BAB 2
TINJAUAN PUSTAKA

2.1 Perusahaan PT Padepokan Tujuh Sembilan

PT Padepokan Tujuh Sembilan, atau yang dikenal sebagai Perusahaan 79 merupakan perusahaan yang menyediakan sumber daya manusia berkeahlian di teknologi informasi yang berkualitas agar siap terlibat dalam layanan teknologi informasi. Beralamat di Gang. Terasana No.6A, Pasir Kaliki, Kecamatan Cicendo, Kota Bandung. Perusahaan 79 didirikan pada tahun 2010 sebagai tanggapan terhadap kebutuhan yang berkembang dalam industri Teknologi Informasi (TI). Sebuah inisiatif dari kelompok individu yang berdedikasi untuk memberikan layanan teknologi informasi dan pada saat yang sama, mengembangkan potensi sumber daya manusia di bidang tersebut. Dengan komitmen untuk menyediakan bakat terampil dalam layanan TI, Perusahaan 79 bertujuan untuk membangun dan meningkatkan kemampuan individu dengan keterampilan dan pengetahuan khusus.

Selain layanan utama di bidang teknologi informasi, Perusahaan 79 menyediakan *Onsite Placement* untuk memenuhi kebutuhan klien dengan menyediakan talenta TI berkualitas dalam berbagai bidang. Dalam pengembangan perangkat lunak juga, Perusahaan 79 menekankan penggunaan teknologi terkini dan semangat untuk memberikan nilai tambah dalam setiap proyek klien. Perusahaan 79 juga menekankan pada pengembangan bakat melalui program *Bootcamp*, di mana talenta muda diberikan keterampilan teknis dan interpersonal selama 3 bulan. Program ini bertujuan agar para talenta siap terlibat dalam proyek TI yang kompleks. Dengan, Perusahaan 79 terus berinovasi dan menyesuaikan diri dengan perkembangan industri TI. Dengan semangat kolaborasi, pemberdayaan sumber daya manusia, dan fokus pada kualitas, Perusahaan 79 terus berperan dalam membentuk masa depan teknologi informasi di Indonesia.

2.2 Visi dan Misi PT Padepokan Tujuh Sembilan

Visi dan Misi PT Padepokan Tujuh Sembilan bisa di rincikan sebagai berikut :

1. Visi

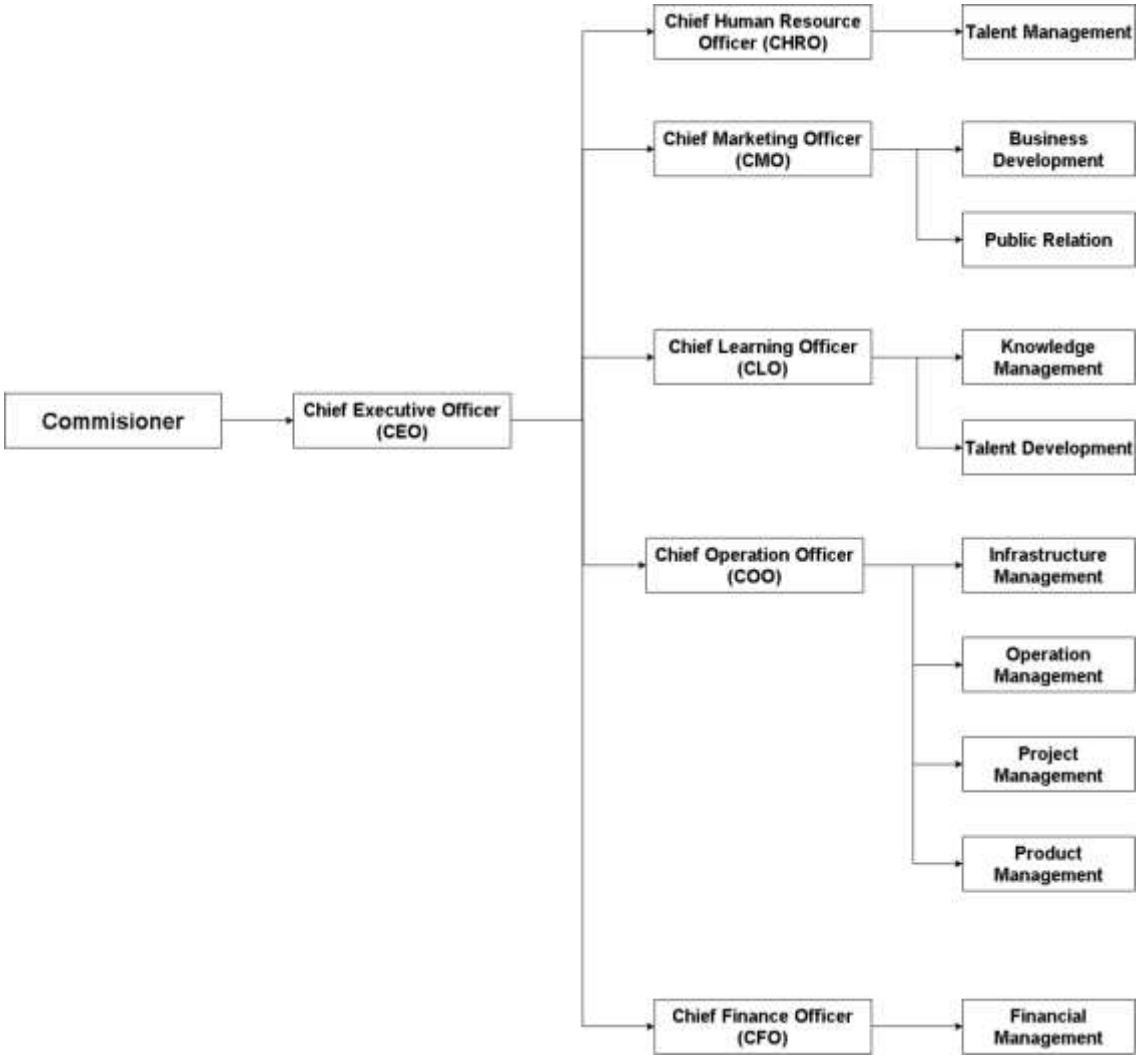
   Visi Perusahaan 79 mencakup ambisi untuk menjadi penyedia sumber daya manusia dan outsourcing TI terbesar di Indonesia pada tahun 2025.

2. Misi

   Misi Perusahaan 79 berkomitmen untuk memberikan solusi perangkat lunak profesional, berkualitas tinggi, andal, dan efisien, dengan tetap memperhatikan kemudahan penggunaan untuk semua pelanggan.

2.3 Struktur Organisasi PT Padepokan Tujuh Sembilan

Untuk kelancaran dan keberhasilan suatu , maka perlu dibentuk struktur organisasi dengan tujuan agar dapat terlaksananya tugas dengan lancar dan baik. Struktur organisasi merupakan suatu cara di mana tanggung jawab dan tugas didelegasikan kepada individu. Individu tersebut dikelompokkan berdasarkan tugas yang dibebankan. Agar tercipta kinerja organisasi dengan proses kerja yang cepat dan efektif [10]. Perusahaan 79 juga memiliki struktur organisasi sendiri yaitu sebagaimana yang di tampilkan pada gambar 2.1 di bawah ini.



Gambar 2.1 Struktur Organisasi Perusahaan 79

Keterangan dari masing-masing bagian berdasarkan Gambar 2.1 sebagai berikut :

1. ***Commisioner***

Commisioner atau Komisaris memiliki peran pengawasan dan penasehat. Komisaris pada perusahaan ini bertanggung jawab mengawasi kebijakan, memberikan nasihat kepada dewan direksi, serta terlibat dalam pemilihan dan evaluasi direksi. Komisaris juga dapat membantu menilai kinerja perusahaan, memastikan kepatuhan terhadap regulasi, dan mengelola risiko.

2. ***Chief Executive Officer*** **(CEO)**

Chief Executive Officer (CEO) di Perusahaan 79 memiliki peran utama adalah memimpin secara strategis untuk mencapai tujuan perusahaan. Bertanggung jawab atas pengambilan keputusan kunci, manajemen operasional, dan pengembangan bisnis, CEO juga memiliki peran penting dalam membangun budaya perusahaan yang kuat.

3. ***Chief Human Resource Officer*** **(CHRO)**

Chief Human Resources Officer (CHRO) di Perusahaan 79 mempunyai peran dalam mengelola aspek manusia perusahaan. CHRO bertanggung jawab atas strategi sumber daya manusia, termasuk perekrutan, pengembangan karyawan, manajemen kinerja, dan kebijakan karyawan. Selain itu, CHRO menaungi 1 divisi yaitu *Talent Management* yang mempunyai fungsi sebagai berikut :

a. *Talent Management*

   Divisi *Talent Management* di bawah naungan CHRO di Perusahaan 79 (PT Padepokan Tujuh Sembilan) memiliki peran kunci dalam mengelola siklus hidup karyawan. Fokusnya termasuk perekrutan, pengembangan, retensi, dan pengelolaan kinerja talenta IT di perusahaan 79. Divisi Talent Management menjadi dalam menciptakan lingkungan kerja yang produktif dan memastikan sumber daya manusia perusahaan sejalan dengan strategi dan tujuan organisasi.

4. ***Chief Marketing Officer*** **(CMO)**

Sebagai *Chief Marketing Officer* (CMO) di Perusahaan 79 (PT Padepokan Tujuh Sembilan), mempunyai peran dalam mengelola strategi pemasaran dan promosi perusahaan. CMO bertanggung jawab atas pengembangan dan pelaksanaan kampanye pemasaran yang efektif, branding perusahaan, serta

hubungan dengan pelanggan. Fokusnya mencakup identifikasi peluang pasar, peningkatan visibilitas merek, dan pengelolaan komunikasi pemasaran. Selain itu CMO menaungi divisi yang berkaitan dengan fungsinya, yang diantaranya :

a. *Business Development*

   Divisi *Business Development* di bawah naungan CMO di Perusahaan 79 memiliki tanggung jawab mengidentifikasi peluang pertumbuhan bisnis baru, menjalin kemitraan strategis, dan merancang inisiatif bisnis yang mendukung strategi pemasaran dan branding. Divisi ini berfokus pada pengembangan pasar dan peningkatan pendapatan melalui ekspansi bisnis.

b. *Public Relation*

   Divisi *Public Relation* yang di bawah CMO bertugas membangun dan memelihara citra positif perusahaan melalui strategi komunikasi eksternal yang efektif. Divisi ini merancang kampanye promosi, menanggapi media, dan menjalankan kegiatan-kegiatan yang meningkatkan visibilitas dan reputasi perusahaan. Keduanya bekerja sama untuk memastikan bahwa pertumbuhan bisnis tidak hanya didukung oleh strategi pemasaran yang kuat tetapi juga oleh hubungan positif dengan pemangku kepentingan dan masyarakat umum.

5. ***Chief Learning Officer*** **(CLO)**

Chief Learning Officer (CLO) di Perusahaan 79 memastikan bahwa karyawan memanfaatkan teknologi terkini melalui strategi pembelajaran yang efektif. CLO bertanggung jawab untuk merancang program pelatihan yang memungkinkan karyawan memahami, menguasai, dan menerapkan teknologi terbaru dalam pekerjaan mereka. Dengan menyediakan pelatihan yang relevan dan mendukung, CLO berperan dalam meningkatkan literasi teknologi karyawan, mendorong adopsi teknologi, dan menciptakan budaya pembelajaran yang menjadikan penggunaan teknologi sebagai bagian integral pekerjaan. Penelitian ini

sendiri akan digunakan oleh divisi ini, CLO sendiri memiliki 2 divisi yang dinaungi. yaitu :

a. *Knowledge Management*

   Divisi *Knowledge Management* yang di bawah naungan CLO di Perusahaan 79 memiliki tanggung jawab mengelola riset teknologi terkini, memastikan akses yang efisien ke informasi terbaru, serta mendorong penggunaan pengetahuan tersebut dalam proyek-proyek perusahaan.

b. *Talent Development*

   Divisi *Talent Development* berfokus pada pengembangan keterampilan dan kompetensi karyawan sebelum terlibat dalam proyek tertentu, memastikan bahwa tim memiliki bakat yang terampil dan siap untuk menghadapi tuntutan proyek dengan kualitas terbaik. Divisi ini juga akan menjadi pengguna dalam penelitian ini dan akan berperan menjadi Tim Pengembang.

6. ***Chief Operation Officer* (COO)**

Chief Operating Officer (COO) di Perusahaan 79 memiliki peran untuk mengelola operasional sehari-hari perusahaan. COO bertanggung jawab atas efisiensi dan kohesi keseluruhan operasional perusahaan, mendukung pertumbuhan, dan memastikan pencapaian tujuan bisnis. Selain itu COO memiliki divisi untuk mendukung operasional perusahaan, divisi itu diantaranya :

a. Infrastructure Management

   Divisi *Infrastructure Management* memiliki fokus utama pada pengelolaan dan pemeliharaan aspek krusial seperti jaringan, server, dan sistem basis data. Tugas utama mereka adalah memastikan ketersediaan dan keandalan infrastruktur guna mendukung operasional perusahaan dengan lancar. Selain itu, mereka juga bertanggung jawab atas keamanan dan skalabilitas sistem teknologi, menjadikan keberlanjutan dan keamanan operasional sebagai prioritas dalam menjaga performa optimal perusahaan.

b. *Operation Management*

Divisi *Operation Management* di Perusahaan 79 bertanggung jawab penuh dalam mengelola operasional sehari-hari, melibatkan aspek seperti produksi, pengelolaan persediaan, dan distribusi barang atau layanan. Mereka memastikan efisiensi proses operasional dan kepatuhan terhadap standar kerja yang telah ditetapkan, sambil menangani perencanaan kapasitas, pemantauan kinerja, serta melakukan perbaikan dan peningkatan terus-menerus pada proses untuk meningkatkan produktivitas dan kesesuaian dengan tujuan perusahaan.

c. *Project Management*

Divisi *Project Management* di Perusahaan 79 memiliki peran integral dalam mengelola proyek-proyek dari tahap perencanaan hingga implementasi. Tugas utama mereka mencakup penentuan tujuan proyek, alokasi sumber daya, dan pemantauan kemajuan secara menyeluruh. Dengan memastikan proyek selesai tepat waktu, sesuai dengan anggaran yang telah ditetapkan, dan sesuai dengan spesifikasi yang diberikan, divisi ini berperan krusial dalam memastikan keberhasilan dan kualitas pelaksanaan proyek yang mendukung pencapaian tujuan perusahaan secara keseluruhan.

d. *Product Management*

Divisi *Product Management* di Perusahaan 79 memegang tanggung jawab penuh atas siklus hidup produk, mulai dari perencanaan hingga pengembangan dan pemasaran. Mereka aktif mengidentifikasi peluang pasar, menentukan fitur produk, dan berkoordinasi dengan tim pengembangan untuk memastikan implementasi yang sukses. Melalui peran ini, divisi ini tidak hanya memastikan kesesuaian produk dengan strategi perusahaan, tetapi juga berupaya memenuhi kebutuhan dan harapan pelanggan.

7. ***Chief Finance Officer*** **(CFO)**

Chief Financial Officer (CFO) di Perusahaan 79 memiliki peran kunci dalam mengelola aspek keuangan perusahaan. Tugas utamanya meliputi perencanaan keuangan, pengelolaan risiko keuangan, dan pelaporan keuangan yang akurat. CFO juga bertanggung jawab atas pengelolaan kas, pemantauan investasi, serta berkolaborasi dengan berbagai departemen untuk memastikan keuangan perusahaan sehat dan sesuai dengan tujuan strategis. CFO juga menaungi divisi yang berkaitan, yang diantaranya :

a. *Financial Management*

   Divisi *Financial Management* di bawah naungan CFO di Perusahaan 79 memiliki tanggung jawab utama dalam mengelola aspek keuangan perusahaan. Fokusnya meliputi perencanaan keuangan, pengelolaan anggaran, pemantauan cash flow, dan penyusunan laporan keuangan yang akurat dan terkini. Divisi ini juga berperan dalam analisis risiko keuangan, pengelolaan investasi, serta memastikan kepatuhan terhadap regulasi keuangan yang berlaku.

2.4 Proyek

Proyek adalah suatu usaha terorganisir yang bersifat sementara dan memiliki batasan waktu untuk mencapai tujuan tertentu. Proyek melibatkan pengelolaan sumber daya manusia, keuangan, dan materi dengan tujuan menghasilkan hasil unik atau produk tertentu . Proses proyek mencakup perencanaan, pelaksanaan, dan pengendalian untuk memastikan bahwa sasaran yang ditetapkan dapat dicapai sesuai dengan waktu dan anggaran yang telah ditentukan [1].

2.5 Pemantauan Proyek

Pemantauan proyek adalah proses sistematis untuk mengamati, mengevaluasi, dan melacak kemajuan serta kinerja proyek secara berkala.

Pemantauan proyek dalam manajemen proyek memungkinkan manajer proyek dan para pemangku kepentingan untuk mengidentifikasi dan menangani setiap penyimpangan atau risiko dengan cepat, sehingga meningkatkan kesuksesan proyek [11]. Dari definisi diatas terdapat tujuan, jenis – jenis, dan proses dalam melakukan pemantauan proyek yang diantaranya :

2.5.1 Tujuan Monitoring Proyek

Tujuan utama pemantauan proyek mendeteksi dan mencegah permasalahan yang akan muncul, meningkatkan pengambilan keputusan, meningkatkan akuntabilitas di antara anggota tim, pengalokasian sumber daya yang efektif, dan juga memudahkan pemangku kepentingan tentang kemajuan suatu proyek [12]. Proses ini melibatkan pemantauan berbagai aspek proyek, termasuk pelaksanaan tugas, penggunaan sumber daya, kepatuhan terhadap jadwal dan anggaran, serta pencapaian tujuan proyek.

2.5.2 Jenis – jenis Monitoring Proyek

Monitoring proyek sendiri memiliki beberapa jenis yang meliputi [12]:

1. Pemantauan Kemajuan

   Jenis monitoring proyek ini berfokus pada pelacakan kemajuan proyek terhadap jadwal yang telah direncanakan, pencapaian yang akan diraih, dan hasil yang sudah direncanakan. Ini memastikan bahwa proyek tetap berjalan sesuai rencana dan memungkinkan penyesuaian yang tepat waktu jika terjadi masalah di masa depan.
2. Pemantauan Kualitas

   Jenis pemantauan proyek ini melibatkan penilaian dan pengukuran kualitas hasil proyek terhadap standar dan menjadi tolak ukur yang telah ditetapkan sebelumnya. Hal ini akan memastikan bahwa proyek memenuhi kriteria

kualitas yang dibutuhkan dan memfasilitasi tindakan korektif yang tepat waktu.

3. Pemantauan Risiko

   Jenis monitoring proyek ini melibatkan identifikasi, penilaian, dan pemantauan risiko proyek sepanjang siklus hidup proyek. Hal ini akan membantu dalam manajemen risiko proaktif dengan mengidentifikasi risiko potensial dan mengembangkan strategi mitigasi untuk meminimalkan dampaknya pada hasil proyek.

4. Pemantauan Biaya

   Jenis pemantauan proyek ini melibatkan pelacakan dan pengendalian pengeluaran proyek terhadap anggaran yang dialokasikan. Hal ini akan membantu mencegah keterlambatan biaya, memastikan stabilitas keuangan, dan mengoptimalkan pemanfaatan sumber daya.

5. Pemantauan Kinerja

   Jenis pemantauan proyek ini akan berfokus pada evaluasi kinerja anggota tim proyek, pemasok, dan kontraktor. Hal ini akan melibatkan pelacakan metrik kinerja individu dan kolektif untuk memastikan keselarasan dengan tujuan dan target proyek.

2.5.3 Proses Monitoring Proyek

Dalam pemantauan proyek akan banyak melibatkan proses agar proyek tetap sesuai dan rencana dan tujuan proyek dapat tercapai, proses – proses ini melibatkan yang diantaranya [12]:

1. Penetapan Tujuan

   Proses ini akan menetapkan tujuan dan indikator kinerja utama (KPI) yang akan dipantau sepanjang proyek. Tujuan ini harus sejalan dengan tujuan dan pencapaian proyek.

2. Pemerolehan Informasi

   Proses ini akan melibatkan pencarian data yang relevan serta informasi tentang status pekerjaan, aktivitas, dan kinerja.

3. Evaluasi

   Proses ini akan mengevaluasi data yang terkumpul untuk mengukur keberhasilan proyek dan mengidentifikasi setiap variasi dari tujuan yang telah ditetapkan.

4. Pengambilan Tindakan

   Proses ini melibatkan penyelesaian masala yang ditemukan untuk memberikan gambaran dan penilaian.

5. Monitoring Berkala

   Proses ini akan terus memantau kemajuan proyek, kinerja, dan risiko untuk memastikan bahwa proyek tetap berada pada rencana dan sejalan dengan tujuan yang telah ditetapkan.

2.6 Scrum

Scrum merupakan suatu kerangka kerja pengembangan perangkat lunak yang menekankan kolaborasi tim, komunikasi yang terbuka, dan adaptabilitas. Dalam metodologi ini, Proyek yang menerapkan metodologi Scrum mengalami perkembangan melalui serangkaian proses yang berdurasi berkisar tiga puluh hari yang disebut *sprint*. Di awal setiap *sprint*, tim mengevaluasi jumlah pekerjaan yang dapat diselesaikan selama periode tersebut. Tugas-tugas dipilih dari daftar prioritas yang disebut *project backlog*. Pekerjaan yang terselesaikan selama sprint dipindahkan ke dalam daftar yang disebut *sprint backlog*. Selain itu, dilakukan pertemuan harian singkat yang disebut *daily scrum*, yang memungkinkan tim untuk memeriksa kemajuan mereka dan beradaptasi sesuai kebutuhan [13]. Adapun komponen – komponen Scrum serta pelakunya yang diantaranya :

2.6.1 Project Backlog

Project backlog adalah daftar lengkap dari semua pekerjaan yang harus diselesaikan dalam suatu proyek. *Project Backlog* mencakup fitur-fitur baru,

perbaikan-perbaikan pada sistem yang sudah ada, hingga tugas-tugas teknis yang diperlukan. *Backlog* ini biasanya disusun dalam urutan prioritas, dengan item-item yang paling penting atau mendesak diletakkan di bagian atas. Selain itu, project backlog juga dapat berkembang seiring waktu, dengan penambahan item baru atau perubahan prioritas berdasarkan perubahan kebutuhan atau pemahaman yang lebih baik tentang proyek tersebut [13].

2.6.2 User Stories

User stories merupakan alat yang digunakan dalam pengembangan perangkat lunak untuk mengekspresikan kebutuhan pengguna dari sudut pandang fungsionalitas yang diinginkan. *User Stories* biasanya ditulis dalam format naratif yang sederhana dan mudah dimengerti, sering kali berbentuk frasa atau kalimat pendek yang menjelaskan tindakan atau tujuan yang diinginkan oleh pengguna. *User Strories* membantu tim pengembangan memahami kebutuhan pengguna secara jelas dan memungkinkan untuk fokus pada pengiriman nilai yang sesuai dengan harapan pengguna [13].

2.6.3 Sprint Backlog

Sprint backlog adalah daftar dari tugas-tugas atau pekerjaan yang harus diselesaikan oleh tim pengembangan selama sprint dalam metodologi Scrum. Sprint backlog merupakan bagian yang terfokus dari backlog produk yang dipilih oleh tim pengembangan untuk dipecah menjadi tugas-tugas yang dapat diselesaikan dalam periode waktu yang telah ditentukan. Sprint backlog terdiri dari item-item yang dipilih dengan hati-hati dari backlog produk dan diuraikan menjadi tugas-tugas yang lebih spesifik dan terukur [13].

2.6.4 Daily Scrum

Daily Scrum adalah pertemuan harian yang dilakukan oleh tim pengembang dalam metodologi Scrum. Pertemuan ini biasanya dilakukan setiap hari pada waktu yang sama dan berlangsung secara singkat, berkisar 15 menit atau kurang. Tujuan dari *Daily Scrum* dilaksanakan yaitu untuk memungkinkan tim untuk berkolaborasi, memperbarui tentang kemajuan pekerjaan, serta mengidentifikasi dan mengatasi hambatan atau masalah yang mungkin muncul dalam pencapaian tujuan sprint [13].

2.6.5 Sprint Retrospective

Sprint retrospective adalah pertemuan reguler yang diadakan oleh tim Scrum setelah setiap sprint untuk mengevaluasi kinerja mereka, menganalisis cara mereka bekerja, mengidentifikasi peluang perbaikan, dan merencanakan tindakan untuk menerapkan perbaikan tersebut. *Sprint retrospective* digunakan sebagai kesempatan bagi tim untuk secara terbuka meninjau dan membahas semua aspek dari proses pengembangan, termasuk praktik, komunikasi, dan lingkungan kerja. *Sprint retrospective* merupakan proses yang penting dalam Scrum karena memberikan kesempatan untuk perbaikan yang berkelanjutan dan penyesuaian terhadap kebutuhan dan tantangan yang dihadapi oleh tim.

2.6.6 Pelaku Scrum

Pelaku Scrum adalah individu-individu yang bertanggung jawab untuk memastikan pengembangan produk menggunakan metodologi Scrum. Mereka terdiri dari Product Owner, Scrum Master, dan Tim Pengembang. Pelaku Scrum bekerja bersama-sama untuk mengoptimalkan proses pengembangan dengan menggunakan pendekatan kolaboratif, iteratif, dan responsif yang dianut oleh metodologi Scrum [13]. Dan berikut ini adalah fungsi dan peranan dari masing – masing peran pelaku Scrum.

1. *Product Owner*

Product owner adalah otoritas tunggal yang bertanggung jawab atas keputusan tentang fitur dan fungsionalitas produk yang akan dibangun, serta memastikan kesuksesan keseluruhan dari solusi yang sedang dikembangkan atau dipelihara. Mereka bertanggung jawab untuk menjaga visi produk, berkolaborasi dengan tim pengembangan, dan memastikan pekerjaan yang paling bernilai selalu dilakukan.

2. Scrum Master

Scrum Master bertindak sebagai fasilitator yang membantu semua pihak terlibat dalam penggunaan dan pemahaman nilai, prinsip, dan praktik-praktik Scrum. Scrum Master berperan penting dalam mengembangkan pendekatan Scrum yang sesuai dengan kebutuhan organisasi, dan memfasilitasi pemecahan masalah tim. Scrum Master tidak memiliki wewenang untuk mengendalikan tim, tetapi bertanggung jawab untuk memimpin dan mendukung tim dalam menerapkan praktik Scrum yang efektif.

3. Tim Pengembang

Tim pengembang merupakan sebagai kumpulan individu yang beragam dan lintas-fungsional, seperti arsitek, programmer, pengujian, administrator basis data, desainer UI, dan lain-lain. Tugas utama tim pengembang adalah merancang, membangun, dan menguji produk yang diinginkan. Ukuran tim pengembang biasanya antara lima hingga sembilan orang, dengan setiap anggota memiliki keterampilan yang berbeda dan diperlukan untuk menghasilkan perangkat lunak berkualitas tinggi. Jika diperlukan, Scrum juga dapat digunakan untuk tim pengembangan yang lebih besar dengan cara memiliki beberapa tim Scrum dengan anggota masing-masing tim kurang dari sembilan orang. Tim pengembang harus memastikan bahwa setiap tim memiliki semua keterampilan yang diperlukan untuk agar proses pengembangan berjalan dengan baik.

2.7 Website

Website adalah kumpulan halaman web yang saling terhubung dan disimpan dalam server web. Halaman web tersebut dapat berisi teks, gambar, audio, video, dan animasi. *Website* dapat diakses oleh pengguna internet melalui browser web [14]. Selain itu, website juga menjadi jalur komunikasi yang efektif antara pengguna dengan perusahaan atau individu, memungkinkan interaksi langsung melalui fitur – fitur yang tersedia. Dengan fungsi yang dimiliki barusan, sebuah website tidak hanya menjadi representasi digital dari suatu entitas, tetapi juga menjadi tempat untuk menjalin hubungan, menyebarkan informasi, dan memenuhi kebutuhan pengguna.

2.8 API

API (*Application Programming Interface*) adalah serangkaian aturan dan protokol yang memungkinkan satu perangkat lunak atau aplikasi berkomunikasi dengan perangkat lunak atau layanan lainnya [15]. API memungkinkan pengembang mengintegrasikan fungsionalitas atau layanan dari suatu aplikasi atau platform ke dalam aplikasi atau *platform* lain tanpa perlu mengetahui detail implementasi internal. API menyediakan antarmuka yang terstandar untuk pertukaran informasi antar sistem, memungkinkan interaksi yang efisien dan konsisten. Pengembang dapat memanfaatkan API yang ada untuk memperluas fungsionalitas aplikasi mereka, mengintegrasikan layanan pihak ketiga, atau bahkan membangun aplikasi yang sepenuhnya baru dengan memanfaatkan layanan yang tersedia melalui API. Dengan demikian, API menjadi fondasi yang kuat dalam ekosistem perangkat lunak modern, memungkinkan aplikasi untuk saling berinteraksi dan terhubung satu sama lain dengan cara yang aman, terstandarisasi, dan efisien.

2.9 Cloud Computing

Cloud Computing merupakan model komputasi yang memungkinkan akses mudah dan sesuai permintaan ke sekumpulan sumber daya komputasi yang dapat dikonfigurasi, contohnya seperti jaringan, server, penyimpanan, aplikasi, dan layanan yang dapat dengan cepat disediakan dan dilepaskan dengan usaha manajemen minimal atau interaksi penyedia layanan [16]. Model ini memungkinkan organisasi dan individu untuk menggunakan sumber daya teknologi informasi tanpa perlu memiliki dan mengelola infrastruktur fisik sendiri.

Keuntungan utama dari cloud computing meliputi efisiensi biaya, skalabilitas, fleksibilitas, dan aksesibilitas global. Organisasi dapat menghindari biaya besar untuk pembelian dan pemeliharaan infrastruktur IT dengan mengadopsi model bayar sesuai penggunaan. Selain itu, sumber daya dapat dengan mudah ditingkatkan atau dikurangi sesuai kebutuhan, mendukung mobilitas dan kerja jarak jauh. Namun, ada tantangan yang perlu dipertimbangkan, seperti keamanan data dan kepatuhan terhadap regulasi yang harus dipastikan oleh pengguna layanan cloud. Adapun layanan *cloud computing* yang akan dijelaskan sebagai berikut :

2.9.1 Layanan Cloud Computing

Cloud computing dibagi menjadi beberapa model layanan utama yang berbeda, masing-masing menawarkan tingkat kontrol dan tanggung jawab yang berbeda [16]:

1. *Software as a Service* (SaaS):

   SaaS menyediakan aplikasi perangkat lunak melalui internet. Pengguna dapat mengakses aplikasi ini menggunakan web browser tanpa perlu menginstal perangkat lunak di komputer mereka. Contoh populer dari SaaS termasuk Google Workspace, Microsoft Office 365, dan Salesforce.

2. *Platform as a Service* (PaaS):

PaaS menyediakan platform yang memungkinkan pengembang untuk membangun, menguji, dan mengelola aplikasi tanpa harus mengelola infrastruktur dasar seperti server dan jaringan. Contoh PaaS termasuk Google App Engine dan Microsoft Azure .

3. *Infrastructure as a Service* (IaaS):

IaaS menyediakan sumber daya komputasi virtual, seperti server, penyimpanan, dan jaringan, yang dapat dikonfigurasi dan digunakan sesuai kebutuhan. Contoh IaaS termasuk Amazon Web Services (AWS), Google Cloud Platform (GCP), dan IBM Cloud.

4. Layanan Penyimpanan (*Storage Services*)

Layanan penyimpanan cloud menawarkan penyimpanan data yang skalabel dan aman. Data dapat diakses dari mana saja melalui internet, dan banyak penyedia layanan menyertakan fitur seperti backup otomatis dan pemulihan bencana. Contoh layanan penyimpanan termasuk Amazon S3, Google Cloud Storage, MinIO, dan Dropbox.

5. Layanan Jaringan (*Network Services*)

Layanan jaringan dalam cloud computing mencakup berbagai layanan yang memungkinkan konektivitas dan komunikasi yang andal dan aman. Ini termasuk layanan seperti Virtual Private Network (VPN), Content Delivery Networks (CDN), dan load balancers yang memastikan distribusi lalu lintas yang efisien dan keamanan jaringan. Contoh layanan jaringan termasuk Amazon CloudFront dan Google Cloud CDN.

6. Keamanan (*Security Services*)

Keamanan merupakan komponen penting dalam cloud computing yang mencakup berbagai layanan seperti manajemen identitas dan akses (IAM), enkripsi data, dan alat pengelolaan informasi dan kejadian keamanan (SIEM). Ini bertujuan untuk melindungi data dan aplikasi dari ancaman serta memastikan kepatuhan terhadap regulasi dan standar keamanan. Contoh

layanan keamanan termasuk AWS Identity and Access Management (IAM) dan Google Cloud Security Command Center.

7. Layanan Pengelolaan dan Pemantauan (*Management and Monitoring Services*)

   Layanan ini menyediakan alat untuk mengelola, memantau, dan mengoptimalkan kinerja aplikasi dan infrastruktur cloud. Ini termasuk pemantauan kinerja, logging, dan alat otomatisasi yang membantu dalam menjaga operasi yang efisien dan mendeteksi masalah sebelum mereka menjadi kritis. Contoh layanan ini termasuk AWS CloudWatch dan Google Stackdriver.

2.10 MinIO

MinIO adalah server penyimpanan yang berbasis komputasi awan yang terbuka dan ringan, dirancang untuk menyediakan layanan penyimpanan objek yang cepat, mudah digunakan, dan mempunyai performa tinggi. MinIO digunakan untuk menyimpanan data besar, arsip, dan media seperti gambar dan audio. MinIO dipilih karena kemampuannya untuk mengatasi tuntutan penyimpanan data yang besar dan kompleks dengan kecepatan dan skalabilitas tinggi. Dengan arsitektur yang ringan dan *open source*, MinIO memberikan fleksibilitas bagi pengguna untuk menyesuaikan dan mengintegrasikan solusi penyimpanan objek mereka sesuai kebutuhan. MinIO menjadi pilihan di kalangan pengembang dan organisasi yang mencari solusi penyimpanan yang efisien [7].

2.10.1 Komponen MinIO

Berikut adalah penjelasan singkat mengenai komponen-komponen pada MinIO [7]:

1. *Objek Storage Layer* (*Storage Backend*):

Objek Storage Layer adalah lapisan inti dari MinIO yang bertanggung jawab untuk menyimpan data objek. *Objek Storage Layer* akan bekerja secara langsung dengan komponen yang dikenal sebagai *Storage Backend*. *Storage Backend* ini adalah komponen yang memungkinkan MinIO untuk menyimpan dan mengelola data objek dengan efisien. *Storage Backend* pada MinIO merujuk pada tempat fisik di mana objek-objek data sebenarnya disimpan.

Dalam komponen ini terdapat *Bucket*, *bucket* merupakan entitas yang berfungsi sebagai wadah virtual untuk mengorganisir dan menyimpan objek-objek data. Setiap *bucket* memiliki nama unik di dalam klaster penyimpanan dan digunakan sebagai unit dasar untuk mengelompokkan data. Misalnya, dalam konfigurasi MinIO dengan *storage backend* berbasis disk lokal atau *cloud storage*, pengguna dapat membuat beberapa bucket untuk memisahkan dan mengelola data berdasarkan tujuan atau kebutuhan aplikasi mereka. Setiap bucket menyimpan objek-objek yang dikelola, termasuk metadata yang menggambarkan karakteristik dari setiap objek seperti nama, ukuran, dan *timestamp*, memungkinkan pengguna untuk mengakses dan mengatur data secara efisien melalui antarmuka yang disediakan.

2. Kluster

Kluster pada MinIO adalah kumpulan node atau server yang bekerja bersama untuk menyediakan layanan penyimpanan objek yang terdistribusi. Kluster pada MinIO akan bekerja bersama untuk menyediakan layanan penyimpanan berbasis objek yang skalabel, tahan terhadap kegagalan, dan tingkat ketersediaan tinggi.

3. *Node*

Node merujuk pada unit komputasi atau server fisik yang merupakan bagian dari klaster penyimpanan MinIO. *Node-node* ini berperan sebagai penyimpan data dan berkolaborasi untuk menyediakan layanan penyimpanan objek yang skalabel dan tahan banting. Secara teknis, *node-node* ini dapat ditempatkan di

berbagai infrastruktur, baik itu di pusat data lokal maupun di lingkungan *cloud* seperti AWS, Azure, atau Google Cloud Platform.

4. *Gateway Layer* :

Gateway dalam MinIO berfungsi sebagai antarmuka untuk menghubungkan aplikasi atau pengguna dengan layanan penyimpanan objek. *Gateway* ini dapat beroperasi dalam mode yang berbeda, seperti *Gateway* NAS untuk menyediakan penyimpanan objek melalui protokol file seperti NFS atau SMB. NFS (*Network File System*) dan SMB (*Server Message Block*) sendiri adalah dua protokol yang digunakan untuk berbagi file dan penyimpanan dalam jaringan komputer.

5. *Erasure Coding* (Distributed RAID):

MinIO menggunakan teknik *Erasure Coding* untuk memberikan redundansi dan ketahanan terhadap kegagalan hardware atau node dalam klaster penyimpanan. *Erasure Coding* akan memungkinkan MinIO untuk menyimpan salinan data yang terfragmentasi di beberapa node, sehingga data dapat dipulihkan bahkan jika ada beberapa node yang mengalami kegagalan.

6. Metadata Layer:

Metadata dalam MinIO adalah informasi tambahan yang menyertai setiap objek yang disimpan. Metadata ini berisi informasi tentang objek seperti nama, ukuran, tipe MIME, tanggal modifikasi, dan sebagainya. Metadata ini penting untuk manajemen dan pengaturan objek secara efisien.

7. *API Layer*:

API Layer dalam MinIO adalah bagian dari sistem yang memungkinkan pengguna atau aplikasi untuk berinteraksi dengan layanan penyimpanan objek MinIO. MinIO dirancang agar sepenuhnya kompatibel dengan API Amazon S3, yang merupakan cara standar untuk menyimpan dan mengakses data dalam bentuk objek di *cloud computing*. Ini berarti pengguna MinIO dapat menggunakan alat dan aplikasi yang sudah mendukung S3 tanpa perlu

mengubah atau memodifikasi kode mereka. Kompatibilitas ini sangat penting karena memungkinkan integrasi yang mudah dengan ekosistem perangkat lunak yang luas yang sudah ada, memperluas kemungkinan penggunaan MinIO dalam berbagai skenario aplikasi dari mulai pengembangan aplikasi hingga analitik data.

8. *Management Console* dan *CLI Tools*:

   MinIO dilengkapi dengan antarmuka manajemen berbasis web (*Management Console*) serta berbagai alat *Command-Line Interface* (CLI) untuk administrasi dan konfigurasi sistem. Komponen ini akan termasuk memantau performa, pengaturan keamanan, dan manajemen bucket serta objek.

2.10.2 Fitur MinIO

MinIO memiliki beberapa fitur - fitur dalam konteks ruang penyimpanan objek berbasis komputasi awan. Berikut adalah beberapa fitur utama MinIO [7]:

1. *Open Source* dan Gratis: MinIO didistribusikan di bawah lisensi GNU AGPL V3, yang membuatnya gratis untuk digunakan dan dimodifikasi oleh siapa saja. *GNU Affero General Public License* version 3 (AGPLv3) sendiri adalah lisensi perangkat lunak sumber terbuka yang dikeluarkan oleh *Free Software Foundation* (FSF). AGPLv3 merupakan varian dari GNU *General Public License* (GPL) yang dirancang khusus untuk aplikasi yang berjalan di server dan diakses melalui jaringan. MinIO tentu akan memberikan fleksibilitas dan kontrol yang lebih besar kepada pengguna dalam mengelola infrastruktur penyimpanan mereka.
2. Performa Tinggi: MinIO dirancang untuk memberikan kinerja penyimpanan objek yang sangat tinggi. Ini tercapai melalui teknologi seperti paralelisme yang tinggi dan optimasi IO.
3. Skalabilitas: MinIO memungkinkan pengguna untuk membangun klaster penyimpanan objek yang sangat besar dan dapat diperluas dengan mudah.

Ini cocok untuk aplikasi yang membutuhkan skalabilitas horizontal untuk memenuhi kebutuhan data yang terus berkembang.

4. Kompatibilitas: MinIO kompatibel dengan protokol penyimpanan objek standar seperti Amazon S3. MinIO akan memungkinkan integrasi yang mudah dengan aplikasi dan layanan yang sudah ada yang menggunakan S3 API, sehingga memudahkan migrasi ke MinIO atau integrasi dengan infrastruktur cloud publik seperti AWS S3.
5. Keselamatan dan Keamanan: MinIO menyediakan berbagai fitur keamanan termasuk enkripsi data di istirahat (*data at rest*) dan data dalam perjalanan (*data in transit*), serta pengaturan akses berbasis kebijakan yang kuat menggunakan AWS IAM dan pola penyimpanan terenkripsi (SSE).
6. Fleksibilitas Penyimpanan: MinIO mendukung berbagai jenis penyimpanan fisik seperti HDD dan SSD, serta memungkinkan konfigurasi penambahan penyimpanan lain seperti NAS atau cloud storage, memberikan fleksibilitas dalam desain infrastruktur penyimpanan.
7. Dukungan Komunitas yang Kuat: Karena sifatnya yang *open-source*, MinIO memiliki komunitas pengguna yang aktif yang terlibat dalam pengembangan dan dukungan.

2.11 Google Calendar API

Google Calendar API adalah API yang disediakan oleh Google untuk mengintegrasikan layanan Google Calendar ke dalam aplikasi atau situs web pihak ketiga [17]. Dengan menggunakan Google Calendar API, pengembang dapat membuat aplikasi yang terhubung dengan kalender Google pengguna, baik untuk menampilkan informasi kalender, membuat, mengedit, atau menghapus acara, serta melakukan berbagai operasi lainnya. Fungsi utama dari Google Calendar API adalah memungkinkan pengembang untuk mengintegrasikan data kalender Google ke dalam aplikasi mereka. Dengan demikian, Google Calendar API memberikan fleksibilitas dan kekuatan kepada pengembang untuk menciptakan berbagai macam

aplikasi yang terhubung dengan kalender Google, meningkatkan produktivitas dan kenyamanan pengguna dalam mengelola jadwal dan acara mereka.

2.11.1 Fitur Google Calendar API

Berikut adalah beberapa fitur utama yang ditawarkan oleh Google Calendar API [17]:

1. Manajemen Acara:

   Google Calendar API memungkinkan pengguna untuk membuat, membaca, memperbarui, dan menghapus acara dalam kalender pengguna. Google Calendar API dapat menentukan detail seperti judul acara, lokasi, deskripsi, waktu mulai dan selesai, serta pengulangan acara.

2. Manajemen Kalendar:

   API ini mendukung pembuatan dan pengaturan kalendar, termasuk menambahkan kalendar baru, mengubah pengaturan kalendar (seperti warna atau nama kalendar), dan menghapus kalendar.

3. Notifikasi Acara:

   Pengguna dapat mengatur notifikasi untuk acara tertentu agar pengguna mendapatkan pemberitahuan sebelum atau saat acara dimulai. Google Calendar API menyediakan mekanisme untuk mengelola notifikasi ini.

4. Akses Berbasis OAuth 2.0:

   API ini menggunakan protokol OAuth 2.0 untuk otentikasi dan otorisasi. Ini berarti aplikasi harus meminta izin akses dari pemilik kalendar sebelum dapat mengakses atau mengelola data kalendar.

5. Integrasi dengan Layanan Google lainnya:

Google Calendar API dapat diintegrasikan dengan layanan Google lainnya seperti Gmail, Google Drive, dan Google Contacts. Hal ini memungkinkan aplikasi untuk menghubungkan acara dengan email, file, atau kontak terkait.

6. Pencarian dan Filtering:

API menyediakan kemampuan untuk melakukan pencarian dan filtering terhadap acara atau kalendar berdasarkan kriteria tertentu seperti rentang waktu, kata kunci, atau kategori.

7. Multi-platform:

API ini didukung di berbagai platform dan lingkungan pengembangan, termasuk pengembangan aplikasi web, aplikasi seluler, dan integrasi dengan sistem atau perangkat keras lainnya.

8. Skalabilitas dan Ketersediaan Tinggi:

Layanan Google Calendar di belakang API ini menawarkan skalabilitas tinggi dan ketersediaan yang baik, sehingga cocok digunakan dalam aplikasi yang memerlukan waktu operasional yang kritis.

9. Perlindungan Keamanan Data:

Google Calendar API menggunakan keamanan TLS untuk enkripsi data saat transit, serta menyediakan opsi untuk enkripsi data di istirahat dengan menggunakan Server-Side Encryption (SSE).

10. Dokumentasi dan Dukungan:

Google menyediakan dokumentasi yang lengkap dan dukungan teknis untuk pengembang yang menggunakan Google Calendar API. Dokumentasi ini mencakup panduan penggunaan, contoh kode, dan referensi API yang detail.

2.12 Spring Boot

Spring Boot adalah sebuah framework untuk pengembangan aplikasi berbasis Java yang memudahkan pembuatan, pengujian, dan pengelolaan aplikasi yang menggunakan Spring Framework. Dikembangkan oleh Pivotal Software, Spring Boot dirancang untuk menyederhanakan konfigurasi dan pengembangan aplikasi Spring, memungkinkan pengembang fokus pada logika bisnis inti [18]. Secara umum, komponen pada Spring Boot terdiri dari berbagai kelas dan konfigurasi yang bekerja bersama untuk membangun aplikasi secara keseluruhan.

Berikut adalah penjelasan singkat tentang beberapa komponen utama dalam Spring Boot:

1. *Entity* : Merupakan representasi dari objek atau entitas dalam aplikasi. Biasanya, entitas ini memodelkan data yang disimpan dalam database.
2. *Repository* : Bertanggung jawab untuk berinteraksi dengan database. Komponen ini biasanya menyediakan operasi CRUD (*Create*, *Read*, *Update*, *Delete*) untuk entitas pada aplikasi.
3. *Service* : Kelas yang berisi logika bisnis utama pada aplikasi. Komponen ini akan mengimplementasikan operasi-operasi yang lebih kompleks, memanipulasi data, dan berinteraksi dengan repositori.
4. *Controller* : Berfungsi sebagai titik masuk utama ke dalam aplikasi. *Controller* menerima permintaan HTTP dari klien, memprosesnya, dan menghasilkan respons. Singkatnya, komponen ini berfungsi sebagai perantara antara pengguna dan logika bisnis aplikasi.
5. *Configuration* : Kelas yang mengelola konfigurasi pada aplikasi Spring Boot. Spring Boot menggunakan anotasi dan file konfigurasi untuk mengatur berbagai aspek aplikasi, seperti koneksi database, URL endpoint, dan lainnya.
6. *Dependency Injection* (DI) : Spring Boot menggunakan DI untuk mengelola dependensi dengan berbagai komponen pada aplikasi.

2.13 React JS

ReactJS adalah sebuah framework JavaScript yang digunakan untuk membuat antarmuka pengguna (UI) yang interaktif dan responsif. ReactJS adalah framework yang populer dan banyak digunakan untuk mengembangkan aplikasi web dan aplikasi mobile. ReactJS pertama kali dikembangkan oleh Jordan Walke di Facebook pada tahun 2011. ReactJS awalnya dikembangkan sebagai framework untuk mengembangkan aplikasi web yang interaktif dan responsif [19].

Berikut adalah penjelasan singkat tentang komponen-komponen utama dalam ReactJS:

1. Komponen: Unit dasar dalam React untuk membangun antarmuka pengguna. Terdapat komponen fungsional dan komponen kelas.
2. *Props* (*Properties*): Digunakan untuk mengirimkan data dari komponen induk ke komponen anak. Props bersifat read-only di komponen anak.
3. State: Digunakan untuk menyimpan data dinamis dalam komponen. Perubahan state menyebabkan re-rendering komponen.
4. Siklus Hidup (*Lifecycle*): Metode yang dipanggil dalam siklus hidup komponen, seperti *mounting*, *updating*, dan *unmounting*.
5. *Event Handling*: Mengelola interaksi pengguna seperti klik tombol atau input, mirip dengan event handling di HTML.
6. Komposisi Komponen: Memungkinkan membangun UI kompleks dengan mengombinasikan komponen-komponen ke dalam komponen yang lebih besar.
7. Hooks: Fitur baru dalam React untuk mengelola state dan fitur React lainnya dalam komponen fungsional tanpa perlu menggunakan kelas.
8. Perutean (*Routing*): React Router digunakan untuk menangani perutean di aplikasi React, memungkinkan navigasi antar halaman atau komponen secara dinamis.

2.14 DBMS

DBMS (*Database Management System*) didefinisikan sebagai sistem perangkat lunak yang mengelola dan menyediakan akses terstruktur terhadap basis data. DBMS memungkinkan pengguna untuk mendefinisikan, membuat, dan memelihara basis data dengan cara yang terorganisir dan efisien. DBMS mencakup manajemen data seperti pengaturan integritas, pengelolaan transaksi untuk memastikan konsistensi data, dan pengendalian akses untuk melindungi keamanan informasi. Selain itu, DBMS menyediakan mesin query yang memungkinkan pengguna untuk mengambil data menggunakan bahasa query seperti SQL, dengan optimisasi kinerja untuk memastikan eksekusi query yang cepat dan efisien [20]. Melalui fitur-fitur ini, DBMS tidak hanya memfasilitasi pengelolaan data yang terstruktur tetapi juga mendukung aplikasi-aplikasi informasi yang kompleks dalam berbagai lingkungan komputasi modern.

RDBMS (*Relational Database Management System*) adalah salah satu jenis DBMS yang paling umum digunakan dalam industri untuk pengelolaan data yang terstruktur berdasarkan model data relasional [21]. Model ini memanfaatkan tabel untuk menyimpan data, dengan setiap tabel terdiri dari baris (record) yang merepresentasikan entitas tertentu dan kolom (field) yang menyimpan atribut dari entitas tersebut. Penggunaan kunci primer dalam setiap tabel memungkinkan identifikasi unik dari setiap rekaman data, yang penting untuk mempertahankan integritas data. RDBMS menggunakan SQL sebagai bahasa query standar untuk memanipulasi data, dengan fitur-fitur seperti optimisasi kueri yang mendukung eksekusi yang cepat dan efisien. Dengan demikian, RDBMS tidak hanya mendukung pengelolaan basis data yang terstruktur dan aman, tetapi juga memfasilitasi implementasi aplikasi-aplikasi informasi kompleks dalam berbagai lingkungan komputasi modern.

2.15 PostgreSQL

PostgreSQL adalah sebuah sistem manajemen basis data relasional (RDBMS) open-source yang sangat canggih dan kuat. Dikembangkan awalnya di Universitas California, Berkeley, PostgreSQL dirilis di bawah lisensi PostgreSQL, yang mirip dengan lisensi MIT [22]. PostgreSQL merupakan salah satu sistem manajemen basis data relasional (RDBMS) *open source* yang dinilai canggih saat ini. RDBMS adalah tipe sistem manajemen basis data yang berbasis pada model relasional, di mana data disimpan dalam bentuk tabel yang terstruktur dan hubungan antar tabel dijaga melalui kunci asing. Salah satu fitur kunci dari PostgreSQL adalah kemampuan untuk menyimpan dan mengelola data dengan skala besar, serta mendukung berbagai jenis data dan indeks untuk optimisasi kinerja.

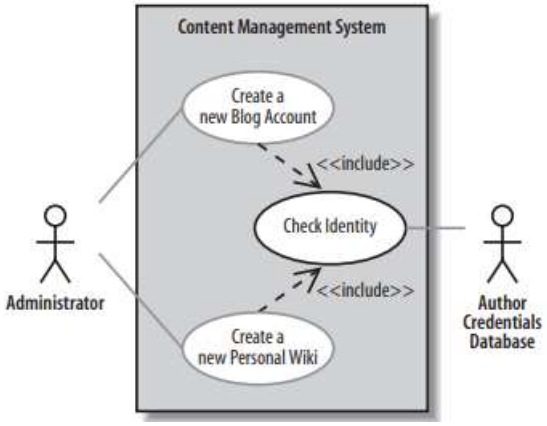
2.16 UML

Unified Modeling Language (UML) adalah bahasa pemodelan visual yang digunakan untuk menggambarkan, mendokumentasikan, dan mengotomatiskan sistem perangkat lunak. UML adalah bahasa yang standar dan dapat digunakan oleh berbagai macam perangkat lunak. UML membantu dalam representasi visual dari struktur dan perilaku sistem yang kompleks.

UML diperkenalkan sebagai alat yang dapat digunakan oleh pengembang perangkat lunak untuk menggambarkan persyaratan sistem, desain struktur, interaksi antar objek, dan berbagai aspek lain dari sistem yang sedang dikembangkan. UML juga memfasilitasi komunikasi yang lebih baik antara anggota tim pengembangan, pemangku kepentingan, dan bahkan di antara komponen-komponen perangkat lunak yang berbeda [23]. Adapun UML yang akan dipakai pada penelitian ini yang diantaranya :

2.16.1 Use Case Diagram

Use Case Diagram adalah representasi visual yang menunjukkan keterkaitan antara aktor dan sistem. Diagram ini mampu merincikan interaksi antara satu atau lebih aktor dengan sistem yang sedang dikembangkan. Selain itu, diagram use case dapat memberikan gambaran terkait fungsi-fungsi yang terdapat di dalam sistem, dan juga dapat mencerminkan interaksi antara aktor dan sistem [23]. Use Case sendiri adalah gambaran fungsional dari sebuah sistem. Berikut Gambar 2.2 Contoh Use Case Diagram adalah contoh dari *use case diagram*.

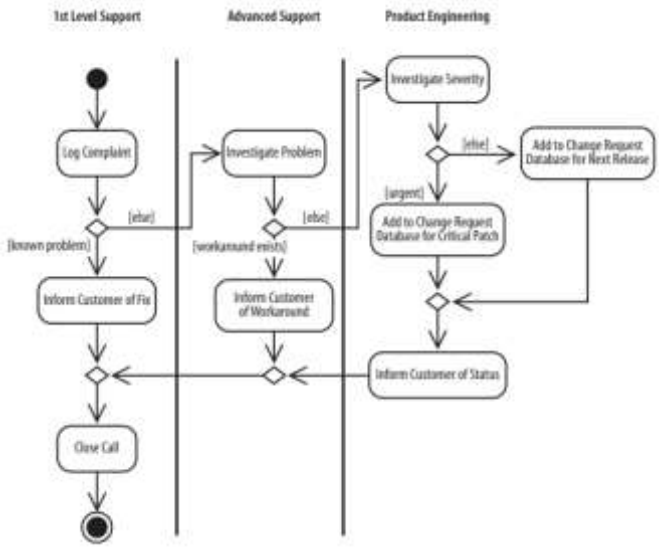


Gambar 2.2 Contoh Use Case Diagram

Sumber : Learning UML 2.0 [23]

2.16.2 Activity Diagram

Activity Diagram adalah satu varian dari diagram UML yang digunakan untuk merinci aktivitas dan alur kerja dalam suatu sistem atau proses. Diagram ini berguna untuk mendokumentasikan dan menggambarkan langkah-langkah atau aktivitas yang terjadi dalam suatu proses atau skenario. Penggambaran diagram aktivitas dimulai dari simpul awal (initial node) dan berakhir pada simpul akhir (end node) [23]. Berikut Gambar 2.3 Contoh Activity Diagram adalah contoh dari *activity diagram*.

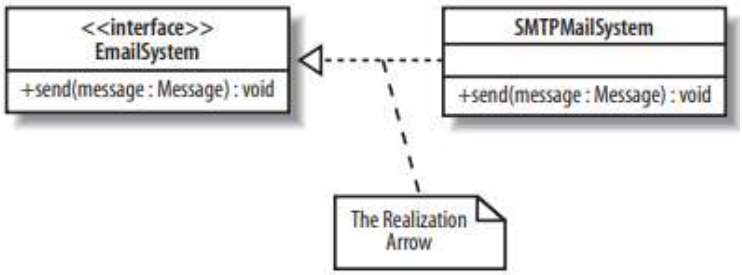


Gambar 2.3 Contoh Activity Diagram

Sumber : Learning UML 2.0 [23]

2.16.3 Class Diagram

Class Diagram adalah jenis diagram UML yang digunakan untuk memodelkan struktur statis dari suatu sistem, dengan fokus pada entitas-entitas (class atau objek) yang terlibat, serta hubungan dan propertinya. Diagram ini membantu dalam mendokumentasikan struktur kelas, hierarki kelas, dan hubungan antar kelas dalam suatu sistem perangkat lunak [23]. Diagram Class akan menggambarkan hubungan apa yang terjadi bukan apa yang terjadi jika mereka berhubunga. Berikut Gambar 2.4 Contoh Class Diagram adalah contoh dari *class diagram*.

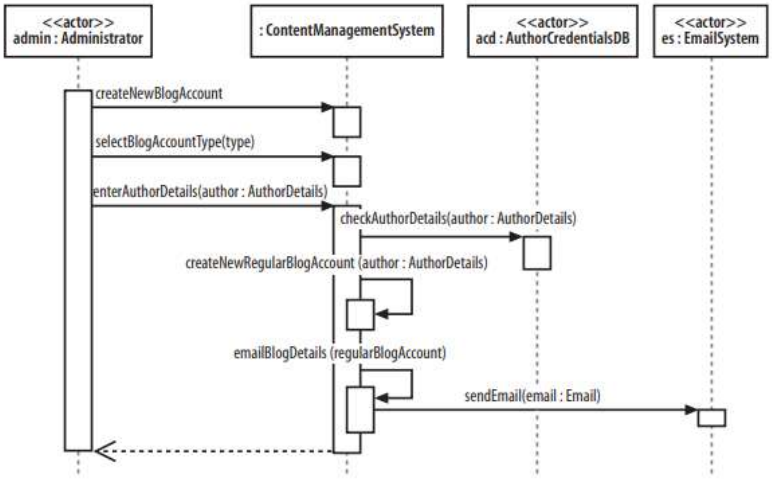


Gambar 2.4 Contoh Class Diagram

Sumber : Learning UML 2.0 [23]

2.16.4 Sequence Diagram

Sequence diagram adalah jenis diagram UML yang digunakan untuk memodelkan interaksi antar objek dalam suatu sistem atau proses. Diagram ini menunjukkan urutan pesan atau panggilan metode yang dikirimkan antar objek selama waktu tertentu, memberikan gambaran visual tentang alur eksekusi dalam suatu skenario tertentu [23]. Berikut Gambar 2.5 Contoh Sequence Diagram adalah contoh dari *sequence diagram*.



Gambar 2.5 Contoh Sequence Diagram

Sumber : Learning UML 2.0 [23]

2.17 Pengujian Blackbox

Blackbox *testing* adalah metode pengujian perangkat lunak yang menilai fungsionalitas aplikasi tanpa memeriksa kode sumber atau struktur internalnya. Blackbox testing berfokus pada input dan output dari sistem, memastikan bahwa perangkat lunak berfungsi sesuai dengan spesifikasi fungsional dan persyaratan pengguna. Pengujian ini menggunakan berbagai teknik, seperti pengujian fungsional, pengujian batas, dan partisi ekivalen, untuk mengidentifikasi cacat dalam sistem dengan menguji bagaimana sistem merespons berbagai kondisi dan input. Dengan pendekatan ini, penguji aplikasi dapat menilai apakah perangkat

lunak memenuhi kebutuhan dan harapan pengguna akhir, tanpa memerlukan pengetahuan mendalam tentang implementasi internal kode [24].