berhasil yang berisi informasi rinci mengenai kinerja dosen dalam bidang pelaksanaan pendidikan, sesuai dengan data Beban Kinerja Dosen (BKD). Namun, jika tidak ditemukan atau format permintaan tidak sesuai, sistem akan memberikan *respon*s gagal.

Tabel 34. Rancangan API *Endpoint* Melihat Kinerja Dosen Bidang Penelitian

Method		GET
Path		/api/penelitian
Description		Menampilkan data kinerja dosen bidang penelitian.
Parameter		fakultas, prodi, nidn, tahun, semester, *limit, offset, sort, order, search*.
Request	***Header***	authorization: bearer {token}
	Body	-
Respons	***Success***	Status Code: 200 Ok Message: "Successful"

		Data: [{ id_dosen: char (15), kode: bigserial, tahun: char (4), semester: char (2), id_kegiatan: bigint, judul_kegiatan: text, nama_penerbit: text, tanggal_terbit: date, id_kategori_capaian_luaran: int, isbn: text, bahasa: smallint, tingkat_publikasi: smallint, jns_isbn: smallint jumlah_halaman: int, judul_makalah: text }]
	Failed	Status Code 404 Not Found Message: "Data not found"

Tabel 34 menjelaskan rancangan API *endpoint* untuk menampilkan data kinerja dosen dalam bidang penelitian. API ini dapat diakses oleh pengguna yang memiliki hak akses, seperti admin, dekan, kaprodi, dan dosen. Permintaan dilakukan melalui metode GET dengan menyertakan parameter tertentu. Setelah permintaan diterima, sistem akan melakukan proses verifikasi terhadap parameter yang dikirimkan dan melakukan pencarian data pada basis data. Apabila permintaan dinyatakan valid dan data tersedia, sistem akan memberikan *respons* berhasil yang berisi informasi rinci mengenai kinerja dosen dalam bidang penelitian, sesuai dengan data Beban Kinerja Dosen (BKD). Namun, jika tidak ditemukan atau format permintaan tidak sesuai, sistem akan memberikan *respon*s gagal.

3. *Develop*

Tahap *Develop* merupakan tahap implementasi dari proses API-*First Development*, di mana pengembangan sistem dilakukan berdasarkan rancangan yang telah ditetapkan sebelumnya. Pada tahap ini, API dikembangkan sebagai komponen utama yang mengacu pada

spesifikasi *Open*API. Pengembangan sistem dilakukan menggunakan Node.js sebagai *platform* utama, karena mampu mendukung proses implementasi yang cepat dan efisien. Untuk menjaga keamanan akses terhadap layanan API, diterapkan mekanisme autentikasi menggunakan JSON *Web* Token (JWT). Penerapan autentikasi ini bertujuan untuk memastikan bahwa hanya pengguna yang memiliki otorisasi yang dapat mengakses data dan fungsi yang disediakan oleh sistem.

a. API *contract* berisi Spesifikasi API menggunakan *Open*API

Tahap awal dalam pengembangan API adalah menyusun API *contract* yang berisi spesifikasi API. Dokumen ini disusun menggunakan *Open*API sebagai standar untuk mendefinisikan cara kerja API secara terstruktur dan terdokumentasi. Dalam penelitian ini, spesifikasi API ditulis menggunakan format JSON dengan standar *Open*API versi 3. Setiap *endpoint* dijelaskan secara rinci, mencakup metode HTTP yang digunakan, parameter yang dibutuhkan, struktur data pada respons, serta mekanisme autentikasi yang digunakan. Melalui spesifikasi ini, seluruh pihak yang terlibat dalam pengembangan dapat memahami dengan jelas bagaimana API diakses dan direspons oleh sistem.

b. Implementasi menggunakan NodeJs

Setelah penyusunan API *contract* selesai dilakukan, tahap selanjutnya adalah mengimplementasikan setiap *endpoint* ke dalam kode program menggunakan *framework* Node.js. Pada proses ini, spesifikasi yang telah didefinisikan dalam dokumen *Open*API v3 dijadikan acuan utama dalam pembuatan fungsi-fungsi yang merepresentasikan *endpoint t*ersebut. Node.js berfungsi untuk menangani proses input-output secara *non-blok* (*non-blocking* I/O), yang memungkinkan sistem merespons banyak permintaan secara bersama dengan performa yang baik. Implementasi

dilakukan dengan menyesuaikan struktur data, parameter, serta metode HTTP yang telah ditentukan dalam spesifikasi, sehingga API berjalan sesuai dengan rancangan awal.

c. Mengimplementasikan JWT

Untuk menjaga keamanan dalam mengakses API, digunakan mekanisme autentikasi berbasis JSON *Web* Token (JWT). Pada tahap ini, menggunakan *library* jsonwebtoken yang berfungsi untuk menghasilkan, memverifikasi, dan mengelola token autentikasi. Token ini diberikan kepada pengguna yang berhasil melakukan proses *login*, dan diberikan pada setiap permintaan yang membutuhkan otorisasi. Melalui implementasi JWT, sistem dapat memastikan bahwa hanya pengguna yang memiliki hak akses yang sah yang dapat mengakses data atau layanan tertentu.

4. *Testing* (Pengujian API)

Tahap pengujian merupakan proses verifikasi untuk memastikan bahwa API yang dikembangkan berfungsi sesuai dengan spesifikasi yang telah ditetapkan. Pengujian dilakukan menggunakan beberapa alat bantu, yaitu Postman untuk pengujian fungsional, Apache JMeter untuk pengujian performa, dan OWASP ZAP untuk pengujian keamanan.

a. Pengujian Fungsional

Pengujian fungsional bertujuan memverifikasi bahwa setiap endpoint API memberikan respons sesuai dengan permintaan yang dikirimkan. Validasi dilakukan terhadap struktur data yang dihasilkan, kode status HTTP, dan logika proses yang dijalankan. Tujuan utamanya adalah memastikan API memenuhi kebutuhan fungsional.

Pengujian ini dilakukan menggunakan Postman dengan metode HTTP yang relevan, seperti GET, POST, PUT, dan DELETE. Setiap permintaan dilengkapi headers yang diperlukan, termasuk *Content-Type*: application/json dan *Authorization* menggunakan JSON *Web* Token (JWT), serta parameter atau *body request* sesuai skenario. Pengujian ini mencakup:

1. Skenario positif

   Mengirimkan data valid untuk memastikan API merespons sukses dengan kode status seperti 200 OK atau 201 *Created*.
2. Skenario negatif

   Mengirimkan data tidak valid atau token yang salah untuk memastikan API menolak permintaan dengan kode status yang sesuai, seperti 400 *Bad Request*, 401 *Unauthorized*, 403 *Forbidde*n, atau 404 *Not Found*.

Tabel 35. Skenario Pengujian Fungsional

Endpoint	Scenario ID	Scenario Test
/api/auth/login	TS.01	Uji autentikasi pada *endpoint login* dari kredensial yang valid/invalid.
/api/fakultas	TS.02	Uji *endpoint* GET Fakultas dari otorisasi peran (Admin, Dekan) dan parameter yang valid/invalid.
/api/programstudi	TS.03	Uji *endpoint* GET Program Studi dari otorisasi peran (Admin, Dekan, Kaprodi) dan parameter yang valid/invalid.
/api/dosen	TS.04	Uji *endpoint* GET Dosen dari otorisasi peran (Admin, Dekan, Kaprodi, Dosen) dan parameter yang valid/invalid.
/api/pendidikan	TS.05	Uji *endpoint* GET Kinerja Bidang Pendidikan dari otorisasi peran (Admin, Dekan, Kaprodi, Dosen) dan parameter yang valid/invalid.

Endpoint	Scenario ID	Scenario Test
/api/pelaksanaan-pendidikan	TS.06	Uji *endpoint* GET Kinerja Bidang Pelaksanaan Pendidikan dari otorisasi peran (Admin, Dekan, Kaprodi, Dosen) dan parameter yang valid/invalid.
/api/penelitian	TS.07	Uji *endpoint* GET Kinerja Bidang Penelitian dari otorisasi peran (Admin, Dekan, Kaprodi, Dosen) dan parameter yang valid/invalid.
/api/users	TS.08	Uji endpoint GET POST PUT DELETE Pengguna dari otorisasi peran (Admin) dan parameter yang valid/invalid.
Akses Otorisasi Seluruh *Endpoint*	TS.09	Uji penanganan otorisasi pada semua akses *endpoint* dari token yang invalid.

b. Pengujian Performa

Pengujian performa bertujuan mengevaluasi kemampuan API dalam memproses permintaan di bawah beban kerja tertentu. Pengujian dilakukan menggunakan Apache JMeter dengan skenario *load testing* dan *stress testing* untuk mengukur kinerja sistem pada kondisi normal maupun beban tinggi.

Pelaksanaan pengujian meliputi simulasi sejumlah pengguna yang mengakses *endpoint* secara bersamaan, kemudian memantau metrik utama seperti waktu respons rata-rata, nilai min dan max, jumlah permintaan yang dapat diproses per detik (*throughput*), dan persentase kegagalan permintaan (*error rate*). Hasil pengujian dianalisis untuk menentukan apakah API memiliki performa yang stabil, cepat, dan andal dalam mendukung kebutuhan operasional.

c. Pengujian Keamanan

Pengujian keamanan dengan OWASP ZAP dilakukan untuk mengidentifikasi kerentanan pada API, seperti kelemahan

autentikasi dan risiko akses tidak sah. Langkah ini bertujuan memastikan API memiliki tingkat keamanan yang memadai sehingga dapat melindungi data dan mencegah penyalahgunaan.

5. *Deploy*

Deploy merupakan tahap akhir dalam pengembangan sistem yang berfokus pada penerapan API ke dalam lingkungan produksi. Tujuanya utama dari tahap ini adalah untuk memastikan bahwa layanan yang telah dibangun dapat diakses dan berfungsi dengan baik. Selain implementasi teknis, tahap ini juga mencakup penyusunan dokumentasi API untuk mendukung proses integrasi dan penggunaan yang lebih mudah oleh pengembang lain.

a. *Deploy* API

Proses penerapan dimulai dengan pengujian dan implementasi API pada lokal yang terhubung dengan basis data yaitu PostgreSQL. Langkah ini dilakukan untuk memastikan bahwa setiap *endpoint* API dapat berjalan dengan baik dan sesuai dengan spesifikasi yang telah dirancang. Setelah dipastikan berfungsi, API kemudian dapat diterapkan pada server produksi.

b. API *Documentation*

Setelah proses penerapan selesai, langkah selanjutnya adalah menyusun dokumentasi API secara lengkap dan sistematis. Dokumentasi ini mencakup penjelasan rinci mengenai setiap *endpoint*, metode HTTP yang digunakan, parameter yang diperlukan, serta mekanisme autentikasi yang diterapkan.

V. SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan hasil pengembangan yang telah dilakukan, maka dapat disimpulkan sebagai berikut:

1. RESTful API berhasil dikembangkan untuk mengakses dan menyajikan data kinerja dosen pada bidang pendidikan dan penelitian yang bersumber dari Sistem Remunerasi Dosen Universitas Lampung secara terstruktur. Data yang digunakan dalam penelitian ini secara khusus berasal dari bagian beban lebih pada sistem tersebut.
2. Autentikasi dan otorisasi menggunakan JSON *Web* Token (JWT) telah diterapkan untuk memastikan keamanan akses serta membatasi hak pengguna sesuai peran, seperti admin, dekan, kaprodi, dan dosen.
3. Hasil pengujian menunjukkan bahwa API berfungsi sesuai dengan kebutuhan, baik dari sisi fungsionalitas maupun keamanan, serta mampu memberikan respons yang konsisten pada setiap skenario uji.
4. API yang dikembangkan memiliki potensi untuk diintegrasikan dengan sistem informasi lain, sehingga dapat meningkatkan pemanfaatan data kinerja dosen dalam proses akreditasi, evaluasi kinerja, serta pelaporan institusional di Universitas Lampung.

5.2 Saran

Berdasarkan hasil dari penelitian yang telah dilakukan, terdapat beberapa saran yang dapat dijadikan acuan untuk pengembangan lebih lanjut, yaitu sebagai berikut:

1. Mengoptimalkan pemanfaatan data pengguna dengan mengintegrasikan sistem ke dalam sumber data terpusat milik universitas untuk mendukung penerapan autentikasi tunggal *Single Sign-On* (SSO) secara terpusat dan efisien.
2. Menyempurnakan mekanisme penggantian *password* agar pengguna dapat melakukan reset secara mandiri saat lupa *password* tanpa memerlukan bantuan administrator.
3. Meningkatkan penggunaan sistem *cache* yang lebih skalabel dan andal untuk menjaga performa aplikasi, khususnya pada skala penggunaan yang lebih luas.

DAFTAR PUSTAKA

Abdillah, R., Hermawan, R., Hermawansyah, W., Agustin, D. P., Adkha, I., & Alam, N. (2025). Analisis dan Pengujian Perangkat Lunak Sistem Informasi Pembayaran Sekolah dengan Metode Pengukuran Kualitas SQuaRE. *Jurnal Penelitian Rumpun Ilmu Teknik*, *4*(1), 208–217. https://doi.org/10.55606/juprit.v4i1.4834

Adam, S. I., Moedjahedy, J. H., & Maramis, J. (2020a). RESTful Web Service Implementation on Unklab Information System Using JSON Web Token (JWT). *2020 2nd International Conference on Cybernetics and Intelligent System, ICORIS 2020*. https://doi.org/10.1109/ICORIS50180.2020.9320801

Adam, S. I., Moedjahedy, J. H., & Maramis, J. (2020b). RESTful Web Service Implementation on Unklab Information System Using JSON Web Token (JWT). *Jurnal IEEE (Institute of Electrical and Electronics Engineers)*. https://doi.org/10.1109/ICORIS50180.2020.9320801

Aji, T. B. A., Ajie, H., & Nugraheni, M. (2022). Pengembangan Web Service Aplikasi Manajemen Aset UPT TIK Universitas Negeri Jakarta. *Jurnal Pintar*, *6*(2), 69–75.

Alfarizi, M. R. sirfatullah, Al-farish, M. Z., Taufiqurrahman, M., Ardiansah, G., Elgar, M., & Encep, M. (2023). Implementasi Sistem Informasi Akademik untuk Meningkatkan Efisiensi dan Kualitas Pendidikan. *Karimah Tauhid*, *2*(1), 46–50.

Amalia, N., Penelitian, S., & Djuanda, P. U. (2024). *Tridharma Perguruan Tinggi untuk Membangun Akademik dan Masyarakat Berpradaban*. *3*(4), 4654–4663.

Ariyadi, T., Fadli, H., Akbar, T., & Prihandoko, M. B. (2025). *Implementasi OWASP untuk Analisis Kerentanan dan Keamanan pada Sistem Informasi Akademik Terintegrasi Universitas Bina Darma*. *4*(1), 1–7. https://doi.org/10.55123

Astowo, U. B., & Sujarwo, A. (2023). Penerapan JSON Web Token sebagai Strategi Pengamanan Data pada Aplikasi MultiMasjid. *Journal Of Social Science Research*, *3*(6), 5279–5292.

Cahyono, S. A. B., Sucipto, Firliana, R., Muzzaki, M. N., Wardani, A. S., Khalid, M. I., Gamas, A. W. M., & Setiawan, H. (2022). Rancangan Pembuatan Api Website Data Tanaman Obat Dan Langka Kabupaten Kediri. *Jurnal Bulletin of Information Technology (BIT)*, *3*(4), 255–260. https://doi.org/10.47065/bit.v3i1

Chandrachood, A. (2021). Api First Development: A Modern Approach to Building Integrated Software Systems. *International Journal of Science and Research (IJSR)*, *10*(10), 1627–1629. https://doi.org/10.21275/sr24608141605

Chang, X., Dou, W., Gao, Y., Wang, J., Wei, J., & Huang, T. (2019). Detecting Atomicity Violations for Event-Driven Node.js Applications. *International Conference on Software Engineering*, 631–642. https://doi.org/10.1109/ICSE.2019.00073

Choirudin, R., & Adil, A. (2019). Implementasi REST API Web Service dalam Membangun Aplikasi Multiplatform untuk Usaha Jasa. *Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, *18*(2), 284–293. https://doi.org/10.30812/matrik.v18i2.407

Darmawan, D., Iriandha, D., Indrianto, D., Sigita, D. S., & Cahyani, D. (2021). Hubungan Remunerasi, Retensi dan Kinerja Karyawan. *Journal of Trends Economics and Accounting Research*, *1*(4), 129–133.

Dias, R. S., & Muhallim, M. (2022). Sistem Informasi Penjualan Berbagai Macam Produk Berbasis Android di Toko De Ari Palopo. *Indonesian Journal Of Education And Humanity*, *2*(1), 34–50.

Direktur Jenderal Pendidikan Tinggi Kementerian Pendidikan dan Kebudayaan. (2021). *Pedoman Operasional Beban Kerja Dosen*.

Dzikria, I., Ardan, M. M., & Ripando, R. N. (2023). Sistem Remunerasi Berbasis Kinerja Menggunakan Metode Full Time Equivalent dan Simple Additive Weighting. *Seminar Nasional Teknoka*, *8*, 123–131.

Edy, Ferdiansyah, Pramusinto, W., & Waluyo, S. (2017). Pengamanan Restful API menggunakan JWT untuk Aplikasi Sales Order. *Jurnal Rekayasa Sistem dan Teknologi Informasi*, *1*(3), 106–112.

Farhandika, R., Sabariah, M. K., & Adrian, M. (2024). Penerapan Arsitektur REST API pada Aplikasi Backend Manajemen Informasi Fakultas Industri Kreatif (MI-FIK) Universitas Telkom. *Jurnal Penelitian Informatika*, *2*(1), 40–51. https://doi.org/10.25124/logic.v2i1.7530

Hafsari, R., Aribe, E., & Maulana, N. (2023). Perancangan Sistem Informasi Manajemen Inventori dan Penjualan Pada Perusahaan PT. Inhutani V. *Jurnal PROSISKO*, *10*(2), 109–116.

Hamidah, I., Haromain, I., & Drehem, I. M. (2025). Evaluasi Pengujian Kinerja Menggunakan Jmeter untuk Menunjang Stabilitas Aplikasi Layanan Perbankan pada PT Bank Rakyat Indonesia TBK. *Journal of Digital Business and Technology Innovation (DBESTI)*, *2*(1), 114–126. https://journal.nurulfikri.ac.id/index.php/DBESTI

Hasanuddin, Asgar, H., & Hartono, B. (2022a). Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan. *4*, *4*(1), 8–14.

Hasanuddin, Asgar, H., & Hartono, B. (2022b). Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan. *Jurnal Informatika Teknologi dan Sains*, *4*(1), 8–14.

Hasanuddin, Asgar, H., & Hartono, B. (2022c). Rancangan Bangun REST API Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan. *JINTEKS (Jurnal Informatika Teknologi dan Sains)*, *4*(1), 8–14.

Hendayun, M., Ginanjar, A., & Ihsan, Y. (2023). Analysis of Application Performance Testing Using Load Testing and Stress Testing Methods in API Service. *Jurnal Sisfotek Global*, *13*(1), 28–34. https://doi.org/10.38101/sisfotek.v13i1.2656

Indrianto. (2023). Performance Testing On Web Information System Using Apache JMeter and Blazemeter. *Jurnal Ilmiah Ilmu Terapan Universitas Jambi*, *7*(2), 138–149. https://doi.org/10.22437/jiituj.v7i2.28440

Ismail, A., Ananta, A. Y., Arief, S. N., & Hamdana, E. N. (2023). Performance Testing Sistem Ujian Online Menggunakan Jmeter pada Lingkungan Virtual. *Jurnal Informatika Polinema*, *9*(2), 159–164.

Junardi, W., Septiani, A. P., Amaliah, A., Bachtiar, A., Mahendra, J. I., & Muttaqin, M. I. (2020). Sistem Informasi Desa Siaga Pangan Menghadapi Covid19 berbasis Web Service. *Jurnal Sistem Cerdas*, *03*(02), 231–240.

Khairina, & Afrizai. (2024). Sistem Remunerasi. *Jurnal Pendidikan Tambusai*, *8*(1), 3532–3540.

Universitas Lampung. (2025). *Penetapan Pedoman Rubrik Remunerasi Universitas Lampung*. http://unila.ac.id

Londjo, M. F. (2021). Implementasi White Box Testing Dengan Teknik Basis Path Pada Pengujian Form Login. *Jurnal Siliwangi*, *7*(2), 35–40.

Malius, H., & Dani, A. A. H. (2021). Sistem Informasi Sekolah Berbasis Web Pada Sekolah Dasar Negri (SDN) 109 Seriti. *Indonesian Journal Of Education And Humanity*, *1*(3), 156–168.

Mu'min, M. A., Tristanti, N., & Fanani, G. P. I. (2024). Analisis dan Pengujian Kerentanan Website Menggunakan OWASP ZAP. *Jurnal Riset Sistem dan Teknologi Informasi (RESTIA*, *3*(1), 36–50.

Narulita, S., Nugroho, A., & Abdillah, M. Z. (2024a). Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS). *Bridge : Jurnal publikasi Sistem Informasi dan Telekomunikasi*, *2*(3), 244–256. https://doi.org/10.62951/bridge.v2i3.174

Narulita, S., Nugroho, A., & Abdillah, M. Z. (2024b). Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS). *Bridge : Jurnal publikasi Sistem Informasi dan Telekomunikasi*, *2*(3), 244–256. https://doi.org/10.62951/bridge.v2i3.174

Nashikhuddin, A. Y., Karaman, J., & Litanianda, Y. (2023). Implementasi API RESTful Dengan JSON Web Token Pada Aplikasi E-commerce Thrifty Shop Untuk Otentikasi dan Otorisasi Pengguna. *METHOMIKA Jurnal Manajemen Informatika dan Komputerisasi Akuntansi*, *7*(2), 239–246. https://doi.org/10.46880/jmika.Vol7No2.pp239-246

Nugraha, M., & Rosmeida, M. (2021). Perancangan Sistem Informasi Beban Kerja Dosen Berbasis Web dengan UML. *Jurnal Algoritma*, *19*(1), 141–150. http://jurnal.itg.ac.id/

Nurhayati, S. T., & Nasution, M. I. P. (2023). Database Management System Pada Perusahaan. *Jurnal Akuntansi Keuangan dan Bisnis*, *1*(2), 62–64. https://jurnal.ittc.web.id/index.php/jakbs/index

Okta. (2024). *Introduction to JSON Web Tokens*. Diakses pada 22 Januari 2025. https://jwt.io/introduction

Pariddudin, A., & Fadhli, I. (2022). Penerapan Web Service untuk Meningkatkan Performa Kecepatan Data pada Sistem Tes Potensi Akademik. *Jurnal Ilmiah Teknologi Informasi & Sains*, *12*(1), 85–94. https://doi.org/10.36350/jbs.v12i1

Perdana, M. A. K. (2018). Pengembangan Rest Api Layanan Penyimpanan Menggunakan Metode Rapid Application Development (Studi Kasus: PT. XYZ). *Jurnal Nasional Informatika dan Teknologi Jaringan*, *3*(1), 100–104.

Perkasa, M. I., & Setiawan, E. B. (2018). Pembangunan Web Service Data Masyarakat Menggunakan REST API dengan Access Token. *ULTIMA Computing*, *X*(1).

Postman. (2025). *Platform API Postman*. Diakses pada 15 Januari 2025.

Praba, A. D., & Safitri, M. (2020). Studi Perbandingan Performasi Antara MySQL dan PostgreSQL. *Jurnal Khatulistiwa Informatika*, *VIII*(2), 88–93. https://www.adminer.org/.

Pranoto, S., Sutiono, S., & Nasution, D. D. (2024). Penerapan UML Dalam Perancangan Sistem Informasi Pelaporan Dan Evaluasi Pembangunan Pada Bagian Administrasi Pembangunan Sekretariat Daerah Kota Tebing Tinggi. *Jurnal Ekonomi dan Bisnis*, *2*(2), 384–401.

Prio, A., Lathifah, A., Indriyanah, A., & Penulis, K. (2022). Literature Review Sistem Informasi Manajemen: Software, Database dan Brainware. *Jurnal Ekonomi Manajemen Sistem Informasi*, *3*(4), 442–451. https://doi.org/10.31933/jemsi.v3i4

Putra, Ubaidi, U., Hamzah, A., Pramadi, W. A., & Nuraini, A. (2024). Systematic Literature Review: Security Gap Detection On Websites Using Owasp Zap. *Brilliance: Research of Artificial Intelligence*, *4*(1), 348–355. https://doi.org/10.47709/brilliance.v4i1.4227

Putra, I. G. A. H. A., Bagia, I. W., & Telagawati, N. L. W. S. (2019). Dampak Remunerasi Terhadap Peningkatan Kinerja Karyawan. *Jurnal Universitas Pendidikan Ganesha*, *7*, 160–171.

Rachmawaty, W., & Pandoyo. (2020). Pengaruh Remunerasi Terhadap Kinerja Pegawai Pada Biro Umum Sekretariat Jenderal Kementerian Perhubungan. *Jurnal Ekonomi, Manajemen, Bisnis dan Sosial*, *1*(1), 1–10.

Ramdany, S. W., Kaidar, S. A., Aguchino, B., Putri, C. A. A., & Anggie, R. (2024). Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web. *Journal of Industrial and Engineering System*, *5*(1), 30–41.

Regita, N. (2023). *Pengertian Tri Dharma Perguruan Tinggi dan Cara Penerapannya*. Diakses pada 22 Januari 2025. https://suteki.co.id/pengertian-tri-dharma-perguruan-tinggi-dan-cara-penerapannya/

Rezeki, S. G., & Nasution, M. I. P. (2023). Peranan Penggunaan Basis Data dalam Sistem Informasi Manajemen. *IJM: Indonesian Journal of Multidisciplinary*, *1*(4), 1243–1251. https://journal.csspublishing/index.php/ijm

Rivando, I. I. (2023). Implementasi Representatif State Transition Application Programming Interface (REST API) Pada Aplikasi Tip.in Berbasis Android. *Jurnal Teknologi Pintar*, *3*(1), 1–16.

Sari, R. A., Dirayati, F., & HS, T. M. F. (2024). Penilaian Kinerja Pengajaran Dosen Menggunakan Hasil Data Pengisian yang Terkomputerisasi. *Jurnal Sistem Informasi & Manajemen Basis Data (SIMADA)*, *7*(1), 26–36.

Sauda, S., & Barokah, M. (2022). Penerapan NodeJS dan PostgreSQL Sebagai Backend Pada Aplikasi Ecommerce Localla. *Jurnal Infotech*, *8*(2), 101–105. https://doi.org/10.31949/infotech.v8I2.2944

Septama, H. D., Komarudin, M., Wintoro, P. B., Pratama, M., Yulianti, T., & Sulistiono, W. E. (2022). Implementation of One Data-based Lecturer Profile Information System for Key Performance Indicator Monitoring. *2022 7th International Conference on Informatics and Computing, ICIC 2022*. https://doi.org/10.1109/ICIC56845.2022.10006928

Setiawan, A., & Purnamasari, A. I. (2017). Implementasi JSON Web Token Berbasis Algoritma SHA-512 untuk Otentikasi Aplikasi BatikKita. *Jurnal Rekayasa Sistem dan Teknologi Informasi*, *1*(3), 1036–1045.

Sianturi, R. A., Sinaga, A. M., Pratama, Y., Simatupang, H., Panjaitan, J., & Sihotang, S. (2021). Perancangan Pengujian Fungsional dan Non Fungsional Aplikasi Siappara di Kabupaten Humbang Hasundutan. *Jurnal Komputer dan Informatika*, *9*(2), 133–141. https://doi.org/10.35508/jicon.v9i2.4706

Stiawan, H., Muzzaki, M. N., Wardani, A. S., Firliana, R., khalid, M. I., Gamas, A. W. M., & Busro, S. A. (2022). Model Visualisasi Informasi Dashboard Pada Pemetaan Tanaman Obat dan Langka Kabupaten Kediri Menggunakan Microsoft Power BI. *Jurnal Informatika Teknologi dan Sains*, *4*(4), 366–371.

Sudarso, A. (2022). Pemanfaatan Basis Data, Perangkat Lunak dan Mesin Industri Dalam Meningkatkan Produksi Perusahaan (Literature Review Executive Support Sistem (ESS) For Business). *Jurnal Manajemen Pendidikan dan Ilmu Sosial*, *3*(1), 1–14. https://doi.org/10.38035/jmpis.v3i1

Tanaem, P. F., Manongga, D., & Iriani, A. (2016). RESTFul Web Service untuk Sistem Pencatatan Transaksi Studi Kasus PT. XYZ. *Jurnal Teknik Informatika dan Sistem Informasi*, *2*(1), 1–10.

Widyastuti, R., Widiyastuti, A. A., & Ramadhan, D. W. (2022). Penerapan Sistem Informasi Akademik di SMK Yaspen Jakarta. *Jurnal Prosisko*, *9*(2), 9–24.