BAB II
TINJAUAN PUSTAKA

2.1 Sistem Pendukung Keputusan (SPK)

SPK adalah sistem pemodelan yang terdiri dari prosedur dalam pengolahan data dan hasil yang dapat membantu *decision maker* dalam memilih keputusan. SPK diciptakan untuk menjadi solusi yang membantu proses pengambilan keputusan atas kriteria yang menjadi syarat pada alternatif (Kusumantara 2021), serta diciptakan untuk meningkatkan akurasi dalam menghasilkan suatu keputusan (Laili and Farida 2020).

Agar mencapai tujuannya maka sistem tersebut harus terdapat interaksi antara sistem dengan user sehingga mudah untuk dikontrol dan mudah beradaptasi dengan data yang banyak untuk diolah menjadi *output* yang sesuai dengan user harapkan (Limbong 2020).

2.1.1 Ciri Khas Sistem Pendukung Keputusan

Setiap program mempunyai ciri khas atau karakteristik yang membuatnya berbeda dengan program lainnya, untuk Sistem pendukung keputusan memiliki karakteristik (Ruskan, Ibrahim, and Hartini 2013):

a. Mulai dari permasalahan terstruktur, semistruktur, dan tidak terstruktur semuanya terdukung oleh sistem pendukung keputusan.
b. Organisasi merasakan hasil akhir yang dibuat oleh sistem pendukung keputusan, mulai dari elemen terkecil dari organisasi hingga elemen yang terbesar.
c. Sistem pendukung keputusan memiliki *user interface* yang diproses oleh mesin namun manusia merupakan peran utama sebagai pengontrol pengambilan keputusan.
d. Model sistem pendukung keputusan yang dipakai adalah model yang terstruktur secara sistematis dan matematis.
e. *Easy to Use*, sebuah program yang mudah digunakan, memungkinkan pengambil keputusan untuk cepat beradaptasi dengan sistem pendukung keputusan di mana ia dibuat, dan

memiliki tingkat fleksibilitas yang tinggi dalam mengembangkan dan memilih pendekatan untuk memecahkan masalah.

2.1.2 Tujuan Sistem Pendukung Keputusan

Sistem pendukung keputusan diciptakan untuk tidak menggantikan peran dari seorang *decision maker*, karena sistem pendukung keputusan diciptakan untuk mencapai tujuan seperti berikut (Alwendi 2020):

a. Membantu *decision maker* untuk mengambil keputusan yang sesuai dengan permasalahan .
b. Sistem pendukung keputusan sebagai penunjang dari keputusan yang diambil *decision maker*.
c. Memberikan akurasi yang tinggi dalam pengambilan keputusan.
d. Memberikan kecepatan sehingga *decision maker* dapat menyelesaikan permasalahan tanpa berlarut-larut.
e. Seiring kecepatan kerja yang meningkat, maka produktivitas juga meningkat, sehingga dapat menghasilkan personel yang memiliki kemampuan diatas rata-rata.

2.2 *Analytical Hierarchy Process* (AHP)

Metode AHP dikembangkan oleh Prof. Thomas L. Saaty dengan tujuan untuk menjadikan metode tersebut sebagai algoritma untuk permasalahan multikriteria. Metode AHP merupakan metode yang mengamati faktor-faktor intuisi, pengalaman, preferensi dan persepsi yang dipakai dalam sistem pendukung keputusan (Umarsyah, Bagaskara, and Jumaryadi 2021).

Pengembangan metode tersebut dengan membandingkan derajat kepentingan antar kriteria serta menangani sistem yang kompleks. Dikatakan kompleks karena struktur permasalahan yang dihadapi merupakan struktur yang tidak jelas dan tidak adanya data kuantitatif terkait permasalahan tersebut, yang menjadi solusi untuk menghadapinya adalah intuisi manusia (Diana and Retno 2019). Dengan menyusun permasalahan menjadi susunan hierarki yang kemudian diberikan nilai numerik, maka

dapat memberikan penilaian terhadapat permasalahan tersebut secara subjektif untuk menentukan variabel mana yang lebih penting (Ruskan 2017).

2.2.1 Tahapan Metode AHP

Dalam pemakaian metode AHP harus sesuai dengan prosedur sebagai berikut (Poningsih and Risna 2020):

a. Membuat Hierarki

Hierarki adalah perwakilan dari suatu permasalahan dengan kompleksitas tinggi dalam sebuah struktur bertingkat dimana tingkat awal adalah tujuan yang akan dicapai tingkat faktor, kriteria, subkriteria, dan seterusnya ke bawah hingga tingkat terakhir dari alternatif (Lestari, S, and Fadlil 2020). Sehingga penjabaran dari suatu permasalahan dapat menggambarkan hubungan antara kriteria dengan alternatif.

b. Buat *pairwise comparison* dari matriks

Berdasarkan matriks kriteria, lakukan *pairwise comparison* antara elemen-elemennya sehingga dapat diperoleh nilai untuk masing-masing kriteria. Nilai pada *pairwise comparison* dapat diukur berupa nilai 1 sampai dengan 9 menggunakan tabel skala (Wantoro 2020).

Tabel 2.1 Nilai *Pairwise Comparison*

NO	NILAI	KETERANGAN
1.	2, 4, 6 dan 8	Nilai pada dua *pairwise comparison* yang berdekatan
2.	1	Nilai pada dua *pairwise comparison* sama pentingnya
3.	3	Nilai pada salah satu *pairwise comparison* sedikit lebih penting
4.	5	Nilai pada salah satu *pairwise comparison* lebih penting
5.	7	Nilai pada salah satu *pairwise comparison* sangat lebih penting
6.	9	Nilai pada salah satu *pairwise comparison* mutlak lebih penting

c. Menormalisasi *pairwise comparis*on

Menormalisasi pairwise comparison sehingga menghasilkan nilai yang dapat dihitung menjadi *eigenvektor*. Normalisasi *pairwise comparison* dengan cara membagi nilai pada kriteria dengan jumlah keseluruhan masing-masing kolom pada matriks *pairwise comparison* (Alif et al. 2021). Dapat digambarkan dengan rumus sebagai berikut:

$$\frac{\text{Nilai Kriteria}}{\Sigma\text{Kolom Kriteria}} \quad (1)$$

d. Menentukan *Eigenvektor*

Eigenvektor atau disebut juga bobot yaitu nilai yang menentukan prioritas dari yang terendah hingga yang tertinggi. Untuk mencari nilai *eigenvektor* dengan cara menjumlahkan tiap baris kriteria, kemudian dibagi dengan banyaknya kriteria (Hanggoro, Aji, and Saputra 2020). dapat digambarkan dengan rumus sebagai berikut:

$$Eigenvektor = \frac{\Sigma\text{Baris Kriteria}}{\text{Banyaknya Elemen Kriteria}} \quad (2)$$

e. *Consistency Measure* (CM)

Consistency Measure adalah nilai yang didapatkan dari perkalian matriks antara baris *pairwise comparison* dengan kolom *eigenvektor* (Informatika, Teknik, and Fitness 2022).

f. Menentukan *Lambda* maksimal (λ_{maks})

Pada *Consistency Measure* dilakukan penjumlahan kemudian dicari nilai rata-rata. Maka hasil rata-rata tersebut adalah *Lambda* maksimal (Zulfikar 2020).

g. Menguji Konsistensi

Pada tahapan ini adalah tahap perhitungan konsistensi pada kriteria. Dapat dikatakan konsisten jika CR < 10% atau 0,1 namun jika kriteria dikatakan tidak konsisten maka penilaian kriteria harus diulang (Suartini, Wirawan, and Divayana 2019). Pertama-tama

melakukan perhitungan *Consistency Index* (CI) digambarkan dengan rumus sebagai berikut:

$$Consistency\ Index\ (CI) = \frac{\lambda^{maks} - n}{n - 1} \quad (3)$$

Jika nilai *Consistency Index* (CI) sudah ditemukan, maka dilanjutkan dengan rumus sebagai berikut:

$$Consistency\ Ratio\ (CR) = \frac{CI}{RI} \quad (4)$$

RI adalah *Random Index* yang akan dihitung bersamaan dengan hasil dari CI. Tabel 2.2 merupakan daftar dari *Random Index*.

Tabel 2.2 Nilai *Random Index* (RI)

Banyaknya elemen (n)	**1 atau 2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
IR	0	0,58	0,9	1,12	1,24	1,32	1,41	1,45	1,49

2.3 *Simple Additive Weighting* (SAW)

Metode *Simple Additive Weighting* (SAW) atau yang disebut juga metode penjumlahan terbobot yang artinya menghitung peringkat kinerja untuk setiap alternatif menggunakan semua kriteria (Hanggoro et al. 2020).

Penggunaan metode SAW kali ini untuk menentukan perangkingan dari tiap alternatif menggunakan nilai akhir yang paling tinggi dengan cara melakukan perhitungan pada bobot kriteria dengan matriks yang telah dinormalisasi (Maratullatifah, Widodo, and Adi 2020).

2.3.1 Tahapan Metode SAW

Tahapan-tahapan yang harus ditempuh agar metode SAW dapat digunakan secara maksimal, sebagai berikut (Puspaningrum and Indriyanti 2022):

a. Menentukan C_i yaitu subkriteria atau tingkat kepentingan yang menjadi dasar *decision maker* dalam mengambil keputusan pada tahap akhir penelitian.

b. Menentukan matriks rating kecocokan tiap-tiap alternatif dengan bentuk matriks keputusan (x).

$$x = \begin{bmatrix} x_{11} & \cdots & x_{1\ldots n} \\ \vdots & \ddots & \vdots \\ x_{3\ldots n} & \cdots & x_{3\ldots n} \end{bmatrix} \quad (5)$$

Keterangan:

x : Nilai pada alternatif

n : Banyaknya elemen

c. Tahap ini merupakan tahap yang membutuhkan normalisasi matriks keputusan (*x*) ke skala yang dapat dibandingkan dengan semua peringkat alternatif (Sari 2021). Untuk membuat matriks ternormalisasi, diawali dengan membuat matriks keputusan berdasarkan kriteria (C_i), lalu melakukan perhitungan dengan rumus yang disesuaikan dengan tiap atribut tersebut.

$$R_{ij} \begin{cases} \dfrac{X_{ij}}{max_i \ X_{ij}} & \text{jika } j \text{ adalah atribut benefit} \\ \dfrac{min_j \ X_{ij}}{X_{ij}} & \text{jika } j \text{ adakah atribut } cost \end{cases} \quad (6)$$

Keterangan:

R_{ij} : Nilai alternatif ternormalisasi

X_{ij} : Nilai Bobot Parameter

max_i : Nilai kriteria yang paling besar pada tabel

min_i : Nilai kriteria yang paling kecil pada tabel

Benefit : Nilai yang paling besar pada alternatif

Cost : Nilai yang paling kecil pada alternatif

d. Tahap akhir dari metode SAW adalah keputusan yang didapatkan dari proses perangkingan. Untuk mendapatkan ranking, dilakukan perhitungan dengan rumus:

$$V_i \sum_{j=1}^{n} w_j r_{ij} \quad (7)$$

Keterangan:

V_i : Nilai penjumlahan akhir alternatif

w_j : Bobot kriteria

r_{ij} : Normalisasi matriks

Sehingga menghasilkan *output* dengan nilai paling besar yang menjadikan nilai tersebut menjadi alternatif yang terbaik untuk dipilih (A_1) sebagai keputusan pada tahap akhir (Wantoro 2020).

2.4 *Agile Software Development*

Agile adalah metodologi *software development* yang dimulai dengan fase perencanaan awal hingga fase perilisan, serta menggunakan prinsip pengembangan skala pendek yang mempunyai kecepatan adaptasi yang tinggi (Alexandes, Aditio, and Jumaryadi 2022). *Software Development Life Cycle* (SDLC) adalah kegiatan yang mengartikan, membangun, melakukan tes uji, mengirimkan, menjalankan, dan memelihara perangkat lunak atau sebuah sistem informasi. *Agile* merupakan salah satu metode yang sering digunakan (Nova, Widodo, and Warsito 2022).

Tujuan dari metode *Agile* adalah mengurangi biaya tambahan dengan cara memberikan kemampuan perangkat lunak untuk beradaptasi dengan perubahan tanpa membahayakan proses atau tanpa mengulang pekerjaan yang berlebihan. Sejalan dengan tujuan, terdapat juga keuntungan utama yang didapatkan jika memakai Agile Software Development yaitu meningkatkan komunikasi dan koordinasi diantara anggota tim, perilisan yang cepat, desain yang fleksibel dan proses yang memiliki dasar (Al-Saqqa, Sawalha, and Abdelnabi 2020).

2.4.1 Nilai Metodologi

Menurut (Osterwalder 2015), *Agile Software Development* memiliki empat buah nilai yaitu:

a. Lebih mementingkan interaksi antar individu daripada pemrosesan dan penggunaan alat (*Individuals And Interactions Over Processes And Tools*).

b. Lebih mementingkan pemakaian dokumen yang sangat diperlukan saja (*Working Software Over Comprehensive Documentation*).

c. Lebih mementingkan kolaborasi antara tim dan pengguna dibandingakan negosiasi atas kontrak untuk menghasilkan guna menghasilkan hasil yang bernilai (*Customer Collaboration Over Contract Negotiation*).

d. Lebih mementingkan fleksibilitas terhadap perubahan kondisi naik atau turunnya pasar dibandingkan persiapan tetapi bukan berarti tanpa persiapan awal (*Responding To Change Over Following A Plan*).

Dalam pengembangan perangkat lunak, *Agile Software Development* menggunakan pendekatan/tahapan yang berulang-ulang sehingga dapat menjadikan metode *Agile Software Development* ini memiliki kemampuan beradaptasi yang cepat. Seperti metode *Waterfall* yang menggunakan pendekatan pengembangan perangkat lunak yang terencana dengan baik, pendekatan seperti ini sangat baik digunakan pada perangkat lunak yang besar dan kompleks, namun menurut (Al-Saqqa et al. 2020) dibalik keunggulan tersebut terdapat kelemahan yakni perencanaan proyek harus diselesaikan terlebih dahulu, persyaratan perangkat lunak yang tertulis sehingga fleksibilitas berkurang, dan sepenuhnya desain untuk memenuhin persyaratan tertulis.

2.4.2 Prinsip Metodologi

Sebelum menggunakan *Agile Software Development*, sebaiknya mengikuti prinsip yang ada pada metodologi tersebut. Dalam *Agile Software Development* terdapat 12 prinsip yaitu (GHEORGHE, GHEORGHE, and IATAN 2020):

a. *Satisfy customers through early and continuous delivery of valuable work* – Pelanggan lebih senang ketika mereka menerima perangkat lunak yang berfungsi dan diimbangi dengan pembangunan berkala.

b. *Break big work down into smaller tasks that can be completed quickly – Agile* adalah tentang melakukan jumlah yang tepat dari sesuatu pada waktu tertentu. Tentu tidak kurang dan tidak lebih.

c. *Recognize that the best work emerges from self-organized teams* – Tim memiliki cara terbaik untuk menyelesaikan permasalahan. Mereka adalah para ahlinya tapi tentunya dengan bantuan anggota tim lainnya.

d. *Provide motivated individuals with the environment and support they need and trust them to get the job done* – Tim harus memberikan dukungan kepada anggota tim, karena yang termotivasi lebih cenderung memberikan hasil terbaik mereka.

e. *Create processes that promote sustainable efforts* – Dapat menciptakan proses yang menunjukkan usaha yang terus berlanjut.

f. *Maintain a constant pace for completed* work – Tim dapat menentukan kecepatan langkah yang pasti dan dipertahankan sebagai upaya memberikan perangkat lunak yang dapat berfungsi.

g. *Welcome changing requirements, even late in a project* – Proses *agile* memanfaatkan perubahan yang cepat untuk berkembang, bahkan perubahan yang telat pun dapat dimanfaatkan.

h. *Assemble the project team and business owners on a daily basis throughout the project* – Pertemuan harian harus diatur untuk mengembangkan perangkat lunak yang terbaik serta memberikan perkembangannya dari awal sampai akhir.

i. *Have the team reflect at regular intervals on how to become more effective, then tune and adjust behavior accordingly* – Peningkatan diri, peningkatan proses, peningkatan keterampilan, dan teknik membantu anggota tim bekerja lebih efisien.

j. *Measure progress by the amount of completed work* – Semua anggota proyek harus ingat bahwa kemajuan adalah untuk memberikan proyek yang bekerja dan memuaskan dan fitur lainnya

adalah untuk menciptakan lingkungan yang didorong oleh pengembangan Perangkat Lunak.

k. *Continually seek excellence* – Kita perlu memperhatikan keunggulan teknis dan desain seiring perkembangan produk kita..

l. *Harness change for a competitive advantage* – Dapat memanfaatkan perubahan pada proses *Agile* sebagai keunggulan

2.5 *Hypertext Preprocessor* (PHP)

PHP pada awalnya merupakan bahasa pemrograman yang pengelolaan datanya diproses oleh komputer server yang digunakan untuk pengembangan web serta sebagai bahasa pemrograman umum. PHP juga merubah *website* menjadi lebih interaktif dengan merubahnya dari statis menjadi dinamis (Tejasukmana Putra, Adi Wibowo, and Agus Pranoto 2021).

PHP berperan penting dalam *website* yang dinamis karena PHP mampu mengolah data dari komputer *client* atau dari komputer *server* sehingga mudah diproses di browser (Mubarak 2019).

2.6 *Unified Modelling Language* (UML)

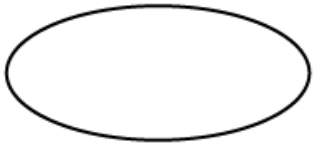
UML adalah alat yang membantu dan populer di dunia pengembangan perangkat lunak. Hal ini dikarenakan UML adalah alat bantu yang dibangun untuk *development* berorientasi objek dan desain. Sehingga dapat digunakan sebagai alat pembuat *blueprint* pengembangan yang akan dibuat serta mudah dipahami dan dipelajari (Febriani 2020).

2.6.1 *Use Case Diagram*

Use Case Diagram adalah diagram yang mendefinisikan hubungan antara *Actor* dengan sistem informasi yang sedang dibangun dan dapat menjelaskan bagaimana sistem dapat membangun relasi dengan dunia luar (Santoso and Diana 2020). Dalam *Use Case Diagram* terdapat komponen-komponen yang menyusunnya, yaitu:

a. *Use Case*

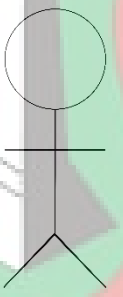
Use case merupakan mendefinisikan tentang tindakan yang menghasilkan *output* dari aktor. *Use case* digambarkan dengan bentuk elips horizontal.



Gambar 2.1 *Use Case*

b. Aktor

Aktor adalah use case yang berperan sebagai subjek yang berinteraksi dengan sistem dengan tujuan bertukar informasi. Aktor bukan mewakili individu atau pekerjaan tunggal namun memiliki peran yang dimiliki oleh seorang pengguna yang berinteraksi dengan sistem.



Gambar 2.2 Aktor

c. *Relationship*

Relationship adalah relasi antara *use case* dengan aktor yang direpresentasikan dalam bentuk garis. Dalam relationship ada relasi antara dua *use case* atau lebih yang berupa:

1. Association adalah simbol yang menggambarkan relasi dari aktor dengan use case.
2. Generalization adalah simbol yang menggambarkan spesialisasi actor untuk dapat terhubung dengan use case.
3. *Include*, menggambarkan satu *use case* termasuk dalam fungsi *use case* lain dan digambarkan dengan garis panah dengan tulisan <<*include*>>.

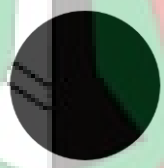
4. *Extend*, menggambarkan satu *use case* adalah fungsi tambahan dari *use case* lain jika kondisi tersebut sudah terpenuhi dan digambarkan dengan garis panah dengan tulisan <<*extend*>>.

2.6.2 *Activity Diagram*

Activity Diagram adalah sebuah diagram yang memodelkan alur data dan aksi dari suatu sistem saat bekerja. *Activity Diagram* dimulai dari administrator yang merupakan *single user* yang mengatur sistem (Muhharam Triayudi Mardiani, Eri, Alfian 2021). Untuk membentuk *activity diagram* diperlukan komponen seperti berikut:

a. Status Awal

Status awal adalah komponen yang menggambarkan awal dari *activity diagram*. Komponen ini ditandai dengan bentuk bulat dan berisi warna hitam.



Gambar 2.3 Status Awal

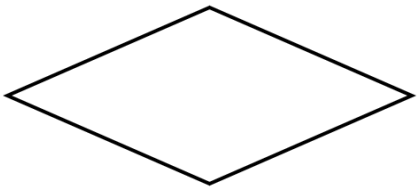
b. Aktivitas

Aktivitas adalah kegiatan yang dilakukan atau kejadian yang terjadi di dalam sebuah perangkat lunak. Penggambaran aktivitas berupa persegi dengan pinggir oval.

Gambar 2.4 Aktivitas

c. Percabangan

Percabangan adalah salah satu kegiatan yang terjadi pada sebuah perangkat lunak. Pada tahap ini, sistem dihadapkan pada sebuah keputusan dimana kondisi yang dihasilkan dapat berubah.



Gambar 2.5 Percabangan

d. Penggabungan

Penggabungan adalah *flow* yang sudah dipecah menjadi beberapa bagian namun disatukan menjadi *flow* seperti awal.

Gambar 2.6 Penggabungan

e. Status Akhir

Status akhir adalah kegiatan yang menandakan langkah akhir dari perangkat lunak. Penggambaran pada tahap ini seperti pada status awal namun ditambahkan cincin merah yang menandakan proses berakhir.

Gambar 2.7 Status Akhir

2.6.3 *Class Diagram*

Class diagram adalah sebuah diagram yang memodelkan struktur sistem *class*, metode, atribut dan relasi antar objek (Yanto 2021). Dalam pembuatan Class Diagram terdapat komponen-komponen yang digunakan yaitu:

a. Komponen atas

Komponen atas merupakan nama dari *class* yang akan dibangun dan memiliki nama-nama yang berbeda baik itu dalam klasifikasi atau objek.

b. Komponen tengah

Komponen tengah merupakan atribut dari *class* yang akan dibangun dan juga mempresentasikan kualitas dari *class*. Komponen tengah ini dibangun dengan memasukann tipe nilai pada *class*.

c. Komponen bawah

Komponen bawah merupakan operasi atau aksi yang akan dijalankan dari *class* yang akan dibangun. Sehingga komponen bawah dapat mewakilkan bagaimana *class* berinteraksi dengan data.

2.7 Pendidikan dan pengembangan (Dikbang)

Pendidikan dan pengembangan atau disebut Dikbang merupakan pendidikan lanjutan setelah pendidikan dan pembentukan. Dikbang diadakan bertujuan untuk meningkatkan kompetensi, pengetahuan dan keterampilan pegawai negeri pada Polri yang terdiri dari Anggota Polri dan Aparatur Sipil Negara (ASN) yang bekerja pada Polri. Dalam melaksanakan seleksi Dikbang menggunakan fungsi pendukung yaitu (Perkap Nomor 99 Tahun 2020):

a. Psikologi Berkala

Psikologi berkala merupakan tes yang dilakukan untuk mengetahui gambaran tingkatan personel pada Polri dengan berdasarkan tes kecerdasan, kepribadian, sikap kerja dan kondisi emosi.

b. Kesamaptaan Jasmani (Kesjas)

Kesamaptaan jasmani adalah tes yang berperan untuk mengetahui kekuatan fisik personel pada Polri baik kesamaptaan A (Tes lari selama 12 menit) dan kesamaptaan B (Tes *pull up, sit up, push up* dan *shuttle run*), maupun kemampuan bela diri personel Polri.

c. Rohani

Rohani adalah tes yang berperan untuk mengetahui kualitas rohani personel pada Polri yang berdasarkan ibadah keseharian,

kepedulian sosial, ahlak dan moral, toleransi dalam agama dan kehidupan bermasyarakat.

d. Akademik

Akademik adalah tes yang menggambarkan kualitas akademik dari personel pada Polri yang diuji berdasarkan Tes Kompetensi Tes Kompetensi Manajerial, Tes Potensi Akademik, Naskah Karya Perorangan, dan pengetahuan umum.

e. Kesehatan Berkala

Kesehatan berkala adalah tes yang berperan untuk mengetahui kondisi kesehatan personel pada Polri baik kesehatan luar dan kesehatan dalam yang diuji secara klinis (Perkap Nomor 7 Tahun 2013).

f. Pencatatan Personel

Pencatatan personel adalah tes yang dilakukan oleh Divisi Profesi dan Pengamanan Polri (Divpropam Polri) untuk mengetahui *track record* dari pegawai pada Polri (Perkap Nomor 13 Tahun 2016).

g. Sistem Manajemen Kinerja (SMK)

SMK adalah nilai yang diberikan oleh pimpinan berdasarkan kontrak kerja dan nilai SMK digunakan untuk mengukur kinerja pegawai pada Polri agar sesuai dengan visi/misi Polri (Perpol Nomor 2 Tahun 2018).