

BAB II. TINJAUAN PUSTAKA

2.1. Sistem Informasi

Sistem informasi adalah suatu kesatuan komponen dalam organisasi yang saling berinteraksi untuk mengolah data menjadi informasi yang dapat digunakan dalam kegiatan operasional serta proses pengambilan keputusan (Dewi & Sundari, 2021). Komponen tersebut meliputi sumber daya manusia, perangkat keras, perangkat lunak, metode, serta basis data yang bekerja sama untuk menghasilkan informasi yang diperlukan.

Dalam organisasi, sistem informasi dapat digunakan guna menunjang berbagai aktivitas, termasuk memfasilitasi saluran komunikasi, mengelola transaksi harian, serta memantau kegiatan baik di dalam maupun di luar organisasi. Informasi yang dihasilkan dapat menjadi salah satu dasar bagi organisasi dalam mengambil keputusan sesuai dengan kebutuhannya. Seiring dengan kemajuan teknologi, penggunaan sistem informasi berbasis komputer telah meningkat secara signifikan karena mempercepat pemrosesan data dan meminimalkan kesalahan. Namun, keterlibatan manusia tetap penting dalam pengendalian sistem dan pengambilan keputusan (Rahmah Mustika G et al., 2022).

Selain itu, penerapan sistem informasi turut membantu organisasi menjalankan kegiatannya dengan cara teratur, sehingga dapat mengelola data dan proses bisnis lebih efisien serta mendukung pencapaian tujuan organisasi (Fernandes, 2022). Sebagai salah satu bagian dalam sistem informasi, teknologi informasi berkontribusi pada pemrosesan serta penyebaran informasi. Informasi itu sendiri adalah data yang sudah diproses agar informasi tersebut bermakna dan dapat dimanfaatkan dalam pengambilan keputusan (Aldi Ramadani, 2025). Oleh karena itu,

penerapan sistem informasi mendukung pengelolaan data dan berbagai kegiatan organisasi secara keseluruhan.

2.2. IT Service Management (ITSM)

IT Service Management (ITSM) merupakan pendekatan yang digunakan untuk mengelola layanan TI agar selaras dengan kebutuhan organisasi dan membantu mencapai tujuan bisnis (Arjun et al., 2023). Penerapan ITSM tidak hanya terbatas di aspek teknologi, melainkan juga mencakup bagaimana layanan TI memberikan manfaat bagi organisasi dan penggunaannya (Celvine Adi Putra & Tata Sutabri, 2023).

ITSM menerapkan berbagai langkah, metode, dan standar yang diterapkan untuk merancang, mengelola, dan terus meningkatkan layanan teknologi informasi. Metode ini mencakup berbagai elemen, seperti manajemen layanan, manajemen sumber daya, manajemen risiko, dan pengendalian operasional untuk layanan TI (Krismayanti & Sutabri, 2023).

Seiring kemajuan teknologi, layanan TI kini tidak sekedar menjadi pendukung, namun juga memainkan peran strategis dalam operasional organisasi. Oleh karena itu, manajemen layanan TI perlu dilakukan secara terarah untuk mengurangi hambatan layanan dan memfasilitasi kelancaran proses bisnis (Riadi & Pratama, 2026). Dalam pelaksanaannya, ITSM mendukung organisasi dalam mengelola proses layanan, seperti manajemen insiden, manajemen perubahan, dan manajemen permintaan layanan. Selain itu, ITSM berperan penting dalam memantau, mengevaluasi, dan terus meningkatkan kualitas layanan untuk memenuhi harapan organisasi (Krismayanti & Sutabri, 2023).

Dengan demikian, ITSM berperan penting dalam membantu organisasi menyediakan layanan TI berkualitas tinggi, mengurangi risiko gangguan operasional, serta memastikan layanan yang

diberikan memenuhi kebutuhan pengguna dan tujuan organisasi (Herlinudinkhaji & Kurnia Ramadhani, 2023).

2.3. Change Management

Change Management adalah proses yang diterapkan untuk mengatur perubahan di dalam suatu organisasi agar berlangsung secara transparan dan konsisten. Perubahan-perubahan ini dilakukan sebagai bagian dari upaya organisasi untuk beradaptasi dengan perubahan di lingkungan internal maupun eksternal, guna memastikan organisasi tetap relevan dan mencapai tujuan yang telah ditentukan (Syandri Pratama, 2022). Proses perubahan di dalam suatu organisasi tidak hanya melibatkan unsur-unsur teknis, tetapi juga melibatkan struktur organisasi, proses operasional, serta perilaku individu di dalam organisasi. Oleh karena itu, manajemen perubahan harus dilaksanakan secara efektif untuk mencegah dampak negatif terhadap keberlanjutan organisasi.

Dalam dunia teknologi informasi, penerapan sistem baru dapat memengaruhi proses kerja dan interaksi pengguna, sehingga manajemen perubahan yang efektif menjadi sangat penting untuk mendukung keberhasilan implementasi. Manajemen perubahan berperan sebagai strategi yang mendukung organisasi dalam mengatur dan memandu proses perubahan agar sejalan dengan tujuan yang diinginkan (Nuri Aslami & M. Adhie Husni, 2023). Selain itu, praktik *change management* mencakup komunikasi yang transparan, keterlibatan pengguna, dan dukungan manajemen agar proses perubahan dapat diterima dengan baik oleh seluruh pihak yang terlibat. Hal ini penting untuk membantu individu mempersiapkan diri menghadapi perubahan dan mengurangi potensi penolakan di dalam organisasi.

Secara keseluruhan, *Change Management* melibatkan beberapa langkah, seperti mengusulkan perubahan, melakukan analisis dampak, memperoleh persetujuan, dan menerapkan

perubahan tersebut. Setiap langkah ini dilaksanakan secara sistematis untuk memastikan proses perubahan dikelola dan dievaluasi secara efektif.

Oleh karena itu, *Change Management* berperan penting dalam membantu organisasi mengelola perubahan secara terarah, mengurangi potensi risiko, dan memastikan bahwa perubahan yang diterapkan mendukung pencapaian tujuan organisasi.

2.4. Change Request

Change Request adalah pengajuan resmi yang dibuat untuk memperbaiki sistem teknologi informasi, mencakup aspek-aspek seperti hak akses, perangkat lunak, konfigurasi, dan infrastruktur TI. Proses ini digunakan guna mencatat dan mengelola setiap permohonan perubahan sehingga statusnya bisa dilacak dan diverifikasi.

Dalam praktiknya, *Change Request* biasanya diajukan melalui formulir atau dokumen yang dirancang untuk mendokumentasikan kebutuhan perubahan pengguna. Permintaan ini dapat mencakup permintaan akses, permintaan lisensi, atau pemasangan jaringan. Proses ini melibatkan beberapa langkah, seperti mengisi permintaan, proses persetujuan, dan memantau status perubahan (Rufiyadi et al., 2023).

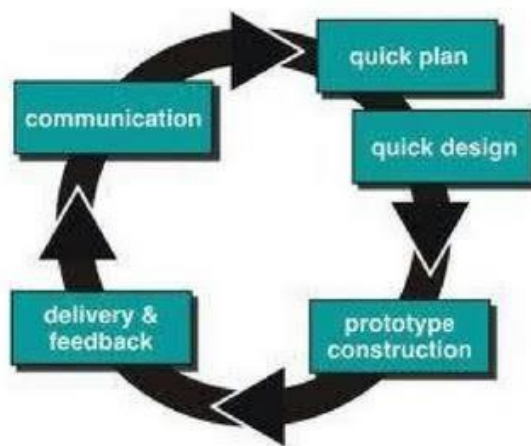
Namun, dalam kondisi tertentu, mengelola *Change Request* tanpa dukungan sistem dapat menimbulkan kendala, seperti kesulitan dalam proses pengajuan, keterlambatan dalam memperoleh persetujuan, dan keterbatasan dalam melacak status permintaan. Selain itu, penggunaan perangkat sederhana untuk mengelola dokumen perubahan juga dapat mempersulit proses pencarian dan pengelolaan arsip (Fernandes, 2022a).

Secara umum, proses *Change Request* mencakup beberapa tahap, antara lain pencatatan permintaan perubahan, penilaian, persetujuan, pelaksanaan perubahan, dan dokumentasi hasil.

Tahap-tahap ini diperlukan untuk memastikan bahwa semua perubahan dilaksanakan secara terarah dan terkendali sesuai dengan kebutuhan organisasi.

Dengan demikian, *Change Request* menjadi komponen penting dalam manajemen layanan teknologi informasi karena mendukung proses perubahan yang terdokumentasi dan dapat dipantau selama pelaksanaannya.

2.5. Model Prototype



(Kurniati 2021)

Gambar 2. 1 Model Prototype

Dalam pengembangan sistem informasi. *System Development Life Cycle (SDLC)* diterapkan sebagai proses pengembangan sistem. Salah satu model yang dapat digunakan adalah model *prototype*, yaitu pendekatan yang melibatkan pembuatan rancangan sistem awal untuk menggambarkan sistem yang akan dibangun. *Prototype* dapat membantu mengidentifikasi kebutuhan pengguna dengan lebih cepat (Ningsih and Nurfauziah 2023). Model *prototype* memungkinkan pengguna untuk melihat dan mengevaluasi rancangan sistem awal serta memberikan umpan balik sebelum sistem dikembangkan lebih lanjut. Melalui keterlibatan pengguna, rancangan tersebut dapat disesuaikan berdasarkan

kebutuhan yang telah ditetapkan (Ningsih and Nurfauziah 2023; Ashfiyan Ramadhani et al. 2025).

Dalam penelitian ini, model *prototype* dipilih karena sistem *IT Change Request* melibatkan banyak pihak dengan kebutuhan dan hak akses yang berbeda-beda. Persyaratan sistem diperoleh melalui observasi, wawancara, dan tinjauan dokumentasi, kemudian digunakan sebagai dasar untuk membuat dan mengembangkan *prototype*. Model ini memungkinkan desain sistem dievaluasi dan disesuaikan berdasarkan masukan pengguna (Marisa et al. 2024).

Proses pengembangan *prototype* umumnya terdiri dari beberapa tahap, yaitu:

1. *Communication* (Komunikasi): Tahapan ini dilakukan untuk mengumpulkan informasi mengenai kebutuhan dan permasalahan pengguna melalui interaksi bersama pihak-pihak yang terlibat dalam system.
2. *Quick Plan* (Perencanaan Cepat): Pada tahap ini, masalah diidentifikasi dan persyaratan sistem ditentukan berdasarkan informasi yang diperoleh. Persyaratan sistem kemudian dikelompokkan sesuai dengan fungsi pengguna dan hak akses.
3. *Modelling Quick Design* (Pemodelan dan Perancangan Cepat): Tahapan ini digunakan untuk membuat rancangan sistem awal berdasarkan persyaratan yang telah ditetapkan, rancangan tersebut dapat diilustrasikan menggunakan UML.
4. *Construction of Prototype* (Pembuatan Prototype): Langkah ini, desain diubah menjadi *prototype* sistem dikembangkan berdasarkan hasil analisis dan desain sebelumnya sehingga pengguna dapat memvisualisasikan sistem yang akan mereka gunakan.

5. *Deployment, Delivery, and Feedback* (Penyebaran, Pengiriman, dan Umpan Balik): Pada tahap ini, *prototype* dievaluasi dan mengumpulkan umpan balik pengguna sebagai dasar untuk perbaikan sistem.

Tabel 2. 1 Kelebihan dan Kekurangan Model Prototype

No	Kelebihan	Kekurangan
1	Pengguna dapat memperoleh gambaran awal mengenai sistem sebelum sistem dikembangkan sepenuhnya	Membutuhkan keterlibatan pengguna sepanjang proses pengembangan
2	Persyaratan dapat dimodifikasi saat sistem masih berada pada tahap prototype	Proses pengembangan mungkin memerlukan beberapa putaran penyempurnaan prototype
3	Pengguna dapat memberikan masukan langsung mengenai desain sistem	Jika masukan pengguna terus berubah, proses pengembangan mungkin akan memakan waktu lebih lama
4	Pengujian dan evaluasi dapat dimulai segera setelah prototype dibuat, sehingga desain dapat disesuaikan untuk memenuhi kebutuhan pengguna	Prototype tersebut belum mewakili versi akhir dan keseluruhan sistem
5	Meningkatkan kesesuaian sistem dengan kebutuhan dan kenyamanan pengguna karena pengguna terlibat	Membutuhkan koordinasi yang efektif antara pengembang serta pengguna agar persyaratan sistem dapat didefinisikan dengan jelas

	secara aktif dalam proses pengembangan	
--	--	--

(Adi Kurniyanti & Murdiani, 2022)

Berdasar tabel tersebut, model *Prototype* memiliki keunggulan dalam memberikan gambaran awal mengenai sistem dan memungkinkan pengguna untuk memberikan masukan selama proses pengembangan. Namun, model ini memerlukan keterlibatan pengguna dan mungkin membutuhkan beberapa putaran perbaikan *prototype*. Dengan demikian, memilih model *prototype* pada studi ini disesuaikan dengan kriteria sistem *IT Change Request*, yang melibatkan banyak pengguna dan memerlukan penyesuaian sistem berdasarkan kebutuhan pengguna.

2.6. ASP.NET Core

ASP.NET Core merupakan *framework* pengembangan aplikasi *web* bersifat *open-source* yang dibuat oleh *Microsoft* sebagai penerus *ASP.NET Framework* ini dirancang agar dapat berjalan di beragam sistem operasi, termasuk *Windows*, *Linux*, dan *macOS*. *ASP.NET Core* memiliki arsitektur modular dan mendukung pengembangan *cross-platform* dengan beragam pilihan *hosting*. *Framework* ini juga menyediakan model pengembangan berbasis *Model-View-Controller (MVC)* serta *web service*, sehingga memudahkan pengembangan sistem sesuai kebutuhan pengguna (Fajri et al., 2022).

Pada penelitian ini, *ASP.NET Core* dipilih sebagai *framework* pengembangan sistem karena selaras dengan teknologi yang digunakan oleh *Departement IT & System di PT Agincourt Resources*. Penggunaan *framework* yang sama mendukung kesesuaian sistem yang dirancang dengan lingkungan pengembangan perusahaan, serta mempermudah implementasi, pemeliharaan, dan pengembangan lanjutan sesuai kebutuhan. Selain itu, *ASP.NET Core* mendukung pengembangan aplikasi berbasis

web dengan arsitektur yang terstruktur, sehingga cocok untuk membangun sistem *IT Change Request* dengan alur kerja yang jelas dan hak akses pengguna yang tepat.

Dalam pengembangannya, *ASP.NET Core* menggunakan pola *Model-View-Controller (MVC)*, guna membedakan logika bisnis, tampilan, serta pemrosesan data (Christanto & Wibowo, 2020). Selain itu, *ASP.NET Core* dapat digunakan untuk mengembangkan beberapa jenis aplikasi, seperti aplikasi *web*, layanan *backend*, dan aplikasi *Internet of Things (IoT)*. *Framework* ini menggunakan bahasa pemrograman C# dan dirancang untuk memfasilitasi pengembangan aplikasi yang fleksibel dan dapat disesuaikan lebih lanjut sesuai kebutuhan (Fajri et al., 2022).

Berdasarkan karakteristik tersebut, *ASP.NET Core* dianggap cocok untuk mendukung perancangan sistem *IT Change Request* dalam penelitian ini. Selain arsitektur yang terstruktur, penggunaan *framework* ini juga selaras dengan teknologi yang diterapkan di *PT Agincourt Resources*, sehingga memudahkan proses perancangan dan implementasi sistem.

2.6.1. Arsitektur Model-View-Controller (MVC)

Model-View-Controller (MVC) merupakan pola arsitektur perangkat lunak yang memisahkan aplikasi menjadi tiga bagian utama, yaitu *Model*, *View*, dan *Controller*. Pendekatan ini membedakan logika bisnis, tampilan pengguna, serta aliran data, sehingga struktur aplikasi menjadi lebih terorganisir dan lebih mudah dikembangkan. *Model* bertanggung jawab terhadap pengelolaan data dan interaksi dengan basis data, sedangkan *View* digunakan untuk menunjukkan informasi ke pengguna, *Controller* bertindak sebagai penghubung antara *Model* dan *View* serta mengelola interaksi pengguna dengan sistem (Rahmawati & Sumarsono, 2024).

Penerapan arsitektur MVC menawarkan beberapa manfaat, termasuk pengelolaan kode lebih mudah, pengorganisasian struktur aplikasi yang lebih baik, serta dukungan untuk pemeliharaan dan pengembangan sistem jangka panjang. Dengan pemisahan tanggung jawab di antara komponen-komponen, setiap bagian dapat dikembangkan atau diubah tanpa memengaruhi bagian lainnya (Anggarah et al., 2025).

Selain itu, penggunaan arsitektur MVC mendukung pengembangan aplikasi secara bertahap, dan mempermudah penambahan fitur serta penyesuaian sistem pada tahap pengembangan selanjutnya. Pendekatan ini dianggap efektif dalam mendukung fleksibilitas pengembangan sistem, terutama untuk aplikasi dengan persyaratan yang terus berkembang (Tarigan et al., 2020).

Oleh karena itu, arsitektur *Model-View-Controller* (MVC) banyak diterapkan dalam pengembangan aplikasi karena mampu mendukung pembentukan sistem yang terstruktur, mudah dikelola, dan mendukung proses pengembangan yang berkelanjutan.

2.6.2. Bahasa Pemrograman C#

C# (*C Sharp*) adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh *Microsoft* dan digunakan pada *platform* .NET untuk membangun berbagai jenis aplikasi, termasuk aplikasi desktop, aplikasi *web*, serta layanan *web*. C# memiliki sintaks yang terstruktur dengan baik dan mendukung konsep *Object-Oriented Programming* (OPP), sehingga pengelolaan logika program dapat dilakukan secara modular dan terstruktur. Selain itu, C# terintegrasi dengan baik pada teknologi *Microsoft* dan membantu

memastikan keamanan serta kinerja yang optimal dalam pengembangan aplikasi (Riziq et al., 2023).

Dalam praktiknya, C# banyak digunakan dalam pengembangan aplikasi desktop maupun *web* karena didukung oleh ekosistem .NET yang luas. Bahasa ini memungkinkan pembuatan aplikasi yang terintegrasi dengan sistem operasi serta memanfaatkan berbagai pustaka dan kerangka kerja yang sudah ada (Amrullah A et al., 2025). C# juga dikenal sebagai bahasa pemrograman dengan sintaks yang *relative* sederhana dan kata kunci (*keywords*) yang jelas, sehingga lebih mudah dipelajari dan digunakan untuk pengembangan aplikasi. Bahasa ini telah distandarisasi secara internasional dan digunakan secara luas dalam berbagai bidang pengembangan perangkat lunak (Hasbi et al., 2021).

Dengan fitur-fitur tersebut, C# banyak digunakan dalam pengembangan aplikasi karena mendukung pengelolaan kode yang terstruktur, integrasi dengan berbagai teknologi, serta fleksibilitas dalam membangun beragam sistem.

2.7. Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan bahasa pemodelan visual yang berfungsi untuk mendeskripsikan, merancang, dan mendokumentasikan sistem perangkat lunak. UML membantu pengembang memahami persyaratan sistem dan memvisualisasikan desain sistem sebelum fase implementasi dimulai. Pada pengembangan perangkat lunak, UML digunakan dalam proses pemodelan dalam *System Development Life Cycle (SDLC)* untuk mendeskripsikan hubungan antara komponen sistem,

interaksi pengguna, dan alur proses yang terjadi di dalam sistem (Wayahdi & Ruziq, 2023).

UML juga menjadi salah satu standar yang digunakan dalam pengembangan sistem berorientasi objek yang untuk menjelaskan, menggambarkan, dan membuat perangkat lunak secara terstruktur. UML memiliki beberapa jenis diagram, di antaranya *use case diagram*, *activity diagram*, dan *class diagram*, setiap diagram berfungsi untuk menggambarkan aspek-aspek spesifik dari sistem (Ayu Binangkit et al., 2023).

UML juga dapat digunakan sebagai media dokumentasi untuk memfasilitasi komunikasi antara pengembang dan pengguna sistem. Dengan dokumentasi visual yang jelas, proses pengembangan tetap berjalan sesuai rencana serta mendukung pemeliharaan sistem dan pengembangan lebih lanjut pada tahap berikutnya (Aryani et al., 2025).

Pada penelitian ini, UML digunakan untuk membantu merancang sistem dengan alur proses yang jelas dan struktur yang terdefinisi dengan baik. Penggunaan UML diharapkan dapat membantu memvisualisasikan persyaratan sistem dan mendukung proses pengembangan aplikasi yang lebih terarah.

2.7.1. Use Case Diagram

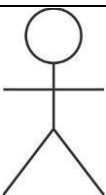
Use Case Diagram adalah salah satu jenis diagram pada *Unified Modeling Language (UML)* yang digunakan untuk memvisualisasikan interaksi antara aktor dengan sistem. Diagram tersebut menggambarkan pihak-pihak yang terlibat dalam sistem (*actor*) beserta fungsi atau layanan yang tersedia di dalam sistem. Aktor dapat berupa pengguna maupun sistem lain yang melakukan interaksi secara langsung untuk menjalankan fungsi tertentu (Wayahdi & Ruziq, 2023).

Selain itu, *Use Case Diagram* juga digunakan untuk menunjukkan batas sistem (*system boundary*), yang membedakan proses di dalam sistem dengan aktivitas pengguna atau lingkungan di luar sistem. Diagram ini juga sering digunakan untuk menggambarkan gambaran umum (*high-level*) dari sistem, sehingga membantu penyampaian informasi antara pengembang dan pemangku kepentingan non-teknis (Aldi Ramadani, 2025; Wayahdi & Ruziq, 2023).

Dalam penelitian ini, *use case diagram* digunakan untuk menentukan aktor yang berinteraksi dengan sistem serta fungsi utama dalam sistem *IT Change Request*. Melalui diagram tersebut, hubungan antara pengguna dan sistem dapat divisualisasikan dengan jelas, sehingga membantu proses perancangan sistem.

Berdasarkan penjelasan tersebut, simbol-simbol yang digunakan dalam *Use Case Diagram* dapat disajikan pada tabel berikut:

Tabel 2. 2 Simbol pada Use Case Diagram

Simbol	Keterangan
 <i>Actor</i>	Membuat entitas yang saling berinteraksi dengan sistem, seperti pengguna atau sistem lain.
 Use Case	Menggambarkan fungsi atau layanan yang diberikan sistem kepada aktor.
<i>Garis/ Association</i> 	Menunjukkan hubungan antara aktor dan <i>use case</i> .

<code><<Include>></code> 	Menunjukkan bahwa suatu <i>use case</i> selalu mencakup fungsi dari kasus penggunaan lain.
<code><<Extend>></code> 	Menunjukkan adanya perluasan suatu <i>use case</i> terhadap <i>use case</i> lain pada kondisi tertentu.

(Dicoding, dicoding 2021)

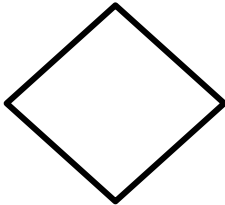
2.7.2. Activity Diagram (Swimlanes)

Activity Diagram merupakan diagram dalam UML yang digunakan untuk memvisualisasikan rangkaian aktivitas dan alur kerja yang berlangsung dalam suatu sistem. Diagram ini menunjukkan proses aktivitas mulai dari tahap awal hingga tahap akhir, termasuk perpindahan alur dari satu aktivitas ke aktivitas selanjutnya (Wayahdi & Ruziq, 2023).

Activity Diagram dapat digunakan untuk menggambarkan proses bisnis dan logika sistem secara lebih rinci, termasuk menggambarkan urutan tindakan, cabang keputusan, serta kondisi awal dan akhir suatu proses. Selain itu, diagram ini dapat merepresentasikan aktivitas yang berlangsung secara berurutan maupun paralel dalam sistem (Aldi Ramadani, 2025; Wayahdi & Ruziq, 2023). Dalam pengembangannya, *activity diagram* dapat digabungkan dengan jalur (*Swimlanes*) untuk mengelompokkan aktivitas berdasarkan pihak atau peran yang bertanggung jawab, pembagian tersebut membantu memperjelas alur kerja pada sistem.

Pada studi ini, *activity diagram* berfungsi sebagai alat guna memvisualisasikan alur proses dalam sistem *IT Change Request*, memastikan bahwa setiap tahap aktivitas dipahami dengan jelas dan memfasilitasi proses desain sistem

Tabel 2. 3 Simbol pada Activity Diagram

Simbol	Keterangan
<i>Initial Node</i> 	Menandai awal dari suatu rangkaian proses atau aktivitas.
<i>Activity</i> 	Menggambarkan aktivitas atau proses yang berlangsung di dalam sistem.
<i>Decision</i> 	Menunjukkan percabangan proses yang ditentukan berdasarkan suatu kondisi.
<i>Control Flow</i> 	Menggambarkan perpindahan alur dari satu aktivitas menuju aktivitas berikutnya.
<i>Final Node</i> 	Menandai berakhirnya suatu rangkaian proses atau aktivitas.
<i>Swimlane</i> 	Digunakan untuk mengelompokkan kegiatan berdasarkan peran atau pihak yang bertanggung jawab.

(Dicoding, dicoding 2021)

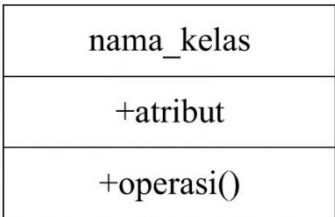
2.7.3. Class Diagram

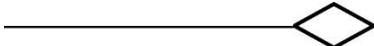
Class Diagram merupakan salah satu diagram UML yang berfungsi untuk memvisualisasikan struktur statis pada sistem. Diagram ini memperlihatkan kelas-kelas yang terdapat dalam sistem, termasuk atribut, metode (operasi), serta hubungan antar kelas. *Class diagram* membantu menggambarkan bagaimana objek saling terhubung melalui hubungan seperti *association*, *aggregation*, dan *dependency*. Setiap kelas mewakili sebuah objek yang memiliki atribut dan metode spesifik yang mendefinisikan data dan fungsi objek tersebut dalam sistem (Wayahdi & Ruziq, 2023).

Class diagram juga berfungsi sebagai alat desain karena memberikan gambaran mengenai struktur data dan keterkaitan antar komponen sistem. Diagram ini juga memfasilitasi komunikasi di antara para pengembang dan berfungsi sebagai landasan untuk fase implementasi sistem (Aldi Ramadani, 2025; Wayahdi & Ruziq, 2023).

Dengan demikian, *class diagram* digunakan untuk memberikan gambaran menyeluruh mengenai struktur sistem. Pemodelan tersebut dapat menjadi acuan dalam pengembangan aplikasi agar implementasinya tetap sesuai dengan kebutuhan yang sudah ditetapkan.

Tabel 2. 4 Simbol pada Class Diagram

Simbol	Keterangan
<i>Class</i> 	Menunjukkan struktur suatu objek yang memiliki atribut dan metode (operasi).

<i>Association</i> 	Menunjukkan keterkaitan antara satu kelas dengan kelas lainnya.
<i>Aggregation</i> 	Menunjukkan hubungan “ <i>has-a</i> ” yaitu ketika suatu kelas menjadi bagian dari kelas lainnya namun tetap dapat berdiri secara sendiri.
<i>Dependency</i> 	Menggambarkan ketergantungan suatu kelas terhadap kelas lainnya.

(firmansyah 2025)

2.8. MySQL

MySQL merupakan sistem manajemen basis data relasional (*Relational Database Management System*) yang digunakan dalam pengembangan sistem informasi, terutama untuk aplikasi berbasis *web*. MySQL dirancang untuk menyimpan, mengelola, dan mengolah data secara teratur, sehingga mempermudah pengaturan informasi pada sistem. Selain itu, MySQL mendukung operasi pemrosesan data seperti *Create*, *Read*, *Update*, dan *Delete* (*CRUD*) serta memastikan konsistensi dan integritas data di berbagai jenis aplikasi (Jesika Vinatalia Dachi & Jiyan Suhada, 2025).

MySQL sering digunakan bersama dengan bahasa pemrograman *web* seperti PHP untuk membangun aplikasi terintegrasi. Kombinasi ini memungkinkan penyimpanan dan pengambilan data dilakukan dengan cepat dan efisien, sehingga mendukung kinerja sistem dalam mengelola data secara *real-time* (Handayani et al., 2024). Penggunaan MySQL pada sistem informasi juga terbukti meningkatkan efisiensi pengelolaan data karena data

dapat tersusun secara teratur dan lebih mudah untuk diakses. Hal ini menyederhanakan kegiatan pencarian, pemrosesan, dan pembuatan laporan di dalam sistem (Rizmawan et al., 2025).

Berdasarkan uraian di atas, MySQL dapat digunakan sebagai database pada aplikasi sistem informasi guna mengarsipkan berbagai bentuk data, seperti data pengguna, data transaksi, dan catatan aktivitas sistem. Penggunaan MySQL mendukung penyimpanan data terpusat, menjaga konsistensi hubungan antar entitas, serta memudahkan proses pemantauan dan pelaporan.

2.9. Black Box Testing

Black Box Testing adalah teknik pengujian perangkat lunak yang berfokus pada pemeriksaan fungsi sistem tanpa mempertimbangkan bagian internal atau kode program yang digunakan. Pengujian dilakukan dengan memberikan sejumlah input dan selanjutnya melihat output yang diberikan untuk memastikan kesesuaian terhadap spesifikasi yang sudah ditentukan sebelumnya (Putri et al., 2024).

Metode ini menekankan pada perspektif pengguna, sehingga pengujian berfokus terhadap fungsi sistem, tampilan pengguna, dan *respons* sistem pada data yang dimasukkan. Dengan pendekatan ini, penguji tidak harus memahami bahasa pemrograman ataupun logika internal sistem, karena yang diuji adalah perilaku eksternal sistem (Novalia & Voutama, 2022).

Black Box Testing dapat dimanfaatkan untuk mengidentifikasi beragam jenis kesalahan, seperti fungsi yang tidak beroperasi sesuai yang diharapkan, kesalahan dalam pemrosesan data, kesalahan pada struktur maupun akses basis data, serta kesalahan pada antarmuka pengguna. Selain itu, metode ini dapat mendeteksi kesalahan dalam inisialisasi dan penghentian sistem (Novalia & Voutama, 2022; Putri et al., 2024). Pengujian dilakukan dengan

mengembangkan kasus uji berdasarkan persyaratan sistem, kemudian memberikan variasi input, termasuk data yang valid dan tidak valid. Hal tersebut dilakukan untuk memastikan bahwa sistem dapat menangani berbagai kondisi serta menghasilkan output yang memenuhi harapan (Putri et al., 2024).

Berdasarkan uraian tersebut, *Black Box Testing* diterapkan sebagai metode pengujian guna memastikan setiap fungsi dalam aplikasi berjalan sesuai dengan persyaratan pengguna dan mampu memproses *input* serta menghasilkan *output* dengan benar sesuai dengan persyaratan yang ditentukan.

2.10. Machine Learning

Machine Learning merupakan bagian dari *Artificial Intelligence (AI)* yang memungkinkan perangkat mempelajari pola dari informasi untuk menghasilkan prediksi maupun melakukan klasifikasi tanpa harus diprogram khusus untuk setiap kondisi. Proses *machine learning* menggunakan algoritma dan model untuk memproses data serta mengidentifikasi pola yang dapat digunakan dalam pengambilan keputusan atau memprediksi hasil untuk data baru (Barati Bagherabad et al. 2026).

Machine learning sudah banyak diterapkan dalam berbagai bidang untuk menyelesaikan permasalahan yang didorong oleh data. Beberapa teknik ML dapat digunakan untuk *classification*, *regression*, dan *clustering*. Hasil penerapan ML dapat dipengaruhi oleh karakteristik data dan metode yang digunakan, sehingga pemilihan metode harus disesuaikan dengan masalah yang dihadapi (Razzaq and Shah 2025). Dalam pengembangan sistem perangkat lunak, penerapan ML memerlukan pengelolaan data dan model yang tepat untuk memastikan keduanya dapat dimanfaatkan dan dipelihara secara efektif (Berberi et al. 2025). Pada penelitian ini, ML dimanfaatkan sebagai fitur pendukung untuk memberikan prediksi

awal mengenai *change priority* berdasarkan informasi tekstual dalam *change request*. Proses prediksi dilakukan dengan menggunakan *multilingual embedding model* untuk merepresentasikan teks, dan hasilnya selanjutnya dikelompokkan dengan algoritma *Support Vector Machine (SVM)*.

2.11. Multilingual Embedding Model

Multilingual Embedding Model merupakan pendekatan dalam *Natural Language Processing (NLP)* yang bertujuan mengonversi teks ke dalam bentuk numerik atau *vector*. Hasil representasi tersebut memungkinkan informasi dalam teks diproses oleh model *machine learning*. Perkembangan model NLP berbasis *Transformer* juga mendukung pemrosesan teks dengan memahami pola bahasa dan konteks (Rosado Da Cruz and Cruz 2025).

Pemrosesan teks yang melibatkan lebih dari satu bahasa memiliki kompleksitas tersendiri karena setiap bahasa memiliki karakteristik dan pola yang berbeda (Hidayatullah et al. 2024). Pendekatan *multilingual embedding* memungkinkan teks dari berbagai bahasa direpresentasikan dalam ruang *vector* yang sama, sehingga teks dengan makna atau konteks serupa dapat memiliki representasi yang dekat meskipun menggunakan bahasa yang berbeda (Krisnawati et al. 2024).

Pada penelitian ini, *multilingual embedding model* diterapkan karena dataset yang diterapkan untuk melatih model berasal dari Jira dan berbahasa Inggris, sedangkan pengguna sistem dapat memasukkan informasi *change request* baik dalam bahasa Indonesia ataupun bahasa Inggris. Oleh sebab itu, model *embedding multibahasa* digunakan untuk mengubah teks dari kedua bahasa tersebut menjadi representasi *vector* sebelum melakukan proses klasifikasi menggunakan algoritma SVM.

2.12. Support Vector Machine (SVM)

Support Vector Machine (SVM) merupakan *algoritma supervised learning* yang dapat diterapkan untuk proses klasifikasi. Algoritma ini melakukan analisis terhadap data yang sudah diberi label untuk menemukan pola dan mengelompokkan data ke dalam kategori tertentu (Jabardi 2025).

Dalam proses klasifikasi, SVM mencari batas pemisah terbaik atau *hyperplane*, yang dapat memisahkan kelas-kelas dengan jarak atau margin maksimum. SVM juga dapat diaplikasikan untuk mengklasifikasikan lebih dari dua grup (*multiclass classification*) dengan menggunakan strategi tertentu (Jabardi 2025).

Dalam penelitian ini, SVM digunakan untuk mengklasifikasikan representasi teks dari *change title*, *description of change*, dan *reason for change* ke dalam tiga kelas *change priority*, yaitu *high*, *medium*, dan *low*. Sebelum diproses oleh SVM, teks tersebut terlebih dahulu dikonversi menjadi representasi *vector* menggunakan *multilingual embedding model*, sehingga model dapat memproses masukan dalam bahasa Indonesia maupun bahasa Inggris.

2.13. Evaluasi Kinerja Model Machine Learning

Evaluasi kinerja dari model *machine learning* merupakan langkah yang dilakukan untuk menentukan seberapa baik model tersebut bekerja dalam membuat prediksi atau klasifikasi berdasarkan data. Dalam proses *supervised learning*, model dilatih dengan data pelatihan dan kemudian diuji dengan data uji, hasil evaluasi ini dapat digunakan untuk menilai kinerja model dan membandingkannya dengan model lain yang digunakan dalam penelitian (Rainio et al. 2024). Dalam penelitian klasifikasi, *confusion matrix* digunakan untuk menilai prediksi model. Ini menunjukkan perbandingan antara kelas yang diprediksi dan kelas yang

sebenarnya. Berdasarkan *confusion matrix*, kinerja model dapat dinilai dengan metrik seperti *accuracy*, *precision*, *recall* dan *F1-score* (Rainio et al. 2024; Opitz 2024).

1. Accuracy

Accuracy digunakan untuk menghitung proporsi prediksi yang benar *relative* terhadap jumlah total seluruh data yang diuji. Nilai *accuracy* dapat dihitung menggunakan persamaan berikut:

$$Accuracy = \frac{Jumlah\ Prediksi\ Benar}{Jumlah\ Seluruh\ Data}$$

Semakin tinggi skor *accuracy*, semakin besar proporsi data keseluruhan yang berhasil diklasifikasikan dengan benar oleh model (Rainio et al. 2024).

2. Precision

Precision menunjukkan akurasi model saat membuat prediksi untuk kelas tertentu. Nilai ini menggambarkan berapa banyak prediksi untuk suatu kelas yang benar dibandingkan dengan jumlah total seluruh data yang diprediksi termasuk dalam kelas tersebut (Rainio et al. 2024).

$$Precision = \frac{TP}{TP + FP}$$

3. Recall

Nilai *recall* menunjukkan kemampuan suatu model untuk mengidentifikasi data yang benar-benar termasuk dalam suatu kelas tertentu. Nilai *recall* menunjukkan rasio antara jumlah total data dalam kelas yang diprediksi dengan benar terhadap jumlah total data yang sebenarnya termasuk dalam kelas tersebut (Rainio et al. 2024).

$$Recall = \frac{TP}{TP + FN}$$

4. F1-Score

F1-Score adalah nilai dalam metrik yang menggabungkan *precision* dan *recall* menjadi satu nilai. Nilai ini dapat digunakan untuk memberikan gambaran yang lebih seimbang mengenai kinerja klasifikasi suatu model (Rainio et al. 2024; Opitz 2024).

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Dalam studi ini, metrik *accuracy*, *precision*, *recall*, dan *F1-score* digunakan untuk mengevaluasi kinerja model *Support Vector Machine (SVM)* untuk mengelompokkan prioritas *IT Change Request* ke dalam kategori *high*, *medium*, dan *low*. Hasilnya kemampuan model ditentukan melalui evaluasi ini, dalam menyusun data dan mengevaluasi kinerja model pada semua kategori yang memiliki prioritas.

2.14. Analysis of Research Gap

Tabel 2. 5 Analysis Research Gap

Peneliti	Fokus Penelitian	Metode/Teknologi	Research Gap	Posisi Penelitian ini
(Fernandes, 2022)	<i>Monitoring Engineering Change Request</i>	<i>Web-Based</i>	Berfokus pada pemantauan ECR dalam proses engineering dan belum membahas penerapan IT Change Request maupun prediksi priority	Merancang sistem <i>IT Change Request</i> berbasis web serta menerapkan ML untuk prediksi change priority
(Rufiyadi et al., 2023)	<i>Sistem Change Request Form (CRF)</i>	<i>V-Model</i>	Sistem ini dikembangkan menggunakan PHP dan MySQL untuk menangani proses CRF di PT ADM tetapi belum mengintegrasikan prediksi priority berbasis ML	Merancang sistem IT Change Request menggunakan ASP.NET Core dengan arsitektur MVC dan ML
(Meisha Putri et al., 2024)	Implementasi ITSM	<i>ITIL V3</i>	Menghasilkan rekomendasi ITSM dan roadmap implementasi, dan belum	Mengimplementasikan proses IT Change Request ke dalam sistem berbasis web dan

			tersedianya tools change management dan ML yang terintegrasi	prediksi change priority untuk mendukung pengelolaan perubahan secara terpusat
(Syandri Pratama, 2022)	Implementasi <i>Change Management</i>	Model 8 Tahap	Penelitian ini berfokus pada penerapan change management dalam organisasi dan belum menerapkan proses ini ke dalam sistem informasi change request dan prediksi priority	Merancang sistem <i>change request</i> berbasis web dan menerapkan ML untuk proses change management yang terstruktur
(Kurniawan & Sutabri, 2023)	Analisis ITSM pada Sistem Informasi	COBIT 5	Penelitian ini berfokus pada analisis tingkat kematangan ITSM menggunakan COBIT 5 dan rekomendasi perbaikan, belum pada implementasi sistem.	Menghasilkan sistem IT Change Request serta fitur prediksi priority ke dalam sistem berbasis web untuk membantu pengelolaan perubahan secara terstruktur
(Yuliana et al., 2021)	Manajemen Perubahan terhadap	<i>ANP (Analytic Network Process)</i>	Penelitian berfokus pada analisis manajemen perubahan terhadap kinerja organisasi	Merancang sistem <i>change request</i> yang secara langsung

	kinerja Organisasi		menggunakan ANP dan belum menghasilkan sistem change request.	menerapkan ML dalam format berbasis <i>web</i>
--	-----------------------	--	---	---

Berdasarkan tinjauan terhadap studi-studi sebelumnya, penelitian mengenai *change request*, ITSM, *change management* telah mencakup beragam topik, mulai dari pemantauan *Engineering Change Request*, dan pengembangan *Change Request Form (CRF)* hingga analisis dan rekomendasi terkait ITSM, serta analisis penerapan *change management* di dalam organisasi. Beberapa studi telah menghasilkan sistem *change request*, sementara yang lain menghasilkan analisis, rekomendasi, atau penerapan konsep *change management* tanpa berfokus pada desain sistem *IT Change Request* yang disesuaikan dengan kebutuhan *Departement IT & System*.

Berdasarkan perbandingan studi-studi sebelumnya, penelitian yang ditinjau memiliki fokus yang beragam, seperti engineering change requests, change request form, IT service management (ITSM), dan change management. Namun, penelitian yang ditinjau belum mengintegrasikan proses IT Change Request sesuai dengan alur kerja Departement IT & System di PT Agincourt Resources ke dalam satu sistem terpadu. Selain itu, penelitian yang ditinjau belum mengidentifikasi penerapan Machine Learning untuk memberikan prediksi awal mengenai prioritas perubahan berdasarkan informasi teks dari change request.

Dengan demikian, penelitian ini bertujuan untuk membahas kesenjangan dengan merancang sistem informasi *IT Change Request* berbasis ASP.NET Core yang mengintegrasikan alur kerja *change management* sesuai dengan kebutuhan *Departement IT & System di PT Agincourt Resources*. Sistem ini dirancang sebagai solusi operasional yang mampu mendukung proses manajemen *IT Change Request* yang terstruktur dan terpusat.

2.15. State Of The Art

State of the Art adalah posisi dan kontribusi penelitian ini dibandingkan dengan studi-studi sebelumnya. Bagian ini menunjukkan bagaimana penelitian ini memberikan pembaruan atau kontribusi baru terhadap solusi yang sudah ada.

Tabel 2. 6 State of the Art

Peneliti	Fokus Penelitian	Teknologi	Kontribusi Penelitian Sebelumnya	Kebaruan Penelitian Ini
(Fernandes, 2022)	<i>Monitoring Engineering Change Request</i>	<i>Web-Based</i>	Berhasil mengembangkan sistem pemantauan perubahan berbasis <i>web</i>	Merancang workflow IT Change Request yang disesuaikan dengan kebutuhan Departement IT & System, mulai dari pengajuan hingga closure, serta menerapkan Machine Learning untuk memberikan prediksi awal change priority.
(Rufiyadi et al., 2023)	Sistem <i>Change Request Form (CRF)</i>	<i>V-Model</i>	Berhasil mengembangkan sistem pengajuan <i>change request</i> secara digital	Menerapkan pengelolaan IT Change Request dengan role-based access, pemantauan

				status, notifikasi serta menambahkan prediksi change priority menggunakan multilingual embedding model dan SVM.
(Meisha Putri et al., 2024)	Implementasi ITSM	ITIL V3	Berhasil menganalisis dan memetakan proses ITSM di dalam organisasi	Mengimplementasikan pengelolaan IT Change Request dalam sistem berbasis web berdasarkan kebutuhan proses perubahan pada objek penelitian serta menerapkan ML untuk prediksi awal change priority.
(Syandri Pratama, 2022)	Implementasi <i>Change Management</i>	Model 8 Tahap	Menerapkan change management menggunakan model Kotter Eight Stage Change Process dalam organisasi.	Mengimplementasikan proses pengelolaan perubahan ke dalam sistem IT Change Request berbasis web sesuai kebutuhan organisasi serta

				menambahkan prediksi change priority.
(Kurniawan & Sutabri, 2023)	Analisis ITSM	COBIT 5	Berhasil mengevaluasi tingkat kematangan ITSM sistem informasi	Menerapkan pengelolaan IT Change Request dalam sistem berbasis web dengan alur yang terstruktur dan terdokumentasi dan menambahkan fitur prediksi change priority berbasis ML
(Yuliana et al., 2021)	Manajemen Perubahan terhadap Kinerja Organisasi	<i>ANP (Analytic Network Process)</i>	Berhasil melakukan analisis dampak <i>change management</i> terhadap kinerja organisasi	Menerapkan konsep <i>change management</i> ke dalam sistem <i>change request</i> berbasis web serta mengembangkan fitur prediksi awal change priority menggunakan ML

Berdasarkan penelitian sebelumnya, telah ada studi yang membahas dan mengembangkan sistem terkait *Engineering Change Request* dan *Change Request Form*, serta studi yang mengkaji ITSM dan *Change Management* dari sudut pandang analisis, rekomendasi, dan implementasi di dalam organisasi. Studi-studi sebelumnya ini memberikan landasan bagi pengembangan sistem dan penerapan konsep *change management*, tetapi prediksi *change priority* menggunakan *Machine Learning* belum menjadi fokus pada penelitian yang ditinjau namun berfokus pada pokok bahasan dan pendekatannya berbeda dari kebutuhan *IT Change Request* yang dimiliki oleh *Departement IT & System di PT Agincourt Resources*.

Penelitian ini bukan hanya berfokus dalam mengembangkan *IT Change Request*, namun juga menerapkan *machine learning* untuk memberikan prediksi awal mengenai *change priority*. Prediksi tersebut dibuat dengan menggunakan *multilingual embedding model* dan algoritma *Support Vector Machine (SVM)* berdasarkan informasi tekstual dari dataset jira.

Penelitian ini memberikan kontribusi dengan merancang sistem *IT Change Request* berbasis ASP.NET Core menggunakan arsitektur MVC yang disesuaikan dengan alur kerja *Departement IT & System*. Sistem ini mencakup proses pengajuan, *preliminary review*, *assessment*, *consolidation*, *approval*, *implementation*, *post implementation review (PIR)*, dan *Closure*. Selain itu, sistem ini menerapkan manajemen akses berbasis *role*, *notification*, dan *monitoring* terpusat terhadap status perubahan.

Dengan demikian, kebaruan penelitian ini terletak pada perancangan sistem *IT Change Request* yang menerapkan alur kerja *change management* yang disesuaikan dengan kebutuhan

Departement IT & System di PT Agincourt Resources, sehingga berbagai proses yang biasanya dilakukan pada platform terpisah menjadi lebih terstruktur dan dapat dikelola dalam satu sistem berbasis web.

2.16. Novelty Penelitian

Kebaruan dari penelitian ini terletak pada perancangan sistem informasi *IT Change Request* berbasis web yang menggabungkan semua proses *change management* ke dalam satu sistem. Sistem yang dikembangkan tidak hanya mendukung proses pengajuan *change request*, tetapi juga mengelola tahapan *preliminary review, assessment, consolidation, approval, implementation, post implementation review (PIR)*, hingga proses *closure* sesuai alur kerja yang diimplementasikan di *Departement IT & System PT Agincourt Resources*.

Selain itu, sistem ini menggunakan metode *role-based access* sehingga masing-masing pengguna hanya dapat mengakses menu dan menjalankan proses sesuai dengan tugas dan tanggung jawabnya. Sistem ini juga dilengkapi dengan *notifikasi* otomatis guna setiap perubahan status permintaan dan *monitoring* status secara *real-time*, sehingga semua pihak yang terlibat dapat memantau perkembangan proses tanpa harus melakukan koordinasi secara manual.

Kebaruan lainnya terletak pada penerapan ML untuk memberikan prediksi awal mengenai *change priority*. Prediksi ini dibuat dengan menggabungkan multilingual embedding model yang merepresentasikan informasi tekstual dalam *change request* dengan algoritma Support Vector Machine (SVM) untuk mengklasifikasikan *change priority* di kategori low, medium, dan high. Informasi tekstual yang digunakan terdiri dari *change title, description of change, dan reason for change*.

Berbeda dengan penelitian sebelumnya yang berfokus pada pemantauan Engineering Change Request, pengembangan Change Request Form, analisis dan memberikan rekomendasi ITSM, serta penerapan change management, penelitian ini menghasilkan sebuah sistem yang dapat diterapkan secara operasional untuk mendukung proses IT Change Request dalam lingkungan perusahaan. Dengan demikian, penelitian ini berkontribusi pada implementasi sistem informasi yang mampu mengintegrasikan seluruh proses *IT Change Request* secara terstruktur dan terdokumentasi, sesuai dengan kebutuhan operasional *Departement IT & System di PT Agincourt Resources*.

Berdasarkan pembahasan di atas, kebaruan yang diberikan oleh penelitian ini dapat dirangkum sebagai berikut:

Tabel 2. 7 Novelty Penelitian

Aspek	Penelitian Sebelumnya	Penelitian ini
Workflow	Memiliki focus dan cakupan workflow yang berbeda sesuai objek penelitian	Mengintegrasikan Seluruh workflow mulai dari submit hingga closure
Hak akses	Pengelolaan penggunaan hak akses disesuaikan dengan fokus masing-masing penelitian	Menerapkan Role-based access sesuai struktur organisasi
Monitoring	Monitoring status masih dikembangkan sesuai fokus masing-masing penelitian	Pemantauan status pada setiap tahapan proses

Notifikasi	Belum menjadi fokus utama pada penelitian yang dikaji	Notifikasi otomatis pada perubahan status request
Implementasi	Terdapat penelitian yang menghasilkan sistem, sementara penelitian lainnya berfokus pada analisis, rekomendasi atau penerapan change management	Sistem operasional berbasis web di PT Agincourt Resources

2.17. Justifikasi Penggunaan Dataset Alternatif

Data historis *IT Change Request* dari perusahaan tersebut belum tersedia dalam format terpusat dan oleh karena itu tidak dapat digunakan sebagai dataset pelatihan ML. Sehingga penelitian ini menggunakan dataset *Jira public* sebagai sumber data alternatif. Penggunaan dataset ini didukung oleh penelitian sebelumnya yang telah memanfaatkan dataset public atau data Jira dalam studi yang berkaitan dengan *software engineering* dan ML. alasan yang didasarkan pada penelitian sebelumnya dapat dilihat pada Tabel 2.8.

Tabel 2. 8 Justifikasi Penggunaan Dataset Alternatif

Penelitian	Dataset	Metode	Justifikasi
(Li et al. 2022)	4.200 issue dari 7 proyek open-source menggunakan Jira dan Google Monorail	Supervised ML untuk klasifikasi Self-Admitted Technical Debt (SATD) pada issue, dengan pendekatan CNN	Menunjukkan bahwa data issue dari issue tracking system yang tersedia secara public dapat dikumpulkan dan digunakan sebagai dataset untuk penelitian ML.
(Lüders et al. 2023)	16 public JIRA repositories	ML menggunakan model transformer (BERT) untuk memprediksi issue links berdasarkan title dan description	Menunjukkan penggunaan dataset JIRA public dan informasi tekstual issue sebagai data untuk pengembangan model ML.

(Kondo et al. 2022)	5 proyek open-source Apache dengan repository GitHub dan issue reports pada JIRA	Evaluasi 10 Issue-Link Algoritma (ILA) untuk mendeteksi defect-fixing commits dan mengevaluasi pengaruhnya terhadap just-in-time defect prediction, pendekatan menggunakan Random Forest dan SVM	Menunjukkan data issue dari project open-source berupa repository GitHub dan issue reports pada JIRA dapat digunakan sebagai data penelitian untuk evaluasi pendekatan issue-link dan just-in-time defect prediction.
(Pasuksmit et al. 2022)	19.349 work items dari 7 proyek open-source yang menggunakan JIRA	ML menggunakan classifier untuk memprediksi perubahan Story Points pada work item, dengan Logistic Regression sebagai classifier utama dan dibandingkan dengan Random Forest, SVM, dan CART	Menunjukkan data work item dari proyek open-source yang menggunakan JIRA dapat digunakan sebagai data historis guna pengembangan model prediksi.
(Ahmad and Kuo 2025)	21.064 issue dari 14 proyek open-source menggunakan JIRA memiliki issue key, title, description, dan story point	Ensemble stacking menggunakan RoBERTa dan BiLSTM untuk estimasi Story Points	Menunjukkan dataset open-source yang tersedia secara public dan berasal dari JIRA dapat digunakan untuk pengembangan dan evaluasi model ML pada data issue yang

			memiliki informasi seperti title dan description.
(Oyetoyan and Morrison 2021)	Dataset dari beberapa sumber, salah satunya data issue dari proyek open-source Apache JIRA yang diberi label security-related	Text classification untuk mengidentifikasi security-related messages menggunakan SVM, Random Forest, Naïve Bayes, Logistic Regression, dan K-Nearest Neighbour	Menunjukkan data issue dari proyek open-source pada JIRA dapat digunakan sebagai data penelitian untuk membangun dan mengevaluasi model text classification berbasis ML.
(Colavito et al. 2024)	Dataset issue public dari Github dan JIRA, dataset Jira berasal dari 16 public Jira repositories	Pre-trained language models untuk automatic issue classification, dengan fine-tuning BERT, ALBERT, dan RoBERTa	Menunjukkan dataset issue yang tersedia secara public dari GitHub dan Jira dapat digunakan untuk melatih dan mengevaluasi model Machine Learning untuk klasifikasi issue berdasarkan informasi tekstual issue.
(Saleem et al. 2023)	Dataset bug report Eclipse dan Mozilla	Model bug prioritization menggunakan Average One Dependence Estimator (AODE), dengan perbandingan terhadap SVM dan Naïve Bayes	Menunjukkan bahwa data bug report dapat digunakan untuk membangun dan mengevaluasi model ML dalam melakukan prediksi atau penentuan prioritas bug.

(Van Oosten et al. 2023)	JIRA issues yang merepresentasikan requirements dan bugs dari 7 proyek MDD	ML untuk trace link recovery antara perubahan model dan JIRA issues menggunakan Random Forest, XGBoost, dan LightGBM	Menunjukkan bahwa JIRA issues dapat digunakan sebagai data dalam penelitian untuk penerapan ML pada trace link recovery.
(Sutoyo et al. 2026)	Jira issues dari 10 proyek open-source, dengan lebih dari 103.000 kandidat issue	Keyword-based filtering, active learning dengan model BERT, serta Explainable AI menggunakan LIME dan SHAP untuk deteksi Architecture Technical Debt (ATD)	Menunjukkan Jira issues dari proyek open-source dapat digunakan sebagai sumber data untuk membangun dataset dan mengembangkan model ML pada penelitian Software Engineering.