BAB II

KAJIAN PUSTAKA

A. Tinjauan Pustaka

Penilaian kinerja dosen yang bertumpu pada publikasi jurnal penting karena output publikasi masih sering dijadikan ukuran produktivitas ilmiah, tingkat keterlihatan riset, serta daya saing institusi. Dalam kerangka responsible research assessment, indikator kuantitatif seperti jumlah publikasi dan sitasi dapat membantu memetakan kontribusi keilmuan, namun penggunaannya perlu dilakukan secara bertanggung jawab dan kontekstual, serta dilengkapi penilaian kualitatif agar tidak menghasilkan kesimpulan evaluatif yang menyesatkan (Himanen et al., 2024; Schöpfel). Selain itu, interpretasi indikator juga harus cermat karena berbagai model/indikator termasuk yang memanfaatkan analitik lanjutan memiliki tantangan terkait reliabilitas, transparansi, dan potensi bias yang dapat memengaruhi pembacaan hasil (Zhang, 2023). Di sisi lain, performa publikasi bersifat fluktuatif karena dipengaruhi karakter disiplin ilmu, strategi kelembagaan, pola kolaborasi, dan dinamika perencanaan riset; akibatnya, pendekatan "satu ukuran untuk semua" berisiko tidak adil sekaligus kurang mendukung keputusan yang strategis (Kouis et al., 2025). Karena itu, evaluasi yang hanya bersifat deskriptif cenderung kurang proaktif, sehingga pemodelan prediktif berbasis time-series menjadi relevan untuk melengkapi evaluasi melalui proyeksi tren kinerja (Cicero, 2025). Dengan demikian, tinjauan ini difokuskan

pada pemodelan prediktif kinerja dosen berbasis data publikasi melalui aplikasi *website* yang tidak hanya merangkum capaian, tetapi juga menyajikan analitik bibliometrik secara interaktif (Mondal, 2025).

Secara konseptual, data publikasi dosen dapat diperlakukan sebagai rangkaian waktu yang merepresentasikan indikator seperti jumlah publikasi, sitasi, atau ukuran turunan lain per periode, sehingga membutuhkan penataan periode dan konsistensi pencatatan agar dapat dianalisis. Perbedaan karakter sumber seperti *Google Scholar* yang lebih inklusif, Scopus yang lebih terkurasi, dan *SINTA* yang membawa logika agregasi/penilaian tertentu mengindikasikan perlunya pembersihan data, harmonisasi entitas, dan penyelarasan granularitas waktu sebelum pemodelan. Dalam konteks peramalan, pemisahan komponen *trend–seasonality–*residual dan pemilihan model *time-series* dapat membantu menjelaskan pola utama sekaligus mengelola variasi residu secara lebih sistematis (Zhang, 2023).

Jika ditinjau dari pendekatan prediksi, model yang berorientasi tren dan musiman cenderung lebih mudah ditafsirkan untuk kebutuhan aplikasi penilaian, sementara model yang lebih kompleks berpotensi menangkap pola nonlinier tetapi menuntut data dan pengaturan yang lebih ketat. Selain itu, perbedaan metode sering memunculkan variasi hasil karena perubahan tren (changepoints) dan pola periodik pada data, sehingga strategi pemodelan perlu disesuaikan dengan karakter time-series publikasi. Dalam konteks tersebut, pendekatan aditif yang menekankan

komponen tren dan musiman dapat menjadi pembanding yang kuat saat data memiliki perubahan tren yang jelas (Azeroual, 2024).

Dengan demikian, sintesis kajian mengarah pada kerangka bahwa kinerja publikasi dosen dapat direpresentasikan sebagai time-series multi-sumber yang dibersihkan dan dinormalisasi sebelum diprediksi. Perbandingan beberapa model diperlukan untuk menemukan keseimbangan antara akurasi, stabilitas prediksi, dan kemudahan interpretasi dalam aplikasi berbasis web. Evaluasi model sebaiknya dilakukan dengan metrik seperti RMSE dan MAE untuk mengukur kesalahan prediksi serta R^2 untuk melihat proporsi variasi yang dapat dijelaskan, sambil menyesuaikan interpretasinya dengan sifat data sekuensial. Secara praktis, hasil prediksi diharapkan tidak berhenti pada angka akurasi, tetapi diterjemahkan menjadi informasi yang mendukung keputusan, misalnya indikasi tren peningkatan/penurunan produktivitas dan horizon target capaian. Literatur forecasting juga mengindikasikan bahwa tidak ada satu model yang selalu unggul di semua kondisi, sehingga pengujian komparatif pada data publikasi dosen menjadi dasar penting untuk menetapkan pendekatan yang paling sesuai (Azeroual, 2024).

B. Landasan Teori

1. Identifikasi

Identifikasi dalam penilaian kinerja dosen berbasis publikasi menunjukkan bahwa banyak perguruan tinggi masih mengandalkan rekap manual dan laporan deskriptif, sehingga data dari *Google Scholar*, *Scopus*, dan *SINTA* belum dimanfaatkan secara optimal. Akibatnya, informasi kinerja yang dihasilkan terlambat, tidak seragam, dan kurang mendukung keputusan strategis. Padahal, kajian bibliometric analysis menegaskan bahwa publikasi dan sitasi dapat menjadi indikator evaluasi yang kuat jika proses identifikasi, pengolahan, dan analisis data dilakukan secara sistematis (Passas, 2024; Nature Research Intelligence, 2023). Dengan demikian, dapat diidentifikasi bahwa belum terdapat sistem penilaian kinerja dosen berbasis publikasi yang terotomasi, terintegrasi, dan didukung indikator yang jelas.

Pada sisi pemodelan prediktif, identifikasi masalah menunjukkan bahwa penilaian kinerja dosen belum mempertimbangkan karakter deret waktu data publikasi yang fluktuatif, mengikuti tren jangka panjang, dan mungkin memiliki pola musiman. Tanpa identifikasi sifat data ini, pemilihan model dan metrik yang kurang tepat berisiko menghasilkan prediksi yang tampak baik tetapi tidak andal pada berbagai kondisi (Hewamalage et al., 2023; Bergmeir, 2023). Oleh karena itu, identifikasi

pola dan karakteristik data menjadi dasar untuk merancang dan membandingkan model time-series agar hasil prediksi benar-benar mendukung pengambilan keputusan dalam sistem penilaian kinerja dosen berbasis web.

2. Publikasi Ilmiah

Publikasi ilmiah dalam penelitian ini dipahami sebagai keluaran tertulis hasil riset yang telah melalui telaah sejawat, dipublikasikan pada media bereputasi, dan menjadi dasar penilaian kinerja dosen melalui indeks seperti *Scopus*, *SINTA*, dan *Google Scholar* (Silva & Mesquita, 2023).

a. Scopus

Jurnal Scopus adalah jurnal ilmiah yang terindeks dalam basis data Scopus, yaitu pangkalan data sitasi internasional yang menyeleksi jurnal berdasarkan standar kualitas, reputasi, dan kelayakan akademik tertentu. Dalam penelitian, publikasi pada jurnal Scopus sering digunakan sebagai indikator kinerja dan reputasi ilmiah dosen maupun institusi.

b. SINTA (*Science and Technology Index*)

SINTA adalah jurnal ilmiah yang sudah terindeks dalam SINTA (Science and Technology Index), yaitu sistem daring milik Kemendikbudristek yang digunakan untuk mengukur kinerja publikasi, sitasi, peneliti, institusi, dan jurnal di Indonesia.

Dengan kata lain, jurnal SINTA adalah jurnal yang telah terdaftar dan dinilai dalam basis data SINTA, biasanya disertai peringkat akreditasi (S1–S6) serta berbagai indikator kinerja seperti jumlah sitasi, h-index, dan dampak jurnal.

c. *Google Scholar*

Google Scholar merupakan mesin telusur akademik yang mengindeks berbagai literatur ilmiah seperti artikel jurnal, prosiding, tesis, buku, dan laporan teknis dari beragam penerbit dan repositori. Platform ini membantu peneliti, dosen, dan mahasiswa menemukan, melacak sitasi, serta menganalisis sebaran dan dampak karya ilmiah secara lebih mudah dan terbuka.

3. Metode Kuantitatif

Metode kuantitatif dalam penelitian "Pemodelan Prediktif Kinerja Dosen Berbasis Data Publikasi dengan *Machine Learning Time-Series*" merujuk pada pendekatan penelitian yang berfokus pada pengolahan data numerik, seperti jumlah publikasi, sitasi, dan indikator kinerja lain yang terukur. Data tersebut dianalisis menggunakan teknik statistik dan algoritma *machine learning time-series* (misalnya ARIMA, LSTM, atau Prophet) untuk membangun model yang mampu memprediksi pola kinerja dosen di masa depan. Dengan demikian, metode kuantitatif menekankan

pengukuran objektif, pengujian model secara empiris, serta evaluasi akurasi prediksi menggunakan metrik seperti RMSE, MAE, dan R^2.

4. *Machine Learning*

Machine learning dalam "Pemodelan Prediktif Kinerja Dosen Berbasis Data Publikasi dengan Machine Learning Time-Series" dapat dipahami sebagai penerapan algoritma komputasi yang belajar dari pola historis data publikasi dosen (misalnya jumlah artikel atau sitasi per periode) untuk memprediksi kinerja pada periode berikutnya (Casolaro, Capone, Iannuzzo, & Camastra, 2023). Dalam konteks ini, model deret waktu berbasis machine learning seperti LSTM, ARIMA yang diperkaya fitur, atau model hybrid lain dilatih pada data publikasi sehingga mampu menangkap tren, pola musiman, dan fluktuasi yang kompleks dalam aktivitas publikasi ilmiah (Elsayed et al., 2023). Prediksi yang dihasilkan kemudian dimanfaatkan sebagai dasar penilaian dan perencanaan kinerja dosen secara lebih proaktif dan berbasis data, khususnya dalam memantau produktivitas riset dan merancang strategi peningkatan publikasi.

5. Model Prediktif

Model prediktif dalam *machine learning time-series* merujuk pada model yang dirancang untuk belajar dari pola pada data deret waktu historis (seperti jumlah publikasi per tahun) dan kemudian dimanfaatkan untuk memperkirakan nilai atau kondisi di periode berikutnya. Dalam

penerapannya, algoritma seperti ARIMA, SVR, LSTM, atau berbagai model hybrid dilatih menggunakan data masa lalu agar mampu mengenali tren, pola musiman, serta fluktuasi yang penting bagi proses pengambilan keputusan. Sejumlah penelitian terbaru di Indonesia menunjukkan bahwa kinerja model prediktif berbasis statistika dan machine learning dapat dibandingkan untuk menentukan metode peramalan deret waktu yang paling sesuai dengan karakteristik data yang digunakan (Andreas, 2025).

6. Bibliometrik

Bibliometrik dalam konteks *machine learning time-series* dapat dipahami sebagai penerapan pendekatan kuantitatif terhadap data publikasi (seperti jumlah artikel, sitasi, kata kunci, dan jejaring kolaborasi) untuk menggambarkan dinamika, kecenderungan, dan performa penelitian pada

suatu bidang. Informasi bibliometrik yang tersusun sebagai deret waktu, misalnya jumlah publikasi tahunan atau akumulasi sitasi, dapat dimodelkan menggunakan teknik *time-series berbasis machine learning* guna memprediksi laju pertumbuhan publikasi atau dampak ilmiah di masa depan. Di Indonesia, sejumlah kajian menunjukkan bahwa analisis bibliometrik efektif untuk memetakan perkembangan dan arah penelitian seperti pada studi inovasi kebijakan yang selanjutnya berpotensi diperluas dengan pemodelan prediktif deret waktu (Rahman, 2023).

7. *Database*

Basis data (database) dapat dipahami sebagai himpunan data/informasi yang terorganisasi dan saling terhubung, sehingga proses pengumpulan, pengolahan, pemanggilan kembali, serta pencarian data dapat dilakukan secara akurat untuk menunjang kegiatan operasional maupun pengambilan keputusan. Dalam pengembangan sistem informasi, rancangan database biasanya disusun melalui tahapan konseptual, logis, sampai fisik guna memodelkan entitas beserta hubungan antarentitasnya (misalnya menggunakan ERD), meminimalkan redundansi, serta menjaga agar pengelolaan data tetap efisien dan konsisten melalui dukungan DBMS dan aturan integritas data (Mukhlis & Santoso, 2023).

8. *Website*

Website adalah layanan berbasis internet yang dibuka melalui peramban (*browser*) dan terdiri dari sejumlah halaman yang saling terhubung untuk menyajikan informasi kepada pengguna melalui protokol web seperti HTTP (Albert & Ermatita, 2023). Karena dibangun dengan teknologi web, website dapat dikembangkan menjadi sistem informasi yang membantu mempercepat dan mengefisienkan proses layanan, misalnya untuk dokumentasi kegiatan, pengumpulan/unggah laporan, sampai layanan administrasi dalam satu portal terpadu (Albert & Ermatita, 2023). Di samping itu, studi terbaru menyatakan bahwa kinerja dan pengalaman

pengguna pada *website* dapat ditingkatkan dengan menerapkan konsep Progressive Web App (PWA) sehingga tampilan lebih adaptif di berbagai perangkat, dapat diinstal layaknya aplikasi, serta didukung *service worker* dan mekanisme caching agar akses lebih stabil dan andal (Apriyanti & Ramli, 2025).

9. *Django*

Django merupakan framework pengembangan web berbasis Python yang membantu proses pembuatan aplikasi web menjadi lebih cepat, terarah, dan relatif aman karena menyediakan struktur kerja yang jelas serta beragam komponen bawaan yang siap digunakan untuk kebutuhan umum aplikasi. Dalam beberapa publikasi jurnal di Indonesia, Django sering dipilih karena mendukung konsep rapid development, memiliki fitur keamanan, serta menawarkan arsitektur yang rapi (kerap dijelaskan dengan pendekatan MVC) sehingga pengelolaan data dapat dilakukan secara efisien dan tetap mampu berkembang sesuai kebutuhan. Pada praktiknya, Django juga mempermudah implementasi fitur seperti autentikasi pengguna dan pengaturan data aplikasi, misalnya pada sistem inventori berbasis QR code maupun platform kerja freelance berbasis web. Selain itu, Django dinilai fleksibel dalam pengelolaan basis data dan memudahkan pembangunan aplikasi web yang efisien melalui pemisahan komponen menggunakan pendekatan MVC (Sunardi, 2025).

10. *Google Colaboratory*

Google Colab berperan sebagai lingkungan komputasi berbasis cloud yang memungkinkan proses pemodelan deret waktu dilakukan secara efisien, terdokumentasi, dan mudah direplikasi tanpa tuntutan spesifikasi perangkat yang tinggi.

Gambar 2.0 *Google Colab Icon SVG Vector*. Diakses dari https://shorturl.at/FFWSz

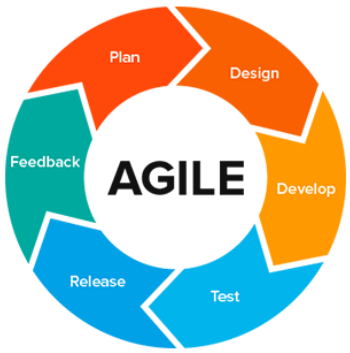
Dalam konteks penelitian di Indonesia, *Colab* kerap dijadikan platform utama untuk menjalankan *workflow* ARIMA (mulai dari pengolahan data hingga pemodelan) dengan dukungan pustaka analitik seperti *pandas*, *NumPy*, dan *statsmodels*, sehingga pelatihan dan evaluasi model dapat disatukan rapi dalam satu *notebook* (Sudewo dkk., 2024). Lebih lanjut, pada pemodelan musiman SARIMA, Colab juga menunjang tahapan pelatihan, evaluasi, dan validasi model secara fleksibel termasuk eksperimen parameter dan pengukuran galat yang mempercepat iterasi eksperimen serta meningkatkan keterlacakan proses penelitian (Alfiansyah dkk., 2025).

Dalam penerapan aplikasi penilaian kinerja dosen berdasarkan publikasi berbasis *website* yang memanfaatkan analisis prediktif deret waktu, penggunaan ARIMA dan SARIMA membuat fungsi peramalan lebih kuat karena sistem tidak hanya merekam capaian masa lalu, tetapi juga memperkirakan arah kinerja pada periode mendatang. ARIMA dapat dijadikan acuan awal untuk menangkap pola perubahan indikator kinerja, sedangkan SARIMA lebih tepat ketika terdapat pola musiman berulang sehingga estimasi menjadi lebih sesuai dengan siklus akademik (Alfiansyah dkk., 2025). Dengan adanya keluaran prediksi tersebut, aplikasi dapat berperan sebagai pendukung keputusan untuk pemantauan dan perumusan kebijakan akademik yang lebih efektif serta transparan (Army, 2025).

Selain itu, hasil peramalan dapat membantu pihak fakultas mengidentifikasi potensi penurunan produktivitas publikasi sejak dini sehingga langkah perbaikan dapat dilakukan lebih cepat. Informasi prediktif ini juga memudahkan perencanaan program peningkatan kapasitas, seperti pelatihan penulisan, pendampingan publikasi, maupun penentuan target kinerja yang realistis. Dengan demikian, sistem tidak hanya berfungsi sebagai alat evaluasi, tetapi juga sebagai instrumen perencanaan strategis yang mendorong peningkatan mutu publikasi secara berkelanjutan.

11. *Metode Agile*

Metode Agile merupakan pendekatan pengembangan perangkat lunak yang bersifat iteratif dan adaptif melalui siklus kerja yang berulang. Pada gambar, proses dimulai dari *Plan* (perencanaan), dilanjutkan *Design* (perancangan), *Develop* (pengembangan), *Test* (pengujian), dan *Release* (rilis). Setelah rilis, tahap *Feedback* (umpan balik) digunakan untuk mengevaluasi hasil dan menentukan perbaikan pada iterasi berikutnya, sehingga pengembangan dapat berlangsung lebih cepat, fleksibel, dan sesuai kebutuhan pengguna.



Gambar 2.1 Aguayo. (t.t.). *Agile Methodology in UX*. Diakses dari https://aguayo.co/en/blog-aguayo-user-experience/agile-methodology-in-ux/

Tahapan yang terdapat pada metode pengembangan *Agile* yang peneliti gunakan adalah sebagai berikut :

1. **Perencanaan (*Plan*)**

   Pada tahap perencanaan, peneliti mengidentifikasi kebutuhan dan tujuan aplikasi penilaian kinerja dosen berdasarkan publikasi ilmiah, termasuk ruang lingkup fitur seperti input data publikasi, perhitungan indikator penilaian, serta modul prediksi *time-series* dengan *machine learning*. Aktivitas utama meliputi pengumpulan kebutuhan dari pihak program studi, penentuan data yang digunakan (misalnya tahun publikasi, jenis publikasi, indeksasi), penyusunan *backlog*, penjadwalan iterasi (*sprint)*, serta penetapan kriteria keberhasilan sistem agar pengembangan berjalan terarah dan terukur.

2. **Desain Iterasi (*Design*)**

   Pada tahap desain iterasi, rancangan sistem dibuat secara bertahap sesuai prioritas *backlog*, mulai dari desain arsitektur aplikasi, alur proses penilaian, dan perancangan basis data publikasi dosen. Selain itu, dilakukan perancangan antarmuka pengguna (UI/UX) seperti halaman *login*, manajemen data dosen, manajemen publikasi, hasil penilaian, dan tampilan grafik prediksi *time-series*. Di sisi *machine learning*, tahap ini juga mencakup desain alur *preprocessing data*, pemilihan pendekatan model prediksi, serta skema integrasi model ke dalam aplikasi.

3. Pengembangan (*Development*)

Pada tahap pengembangan, implementasi dilakukan secara iteratif dengan membangun fitur inti terlebih dahulu, kemudian dikembangkan ke fitur lanjutan sesuai *sprint*. Pengembang membuat modul pengelolaan data (*CRUD*) dosen dan publikasi, modul perhitungan penilaian kinerja berdasarkan indikator yang ditetapkan, serta modul *machine learning* untuk prediksi *time-series* (misalnya prediksi tren publikasi per tahun). Setiap iterasi menghasilkan versi aplikasi yang dapat diuji, sehingga perbaikan dapat dilakukan lebih cepat berdasarkan umpan balik dan hasil evaluasi *sprint*.

4. Pengujian (*Test*)

Pada tahap pengujian, sistem diuji untuk memastikan fungsionalitas dan akurasi berjalan sesuai kebutuhan. Pengujian meliputi uji fitur aplikasi (validasi input, proses perhitungan, laporan penilaian), uji integrasi antara aplikasi dan model *machine learning*, serta uji kinerja prediksi *time-series* menggunakan metrik evaluasi yang sesuai (misalnya MAE/MSE/RMSE/MAD). Selain itu, dilakukan uji pengguna (*user acceptance*) bersama pihak terkait untuk memastikan aplikasi mudah digunakan, hasil penilaian logis, dan tampilan informasi prediksi dapat dipahami dengan baik.

5. Penerapan (*Deployment & Retrospective*)

Pada tahap penerapan, aplikasi yang telah lolos pengujian di *deploy* ke lingkungan penggunaan (server/lokal) agar dapat diakses oleh admin atau pihak program studi, disertai konfigurasi basis data dan dokumentasi penggunaan. Setelah implementasi, dilakukan *retrospective* untuk mengevaluasi proses pengembangan setiap *sprint*, mengidentifikasi kendala yang muncul (misalnya kualitas data publikasi atau kebutuhan fitur tambahan), serta merumuskan perbaikan untuk iterasi berikutnya. Tahap ini memastikan aplikasi tidak hanya berjalan, tetapi juga siap dikembangkan lebih lanjut sesuai kebutuhan prodi di masa depan.

12. UML

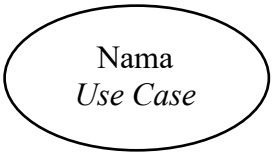
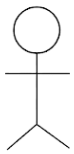
UML (*Unified Modeling Language*) merupakan bahasa pemodelan visual berorientasi objek yang digunakan untuk mendokumentasikan, menspesifikasikan, dan membangun rancangan perangkat lunak sehingga kebutuhan dan struktur sistem dapat dipahami secara konsisten oleh seluruh pemangku kepentingan (Aryani, Aqil, & Paramita, 2024). Dalam praktik perancangan, UML membantu merumuskan kebutuhan fungsional melalui use case diagram serta mengarahkan turunan desain berikutnya

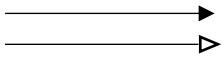
(misalnya skenario, *activity*, *sequence*, dan *class diagram*) agar pengembangan sistem lebih terstruktur, termasuk pada konteks sistem yang menuntut integrasi lintas aplikasi (Rospricilia, 2024). Dengan demikian, penerapan UML tidak hanya memperjelas komunikasi desain, tetapi juga meningkatkan kualitas pemodelan (misalnya ketepatan notasi dan relasi pada *use case*) sehingga risiko kesalahan rancangan dapat ditekan sejak tahap analisis dan desain (Aryani et al., 2024).

Sebagai lanjutan, UML memvisualisasikan kebutuhan sistem secara konsisten melalui diagram seperti *use case* dan alur proses agar desain sesuai kebutuhan pengguna (Aryani, Aqil, & Paramita, 2025). Pada perancangan sistem informasi perpustakaan, UML membantu menata rancangan untuk memperbaiki pendataan, pencarian, dan pencatatan transaksi (Aryani, Aqil, & Paramita, 2025). Dalam integrasi sistem, ketepatan notasi UML serta pemodelan *integration use case* (IUC) ditekankan agar kebutuhan integrasi tergambar jelas sejak awal (Rospricilia & Ma'ady, 2024).

a. *Use Case Diagram*

Tabel 2.1 Simbol Use Case Diagram

NO	SIMBOL	NAMA	KETERANGAN
1		*Use Case*	Sistem bekerja dengan membagi tugas ke beberapa bagian, lalu saling berkomunikasi dan bertukar informasi, baik satu sama lain maupun dengan pengguna.
2		*Actor*	Orang, kegiatan, atau sistem lain yang ikut terlibat dan berinteraksi dengan sistem informasi, tapi tidak termasuk bagian dari sistem itu sendiri.
3		*Assosiation*	Hubungan antar aktor dan Use Case yang berpartisipasi.
4		Ekstensi / extend	Relasi *Use Case* tambahan ke sebuah *use case* yang dapat berdiri sendiri.
5		Generalisasi	Hubungan generalisasi dan spesialisasi antara dua buah *Use Case* yang mana fungsi yang satu lebih umum dari yang lainnya.

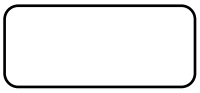
6		Include / Use Case	Relasi tambahan pada sebuah use case, yaitu use case yang ditambahkan membutuhkan use case ini agar fungsinya dapat berjalan.

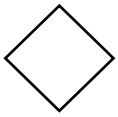
Sumber: Rospricilia, 2024, INTEGER: *Journal of Information*

Use Case Diagram adalah bagian dari UML yang digunakan untuk memetakan kebutuhan fungsional sistem dengan menonjolkan pihak yang berinteraksi (aktor) serta layanan yang diberikan sistem (*use case*), sehingga batasan dan cakupan fungsi sistem dapat dipahami sejak tahap analisis awal. Dalam proses perancangan, diagram ini meringkas hubungan pengguna sistem secara jelas dan sering dijadikan landasan untuk menyusun desain lanjutan dalam pengembangan perangkat lunak (Dharmawan, 2023). Pada unsur simbolnya, aktor biasanya divisualisasikan sebagai gambar manusia sederhana (*stick figure*) untuk mewakili entitas eksternal, sedangkan use case digambarkan berbentuk elips yang menunjukkan fungsi/fitur yang digunakan. Selain itu, Use Case Diagram membantu mengidentifikasi kebutuhan pengguna dan alur interaksi utama secara ringkas tanpa harus menjelaskan detail teknis implementasi.

b. *Activity Diagram*

Tabel 2.2 Simbol *Activity Diagram*

NO	SIMBOL	NAMA	KETERANGAN
1		Status Awal	Setiap diagram aktivitas diawali dari kondisi awal, yaitu titik pertama saat proses atau alur kerja mulai berjalan. Kondisi awal tersebut direpresentasikan dengan simbol lingkaran hitam terisi dan berfungsi sebagai penanda dimulainya keseluruhan rangkaian aktivitas pada sistem yang sedang dimodelkan.
2		Aktivitas	Aktivitas dalam suatu sistem merepresentasikan rangkaian aksi atau prosedur yang dijalankan oleh sistem. Umumnya, aktivitas tersebut dituliskan dengan diawali kata kerja karena menegaskan bentuk tindakan yang dilakukan, misalnya “mengolah data” atau “mengirim

			pemberitahuan". Penggunaan kata kerja pada penamaan aktivitas turut memudahkan pemahaman terhadap tujuan dan kontribusi tiap langkah dalam alur proses sistem.
3		Percabangan/*join*	Relasi bercabang yang digunakan ketika terdapat lebih dari satu alternatif aktivitas.
4		Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5		Status Akhir	*Final state* yang menandakan seluruh rangkaian aktivitas telah selesai. *Final state* ditunjukkan dengan simbol lingkaran konsentris dan berfungsi sebagai penutup proses. Keberadaannya memastikan alur kerja memiliki titik akhir yang jelas serta tidak berhenti pada kondisi yang menggantung.

Sumber: Nasution & Utami, 2024, Jurnal Sistem Informasi (JUSIN), 5(2)

Activity Diagram (Diagram Aktivitas) adalah salah satu jenis diagram perilaku (behavior diagram) pada *Unified Modeling Language* (UML) yang berfungsi untuk menggambarkan aliran kerja (*workflow*) maupun urutan kegiatan dalam suatu proses atau sistem. Melalui diagram ini, setiap aktivitas dapat dipetakan secara terstruktur beserta relasi antar aktivitasnya, sehingga pengembang dan pemangku kepentingan lebih mudah memahami mekanisme serta dinamika proses yang dimodelkan. Dengan demikian, *Activity Diagram* menjadi representasi visual yang menonjolkan cara proses berlangsung dari tahap awal hingga akhir melalui rangkaian aktivitas yang saling terhubung. (Amaliyah et al., 2023).

Pada kegiatan analisis dan perancangan sistem, *Activity Diagram* dimanfaatkan untuk memodelkan *workflow* yang terdapat pada sistem, proses bisnis, maupun fitur/menu dalam perangkat lunak. Penyusunan diagram ini membantu perancang menjabarkan langkah-langkah proses secara berurutan, menentukan titik pengambilan keputusan, serta mengidentifikasi kemungkinan adanya percabangan maupun aktivitas paralel yang perlu ditangani dalam rancangan. Oleh karena itu, *Activity Diagram* berperan penting dalam memperjelas kebutuhan fungsional serta menyelaraskan pemahaman antara hasil analisis kebutuhan dengan rancangan proses yang akan diterapkan. (Aryani et al., 2024).

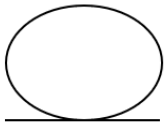
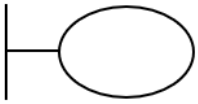
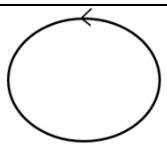
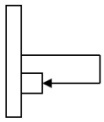
c. *Class Diagram*

Class Diagram merupakan bagian dari diagram struktur (*structural diagram*) pada *Unified Modeling Language* (UML) yang berfungsi untuk menampilkan gambaran struktur statis sebuah sistem. Diagram ini menyajikan kelas-kelas beserta hubungan di antaranya melalui simbol dan notasi grafis secara ringkas, sehingga pemodel dapat menjelaskan komponen inti sistem (misalnya kelas) serta keterkaitan antarkomponen dalam pendekatan berorientasi objek. Oleh karena itu, *Class Diagram* menjadi dasar yang penting untuk memahami susunan elemen sistem sebelum dikembangkan lebih lanjut ke tahap perancangan rinci dan implementasi. (Alturas, 2023).

Pada proses pengembangan perangkat lunak, *Class Diagram* tidak hanya dimanfaatkan sebagai dokumentasi desain, tetapi juga sebagai artefak yang dapat dievaluasi untuk meningkatkan mutu rancangan. Salah satu penerapannya ialah penggunaan metode berbasis *machine learning* untuk melakukan penilaian diagram secara otomatis, misalnya melalui analisis karakteristik kompleksitas desain seperti *coupling* dan kedalaman pewarisan, sehingga potensi kelemahan desain dapat terdeteksi sejak dini. Dengan demikian, *Class Diagram* dapat berperan sebagai objek analisis yang mendukung peningkatan *maintainability* serta kualitas perangkat lunak secara keseluruhan. (Shehzadi et al., 2025).

d. *Sequence Diagram*

Tabel 2.3 Simbol *Sequence Diagram*

Gambar	Nama	Keterangan
	*Entity Class*	Gambar Sistem sebagai landasan dalam menyusun basis data
	*Boundary Class*	Menangani komunikasi antar lingkungan sistem
	*Control Class*	Bertanggung jawab terhadap kelas-kelas objek yang berisi logika
	*Recursive*	Pesan untuk dirinya
	*Activation*	Mewakili proses durasi aktivasi sebuah operasi
	Life Line	Komponen yang digambarkan garis putus terhubung dengan objek

Sumber : Septiansyah Ade et al., 2024, Jurnal Teknologi Informasi

Sequence Diagram merupakan salah satu jenis diagram interaksi (*interaction diagram*) pada *Unified Modeling Language* (UML) yang berfungsi untuk memodelkan interaksi dinamis antarobjek atau komponen

sistem melalui rangkaian pesan (*message*) yang berlangsung seiring berjalannya waktu. Diagram ini menonjolkan aspek urutan kejadian dengan memperlihatkan "kapan" serta "bagaimana" suatu pesan dikirim dan diterima oleh tiap objek, sehingga perilaku sistem pada suatu skenario dapat dipahami secara lebih terstruktur dan nyata. Oleh sebab itu, *Sequence Diagram* membantu menggambarkan alur komunikasi yang merepresentasikan fungsi serta respons sistem sesuai kebutuhan yang diharapkan. (Kong et al., 2025).

Pada tahap analisis dan perancangan, *Sequence Diagram* digunakan untuk merinci *use case* menjadi urutan interaksi yang melibatkan aktor/objek, pemanggilan operasi, dan keterkaitan proses antarkomponen. Hasil pemodelan ini dapat dijadikan dasar penyusunan pengujian, misalnya melalui *path coverage* atau cakupan transisi yang kemudian diterjemahkan menjadi *test set* dan *test plan*. Selain itu, representasi kronologis pada *Sequence Diagram* membantu mengidentifikasi ketergantungan dan potensi titik kegagalan sejak dini sehingga skenario uji dapat disusun lebih terarah. Dengan demikian, *Sequence Diagram* berfungsi tidak hanya sebagai dokumentasi rancangan, tetapi juga mendukung verifikasi dan perencanaan pengujian berbasis model. (Górski & Stecz, 2024).