

BAB II

TINJAUAN PUSTAKA

2.1. Neurodiversitas

2.1.1. Definisi *Neurodivergent*

Neurodivergent mengacu pada individu yang memiliki pola pikir dan perilaku yang berbeda dari standar neurotipikal dalam masyarakat. Kondisi-kondisi yang termasuk dalam kategori neurodivergensi antara lain *Autism Spectrum Disorder* (ASD), *Attention Deficit Hyperactivity Disorder* (ADHD), dispraksia, disleksia, dan sindrom Tourette (Grant et al., 2025).

Individu dengan neurodivergensi kerap menghadapi stigma sosial, yang salah satunya termanifestasi melalui penggunaan bahasa dan representasi dalam komunikasi sehari-hari. Stigma berbasis bahasa ini dapat berupa penggunaan terminologi yang reduktif, pelabelan negatif, atau narasi yang menekankan defisit alih-alih perbedaan. Penelitian menunjukkan bahwa representasi linguistik yang stigmatif dapat memengaruhi persepsi publik, aksesibilitas layanan, serta kesejahteraan psikologis individu *neurodivergent* (Grummt, 2024). Oleh karena itu, pemahaman mendalam terhadap dinamika sistem pembelajaran dalam konteks neurodivergensi menjadi penting untuk mendorong inklusivitas dan mengurangi diskriminasi.

2.1.2. Karakteristik Mahasiswa ADHD

Mahasiswa yang terdiagnosis ADHD menunjukkan adanya bukti kuat bahwa gejala-gejala khas dari kondisi *neurodivergent* ini berkorelasi erat dengan tingkat stres yang tinggi. Hal ini menggarisbawahi dampak signifikan ADHD

terhadap kesejahteraan psikologis dan keseimbangan emosional individu yang mengalaminya. Tingkat stres tersebut juga dapat dikaitkan dengan berbagai tuntutan yang muncul saat memasuki jenjang Pendidikan Tinggi, mengingat konteks universitas mensyaratkan mahasiswa untuk memiliki kemampuan fungsi eksekutif dan kemandirian yang memadai, serta memori kerja dan keterampilan perencanaan studi yang baik, aspek-aspek yang umumnya belum berkembang optimal pada individu dengan ADHD, sehingga berpengaruh terhadap performa akademik mereka (Álvarez-Godos et al., 2023).

Lebih lanjut, pada fase kehidupan baru ini, mahasiswa dengan ADHD menghadapi tantangan ganda: pertama, gejala-gejala yang melekat pada gangguan itu sendiri; kedua, adaptasi terhadap lingkungan sosial yang berbeda, yaitu kampus, di mana mereka berhadapan dengan berbagai godaan dan gangguan (distraksi), sementara pengawasan orang tua berkurang dibandingkan dengan tahapan pendidikan sebelumnya. Berbagai hambatan yang mereka hadapi dalam proses adaptasi meliputi tuntutan kemandirian yang lebih besar, peningkatan sumber distraksi, penyesuaian terhadap struktur dan jadwal perkuliahan yang baru, serta berkurangnya sistem dukungan yang tersedia.

2.1.3. Karakteristik Mahasiswa Autisme (ASD)

Sebuah penelitian menggunakan data survei daring yang melibatkan mahasiswa sarjana dari 14 institusi perguruan tinggi untuk menganalisis hubungan antara kondisi autisme dengan enam indikator kesehatan fisik dan mental, dengan mempertimbangkan keberadaan komorbiditas berupa disabilitas lain (seperti *learning disability*, ADHD, gangguan sensorik, gangguan mobilitas, gangguan

kesehatan mental, atau disabilitas lainnya). Identifikasi status autisme didasarkan pada pendaftaran akomodasi disabilitas terkait autisme dan/atau laporan diri mengenai diagnosis autisme. Variabel kesehatan yang diukur mencakup persepsi terhadap kesehatan fisik, persepsi terhadap kesehatan mental, gejala depresi, gejala kecemasan, tingkat kurang tidur, dan perilaku konsumsi alkohol berlebihan (Li et al., 2019).

Temuan penelitian menunjukkan bahwa mahasiswa dengan autisme melaporkan persepsi kesehatan fisik dan mental yang lebih rendah, tingkat gejala depresi yang lebih tinggi, serta gejala kecemasan yang lebih intens dibandingkan dengan mahasiswa lainnya, bahkan setelah mengontrol variabel disabilitas lain. Hal ini mengindikasikan bahwa tantangan kesehatan mental yang dihadapi mahasiswa autistik tidak semata-mata dapat dijelaskan oleh komorbiditas dengan kondisi lain, melainkan berkaitan spesifik dengan karakteristik autisme itu sendiri (Li et al., 2019).

Menariknya, mahasiswa dengan autisme menunjukkan kecenderungan yang lebih rendah terhadap kurang tidur dan perilaku konsumsi alkohol berlebihan dibandingkan kelompok mahasiswa lainnya. Pola ini dapat dijelaskan melalui beberapa faktor, antara lain:

1. Kecenderungan individu autistik untuk mempertahankan rutinitas yang terstruktur, termasuk jadwal tidur yang konsisten.
2. Sensitivitas sensorik yang dapat membuat mereka lebih menghindari lingkungan sosial yang bising atau penuh stimulus seperti pesta.

3. Preferensi terhadap aktivitas soliter yang mengurangi paparan terhadap tekanan sosial untuk terlibat dalam perilaku berisiko seperti *binge drinking*.

2.1.4. Karakteristik Mahasiswa Disleksia

Mahasiswa dengan disleksia menunjukkan sejumlah karakteristik khas yang dapat memengaruhi pengalaman akademik mereka di perguruan tinggi. Secara umum, mereka mengalami kesulitan dalam pemrosesan fonologis (linguistik) yang berdampak pada kecepatan dan akurasi membaca, sehingga memerlukan waktu lebih lama untuk menyelesaikan tugas-tugas yang melibatkan literasi intensif seperti membaca jurnal, *textbook*, atau menulis esai akademik (Li et al., 2019).

Dari aspek kognitif, mahasiswa dengan disleksia sering menghadapi hambatan dalam memori kerja verbal, kesulitan mengeja kata-kata kompleks, dan kecenderungan melakukan kesalahan dalam penulisan meskipun telah menjalani pendidikan formal yang panjang. Mereka juga dapat mengalami kesulitan dalam memproses informasi tertulis dengan cepat, yang dapat memengaruhi performa dalam ujian berbatas waktu atau tugas dengan *deadline* ketat.

Namun, penting untuk dicatat bahwa disleksia tidak berkaitan dengan tingkat intelegensi. Banyak mahasiswa dengan disleksia memiliki kemampuan berpikir kritis, kreativitas, dan pemecahan masalah yang baik, bahkan di atas rata-rata.

Tantangan yang dihadapi mahasiswa dengan disleksia di lingkungan perguruan tinggi juga mencakup aspek psikososial, seperti kecemasan terkait performa akademik, kekhawatiran akan stigma, dan kebutuhan untuk mengadvokasi diri sendiri dalam memperoleh akomodasi akademik yang sesuai.

Dukungan institusional yang memadai, seperti waktu tambahan dalam ujian, akses terhadap materi dalam format alternatif, dan layanan konseling, menjadi krusial untuk mendukung kesuksesan akademik mereka.

2.2. Sistem Pakar

2.2.1. Definisi Sistem Pakar

Sistem Pakar (*Expert System*) merupakan salah satu cabang kecerdasan buatan (*artificial intelligence*) yang dirancang untuk meniru cara seorang pakar dalam menyelesaikan suatu permasalahan pada bidang tertentu. Sistem ini bekerja dengan menyimpan pengetahuan seorang pakar ke dalam basis pengetahuan (*knowledge base*), kemudian menggunakan pengetahuan tersebut untuk menarik kesimpulan atas suatu permasalahan melalui mekanisme penalaran (*inference*). Dengan demikian, sistem pakar memungkinkan pengguna non-pakar untuk memperoleh saran atau kesimpulan yang menyerupai hasil pemikiran seorang ahli (Sukiakhy et al., 2022).

Dalam konteks penelitian ini, sistem pakar digunakan untuk merekomendasikan strategi belajar yang sesuai dengan karakteristik mahasiswa *neurodivergent*. Pengetahuan psikolog mengenai keterkaitan antara karakteristik belajar dan strategi yang tepat direpresentasikan ke dalam basis pengetahuan, sehingga sistem dapat memberikan rekomendasi secara mandiri kepada mahasiswa tanpa harus selalu berkonsultasi langsung dengan pakar. Penting untuk ditekankan bahwa sistem ini berfungsi sebagai alat bantu peningkatan kesadaran (*awareness*), bukan sebagai alat diagnosis klinis.

2.2.2. Komponen Sistem Pakar

Sebuah Sistem Pakar yang efektif terdiri dari empat komponen utama yang saling terintegrasi untuk menghasilkan sebuah output:

1. Basis Pengetahuan (*Knowledge Base*): Basis pengetahuan merupakan komponen inti yang berisi pengetahuan dari pakar dalam bentuk fakta dan kaidah (*rule*). Pada penelitian ini, basis pengetahuan memuat karakteristik belajar mahasiswa *neurodivergent* serta kaidah yang menghubungkan karakteristik tersebut dengan strategi belajar yang sesuai, beserta nilai keyakinan (*Certainty Factor*) dari pakar
2. Mesin Inferensi (*Inference Engine*): Mesin inferensi adalah komponen yang melakukan penalaran terhadap fakta dan kaidah dalam basis pengetahuan untuk menghasilkan kesimpulan. Pada sistem ini, mesin inferensi menggunakan metode *Forward Chaining*, yaitu penalaran maju yang dimulai dari fakta-fakta (jawaban mahasiswa) untuk menuju kesimpulan berupa rekomendasi strategi belajar.
3. Basis Data (*Database*): Basis data menyimpan data yang diperlukan oleh sistem, seperti daftar pertanyaan, daftar strategi belajar, kaidah, serta nilai *Certainty Factor*. Komponen ini mendukung kerja mesin inferensi dalam mengakses pengetahuan yang dibutuhkan.
4. Antarmuka Pengguna (*User Interface*): Antarmuka pengguna merupakan bagian yang menjembatani interaksi antara pengguna (mahasiswa) dengan sistem. Melalui antarmuka ini, mahasiswa menjawab pertanyaan yang

diajukan sistem dan menerima hasil berupa rekomendasi strategi belajar beserta tingkat keyakinannya.

2.2.3. Forward Chaining sebagai Metode Inferensi

Forward Chaining merupakan salah satu metode penalaran dalam sistem pakar yang bekerja dengan cara penelusuran maju (*data-driven*), yaitu dimulai dari sekumpulan fakta yang diketahui, kemudian dicocokkan dengan kaidah-kaidah yang ada untuk mencapai suatu kesimpulan. Metode ini menelusuri kaidah berbentuk JIKA (*IF*) – MAKA (*THEN*), di mana bagian JIKA memuat kondisi (fakta) dan bagian MAKA memuat kesimpulan (Sapriadi et al., 2023).

Pada penelitian ini, fakta diperoleh dari jawaban mahasiswa atas pertanyaan mengenai karakteristik belajarnya. Setiap fakta yang terpenuhi akan mengaktifkan kaidah tertentu yang mengarah pada rekomendasi strategi belajar. Sebagai contoh, jika mahasiswa menyatakan sering kesulitan mempertahankan fokus, maka kaidah yang merekomendasikan teknik *Pomodoro* akan aktif. Proses ini berlangsung hingga seluruh fakta selesai ditelusuri dan menghasilkan satu atau beberapa rekomendasi strategi belajar.

2.3. Certainty Factor

2.3.1. Definisi Certainty Factor

Certainty Factor (CF) atau faktor kepastian merupakan suatu metode yang digunakan untuk mengukur tingkat keyakinan terhadap suatu kesimpulan dalam sistem pakar. Metode ini pertama kali diperkenalkan oleh Shortliffe dan Buchanan pada sistem pakar MYCIN untuk menangani permasalahan yang mengandung

ketidakpastian (*uncertainty*), yaitu ketika suatu kesimpulan tidak dapat dinyatakan benar atau salah secara mutlak (Nugraha et al., 2024)

Dalam penerapannya, *Certainty Factor* memberikan nilai kepastian pada rentang tertentu untuk menggambarkan seberapa yakin sistem terhadap suatu kesimpulan. Nilai kepastian ini diperoleh dari kombinasi antara keyakinan pakar terhadap suatu kaidah (*CF pakar*) dan keyakinan pengguna terhadap gejala atau kondisi yang dialaminya (*CF pengguna*). Dengan pendekatan ini, sistem tidak hanya menghasilkan kesimpulan berupa "sesuai" atau "tidak sesuai", tetapi juga menyertakan derajat keyakinan yang membuat rekomendasi menjadi lebih informatif.

Pada penelitian ini, *Certainty Factor* digunakan untuk menghitung tingkat keyakinan terhadap kesesuaian antara karakteristik belajar mahasiswa *neurodivergent* dengan strategi belajar yang direkomendasikan. Penggunaan metode ini dinilai tepat karena karakteristik belajar setiap mahasiswa bersifat individual dan tidak selalu pasti, sehingga rekomendasi yang dihasilkan perlu disertai tingkat keyakinan, bukan sekadar keputusan mutlak.

2.3.2. Nilai CF Pakar dan CF Pengguna

Dalam metode *Certainty Factor*, terdapat dua sumber nilai keyakinan yang saling dikombinasikan, yaitu nilai keyakinan dari pakar (*CF pakar*) dan nilai keyakinan dari pengguna (*CF pengguna*).

Nilai *CF pakar* merupakan bobot keyakinan yang diberikan oleh pakar terhadap setiap kaidah dalam basis pengetahuan. Nilai ini menggambarkan seberapa kuat hubungan antara suatu karakteristik belajar dengan strategi belajar yang

direkomendasikan. Pada penelitian ini, nilai *CF pakar* diperoleh dari hasil wawancara dengan psikolog serta dukungan studi literatur, dengan rentang nilai antara 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan keyakinan yang semakin kuat.

Sementara itu, nilai *CF pengguna* merupakan tingkat keyakinan yang diberikan oleh mahasiswa terhadap gejala atau karakteristik yang dialaminya. Nilai ini diperoleh dari jawaban mahasiswa atas pertanyaan yang diajukan sistem. Agar mudah dipahami oleh pengguna, pilihan jawaban disajikan dalam bentuk skala frekuensi yang kemudian dikonversi menjadi nilai numerik, sebagaimana ditunjukkan pada Tabel 2.1.

Tabel 1. 1 Konversi Skala Jawaban ke Nilai CF Pengguna

Pilihan Jawaban	Nilai CF Pengguna
Tidak pernah	0.0
Jarang	0.25
Kadang-kadang	0.5
Sering	0.75
Selalu	1.0

Melalui skala tersebut, mahasiswa cukup memilih jawaban yang paling menggambarkan kondisinya, tanpa perlu memahami perhitungan numerik di baliknya. Sistem selanjutnya akan mengonversi setiap jawaban menjadi nilai *CF pengguna* yang digunakan dalam proses perhitungan.

2.3.3. Rumus dan Perhitungan Certainty Factor

Perhitungan nilai *Certainty Factor* pada penelitian ini dilakukan melalui beberapa tahap. Tahap pertama adalah menghitung nilai CF untuk setiap kaidah, yaitu dengan mengalikan nilai keyakinan pakar (CF_{pakar}) dengan nilai keyakinan pengguna ($CF_{pengguna}$). Secara matematis, perhitungan ini dinyatakan dalam persamaan berikut:

Rumus 1 — CF untuk satu kaidah:

$$CF(H,E)=CF_{pakar}\times CF_{pengguna} \quad (1)$$

dengan:

$CF(H,E)$ = nilai kepastian terhadap hipotesis H (strategi belajar) berdasarkan evidence E (karakteristik mahasiswa);

CF_{pakar} = nilai keyakinan pakar terhadap kaidah;

$CF_{pengguna}$ = nilai keyakinan pengguna terhadap karakteristik yang dialaminya.

Pada kondisi tertentu, satu strategi belajar dapat direkomendasikan oleh lebih dari satu kaidah sekaligus. Sebagai contoh, strategi *Pomodoro* dapat dipicu baik oleh karakteristik kesulitan mempertahankan fokus maupun oleh karakteristik mudah terdistraksi. Apabila hal ini terjadi, nilai CF dari masing-masing kaidah perlu digabungkan menggunakan rumus kombinasi berikut:

Rumus 2 — Kombinasi dua nilai CF:

$$CF_{kombinasi}=CF_1+CF_2\times(1-CF_1) \quad (2)$$

dengan:

CF_1 = nilai CF dari kaidah pertama;

CF_2 = nilai CF dari kaidah kedua.

Apabila terdapat lebih dari dua kaidah yang menghasilkan kesimpulan yang sama, proses kombinasi dilakukan secara bertahap. Hasil kombinasi dari dua kaidah pertama ($CF_{kombinasi}$) kemudian dikombinasikan kembali dengan nilai CF kaidah berikutnya, dan seterusnya, hingga seluruh kaidah yang relevan tergabung. Persamaan untuk tahap lanjutan ini dinyatakan sebagai berikut:

Rumus 3 — Kombinasi bertahap:

$$CF_{kombinasi\ baru} = CF_{kombinasi\ lama} + CF_n \times (1 - CF_{kombinasi\ lama}) \quad (3)$$

Nilai akhir $CF_{kombinasi}$ inilah yang menunjukkan tingkat keyakinan sistem terhadap suatu strategi belajar. Nilai tersebut selanjutnya dapat dikonversi ke dalam bentuk persentase dengan mengalikannya dengan 100%, sehingga lebih mudah dipahami oleh pengguna. Strategi belajar dengan nilai $CF_{kombinasi}$ tertinggi akan menjadi rekomendasi utama bagi mahasiswa.

2.3.4. Contoh Penerapan Certainty Factor

Untuk memberikan gambaran yang lebih jelas mengenai cara kerja metode *Certainty Factor*, berikut disajikan sebuah contoh perhitungan sederhana. Misalkan seorang mahasiswa menjawab pertanyaan yang diajukan sistem, dan dari jawabannya diperoleh dua karakteristik belajar yang keduanya mengarah pada rekomendasi strategi *Pomodoro*, sebagaimana ditunjukkan pada Tabel 2.2.

Tabel 2. 2 Contoh Penerapan Certainty Factor

Kaidah	Karakteristik (Evidence)	Jawaban Pengguna	CF Pengguna	CF Pakar
R1	Sulit mempertahankan fokus	Sering	0.75	0.85
R2	Mudah terdistraksi	Selalu	1.0	0.65

A. Langkah 1: Menghitung CF setiap kaidah (Rumus 1)

Nilai CF untuk masing-masing kaidah dihitung dengan mengalikan CF pakar dan CF pengguna:

$$CF(R1) = 0.85 \times 0.75 = 0.6375$$

$$CF(R2) = 0.65 \times 1.0 = 0.65$$

B. Langkah 2: Mengombinasikan kedua nilai CF (Rumus 2)

Karena kedua kaidah menghasilkan rekomendasi yang sama (*Pomodoro*), nilai CF keduanya digabungkan:

$$CF_{kombinasi} = CF_1 + CF_2 \times (1 - CF_1)$$

$$CF_{kombinasi} = 0.6375 + 0.65 \times (1 - 0.6375)$$

$$CF_{kombinasi} = 0.6375 + 0.65 \times 0.3625$$

$$CF_{kombinasi} = 0.6375 + 0.2356$$

$$CF_{kombinasi} = 0.8731$$

C. Langkah 3: Konversi ke persentase

Nilai akhir dikonversi ke bentuk persentase:

$$CF = 0.8731 \times 100\% = 87.31\%$$

Berdasarkan perhitungan tersebut, sistem menyimpulkan bahwa strategi *Pomodoro* memiliki tingkat keyakinan sebesar 87.31% sebagai strategi belajar yang sesuai untuk mahasiswa tersebut. Nilai ini lebih tinggi dibandingkan jika hanya satu karakteristik yang terpenuhi, yang menunjukkan bahwa semakin banyak bukti yang mendukung suatu rekomendasi, semakin tinggi pula tingkat keyakinan sistem terhadap rekomendasi tersebut.

2.4. Strategi Belajar

2.4.1. Definisi Strategi Belajar

Strategi belajar merupakan sekumpulan tindakan, prosedur, atau teknik yang digunakan oleh individu secara sengaja untuk memfasilitasi perolehan, penyimpanan, dan penggunaan informasi dalam proses akademik. Weinstein dan Mayer (1986) mendefinisikan strategi belajar sebagai perilaku dan proses berpikir yang digunakan siswa untuk memengaruhi cara mereka mengolah informasi selama pembelajaran.

Dalam konteks pendidikan tinggi, strategi belajar tidak hanya mencakup teknik kognitif seperti merangkum, mencatat, atau menghafal, tetapi juga mencakup aspek metakognitif (perencanaan dan pemantauan proses belajar), manajemen diri (pengaturan waktu dan motivasi), serta pengaturan lingkungan belajar agar kondusif bagi proses kognitif.

Bagi mahasiswa *neurodivergent*, pemilihan strategi yang tepat sangat menentukan keberhasilan dalam menghadapi beban tugas yang kompleks di

perguruan tinggi. Hal ini disebabkan oleh perbedaan mendasar dalam fungsi eksekutif (kemampuan merencanakan, mengorganisir, dan menyelesaikan tugas), regulasi emosi (kemampuan mengelola stres dan kecemasan akademik), serta pemrosesan sensorik (sensitivitas terhadap rangsangan visual, auditori, atau taktil). Oleh karena itu, strategi belajar untuk mahasiswa *neurodivergent* perlu bersifat personal, adaptif, dan fleksibel sesuai dengan profil kognitif dan sensorik masing-masing individu.

2.4.2. Jenis-jenis Strategi Belajar untuk *Neurodivergent*

Mahasiswa *neurodivergent* sering kali memerlukan strategi belajar yang lebih spesifik dan adaptif dibandingkan mahasiswa neurotipikal. Berdasarkan karakteristik umum yang dialami oleh mahasiswa dengan ADHD, ASD, dan Disleksia, berikut adalah strategi-strategi belajar yang relevan dan akan menjadi alternatif dalam sistem pakar ini:

1. Teknik Pemecahan Tugas (*Chunking*)

Chunking adalah strategi membagi tugas besar yang terasa memberatkan menjadi sub-tugas yang lebih kecil dan terkelola. Strategi ini sangat efektif untuk mahasiswa dengan ADHD yang sering mengalami *overwhelmed* ketika menghadapi tugas kompleks atau *deadline* yang panjang. Dengan memecah tugas menjadi bagian-bagian kecil, mahasiswa dapat merasakan pencapaian (*sense of accomplishment*) secara bertahap, yang membantu menjaga motivasi dan mengurangi prokrastinasi.

Contoh penerapan: Alih-alih target "menyelesaikan skripsi Bab 1 dalam seminggu", target dipecah menjadi "menulis 1 paragraf latar belakang hari ini", "mencari 3 jurnal besok", dan seterusnya.

2. *Visual Mapping/Mind Mapping*

Strategi ini menggunakan representasi visual untuk mengorganisir ide dan melihat hubungan antar konsep. *Mind mapping* sangat membantu mahasiswa dengan ADHD yang kesulitan dalam pemrosesan informasi linear, serta mahasiswa dengan Disleksia yang lebih mudah memahami informasi dalam bentuk visual dibanding teks panjang. Penggunaan warna, simbol, dan cabang-cabang visual membantu otak mengingat informasi dengan lebih efektif.

Contoh penerapan: Membuat *mind map* untuk merangkum konsep kuliah alih-alih membuat catatan berbentuk esai panjang.

3. *Pomodoro Technique* (Modifikasi Adaptif)

Teknik Pomodoro standar menggunakan interval 25 menit kerja dan 5 menit istirahat. Untuk mahasiswa *neurodivergent*, teknik ini dapat dimodifikasi sesuai rentang fokus individual, misalnya 15 menit kerja untuk mahasiswa ADHD dengan distraktibilitas tinggi, atau 40 menit untuk mahasiswa yang membutuhkan waktu lebih lama untuk "masuk" ke dalam fokus (*hyperfocus*). Penggunaan *timer visual* atau alarm dapat membantu mengatasi *time blindness* yang umum dialami mahasiswa ADHD.

Contoh penerapan: Menggunakan aplikasi *timer* dengan notifikasi visual dan suara yang dapat disesuaikan dengan preferensi sensorik masing-masing mahasiswa.

4. *Body Doubling* (Virtual atau Fisik)

Body doubling adalah strategi bekerja atau belajar bersama orang lain (baik secara fisik maupun virtual) untuk meningkatkan akuntabilitas dan fokus, tanpa harus berinteraksi secara intensif. Strategi ini sangat efektif untuk mahasiswa ADHD yang membutuhkan "*anchor*" eksternal untuk mempertahankan fokus, serta untuk mahasiswa ASD yang lebih nyaman dengan kehadiran orang lain tanpa tekanan untuk berinteraksi sosial yang intens.

Contoh penerapan: Menggunakan platform seperti Focusmate atau *study together session* di Discord, di mana mahasiswa belajar bersama dalam mode kamera tanpa harus berbicara.

5. *Noise Management* (Pengaturan Lingkungan Auditori)

Strategi ini melibatkan pengaturan input suara untuk meminimalkan gangguan atau sebaliknya, menciptakan *white noise* yang membantu fokus. Mahasiswa dengan sensitivitas sensorik tinggi (umum pada ASD) mungkin memerlukan lingkungan yang sangat tenang dengan menggunakan *noise-cancelling headphones*, sementara mahasiswa ADHD mungkin justru lebih fokus dengan *background noise* seperti *lo-fi music* atau *ambient sounds*.

Contoh penerapan: Menggunakan aplikasi *white noise*, *brown noise*, atau *lo-fi beats*; atau memilih lokasi belajar seperti perpustakaan (tenang) vs kafe (dengan *background noise*).

6. *Structured Scheduling* (Penjadwalan Terstruktur dengan *Buffer Time*)

Strategi ini melibatkan pembuatan jadwal harian atau mingguan yang sangat terstruktur dengan alokasi waktu yang jelas untuk setiap aktivitas, disertai *buffer*

time untuk mengatasi kesulitan transisi antar tugas. Mahasiswa ASD yang membutuhkan rutinitas dan prediktabilitas akan sangat terbantu dengan strategi ini. *Buffer time* juga penting untuk mahasiswa ADHD yang sering *underestimate* (menyepelekan) waktu yang dibutuhkan untuk menyelesaikan tugas.

Contoh penerapan: Menggunakan *calendar blocking* dengan warna berbeda untuk setiap jenis aktivitas, serta menambahkan 15-30 menit *buffer* antara tugas-tugas besar.

7. *Assistive Technology* (Teknologi Bantu untuk Literasi)

Strategi ini memanfaatkan teknologi seperti *text-to-speech*, *speech-to-text*, atau *dyslexia-friendly* font untuk mengurangi beban kognitif saat membaca atau menulis. Mahasiswa dengan Disleksia akan sangat terbantu dengan *tools* yang dapat membacakan teks jurnal atau buku, sehingga mereka dapat fokus pada pemahaman konten tanpa terbebani oleh kesulitan *decoding* kata.

Contoh penerapan: Menggunakan aplikasi seperti NaturalReader untuk membacakan PDF jurnal, atau Grammarly untuk membantu penulisan dengan koreksi otomatis.

8. *Movement Breaks* (Istirahat Aktif dengan Gerakan)

Strategi ini melibatkan pengintegrasian gerakan fisik singkat di antara sesi belajar untuk melepaskan energi berlebih dan meningkatkan regulasi sensorik. Mahasiswa ADHD yang mengalami *hyperactivity* atau *restlessness* akan sangat terbantu dengan istirahat aktif seperti *stretching*, berjalan singkat, atau *fidgeting* menggunakan *fidget tools*.

Contoh penerapan: Setiap selesai 1 Pomodoro session, melakukan 5 menit *jumping jacks*, *stretching*, atau berjalan mengelilingi ruangan.

2.5. Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan bahasa pemodelan visual yang digunakan dalam merancang dan mengembangkan perangkat lunak berorientasi objek. UML berperan sebagai standar penulisan cetak biru (*blueprint*) suatu sistem, yang di dalamnya mencakup penggambaran proses bisnis hingga penulisan struktur kelas dengan notasi yang telah dibakukan (Ramdany et al., 2024). Keberadaan UML sebagai bahasa pemodelan yang terstandarisasi oleh *Object Management Group* (OMG) menjadikannya salah satu pendekatan yang paling banyak diadopsi dalam pengembangan perangkat lunak hingga saat ini, karena mampu membantu proses analisis kebutuhan sekaligus menerjemahkannya ke dalam rancangan desain yang independen terhadap teknologi maupun metodologi tertentu (Wayahdi & Ruziq, 2023).

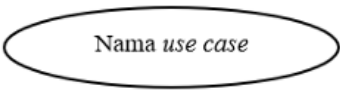
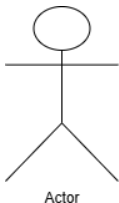
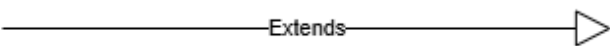
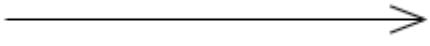
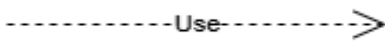
Dalam praktiknya, UML tidak hadir sebagai satu diagram tunggal, melainkan sekumpulan diagram yang masing-masing menyoroti sisi berbeda dari suatu system, mulai dari sisi interaksi pengguna, alur proses, urutan komunikasi antar komponen, hingga struktur data yang mendasarinya. Pada penelitian ini, digunakan empat jenis diagram UML untuk menggambarkan rancangan sistem pakar secara menyeluruh, yaitu *use case diagram* untuk menggambarkan interaksi pengguna dengan sistem, *activity diagram* untuk menggambarkan alur aktivitas sistem, *sequence diagram* untuk menggambarkan urutan interaksi antar komponen, dan *class diagram* untuk menggambarkan struktur data dalam system.

2.5.1. Use Case Diagram

Use case diagram merupakan model hasil analisis perancangan sistem yang bertujuan menggambarkan kebutuhan sistem dari sudut pandang pengguna. Diagram ini bekerja dengan mendeskripsikan interaksi tipikal antara pengguna dan sistem melalui gambaran bagaimana sistem tersebut akan digunakan (Ramdany et al., 2024). Berbeda dengan diagram lain yang lebih menyoroti proses internal, *use case diagram* justru berfokus pada apa yang tampak dari luar sistem. *Use case diagram* mengomunikasikan fungsi-fungsi tingkat tinggi (*high-level*) serta ruang lingkup sistem, sehingga pihak di luar sistem dapat memahami ekspektasi yang bisa diperoleh dari sistem tersebut (Wayahdi & Ruziq, 2023).

Terdapat dua elemen utama yang membentuk *use case diagram*, yaitu *aktor* dan *use case* itu sendiri. Aktor merepresentasikan entitas apa pun yang berinteraksi langsung dengan sistem, baik berupa manusia (yang digambarkan berdasarkan perannya, bukan identitas personalnya) maupun sistem lain di luar sistem yang sedang dirancang. Sementara itu, *use case* menggambarkan satu fungsi yang disediakan sistem dan menghasilkan keluaran yang bermakna bagi aktor tersebut. Identifikasi terhadap kedua elemen ini pada akhirnya membentuk apa yang disebut batas sistem (*system boundary*), yaitu sebuah garis pemisah yang menegaskan mana tugas yang menjadi tanggung jawab sistem dan mana yang berada di luar cakupannya (Wayahdi & Ruziq, 2023). Pada penelitian ini, *use case diagram* digunakan untuk menggambarkan peran mahasiswa sebagai pengguna utama yang berkonsultasi dengan sistem, serta peran admin/psikolog yang mengelola basis pengetahuan di baliknya.

Tabel 2. 3 Use Case Diagram (Ramdany et al., 2024)

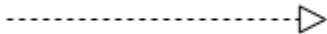
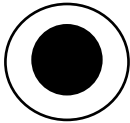
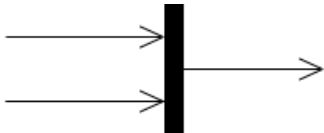
Simbol	Deskripsi
<i>Use Case</i> 	Menggambarkan fungsi yang disediakan sistem
Aktor 	Menggambarkan pengguna yang berinteraksi dengan sistem
Asosiasi/<i>association</i> 	Menghubungkan actor dengan use case
Ekstensi/<i>extend</i> 	Use case tambahan yang bersifat opsional terhadap use case utama
Generalisasi/<i>generalization</i> 	Hubungan umum-khusus antar use case
Menggunakan/<i>include</i> 	Use case yang wajib dijalankan sebagai bagian dari use case utama

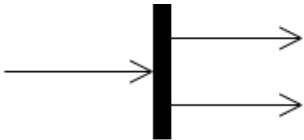
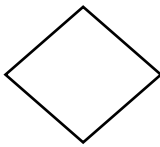
2.5.2. Activity Diagram

Activity diagram merupakan salah satu diagram UML yang paling sering digunakan dibandingkan diagram lainnya, karena kemampuannya menggambarkan urutan tindakan yang dieksekusi sekaligus merepresentasikan aliran data di dalam suatu proses (Wayahdi & Ruziq, 2023). Secara umum, diagram ini menggambarkan aliran fungsional dalam suatu sistem informasi secara menyeluruh dan mendefinisikan di mana suatu alur kerja (*workflow*) dimulai, di mana ia berakhir, aktivitas apa saja yang terjadi selama alur tersebut berlangsung, serta bagaimana urutan kejadian dari aktivitas-aktivitas itu saling berkaitan satu sama lain (Ramdany et al., 2024).

Yang membedakan *activity diagram* dari diagram alur konvensional adalah sifatnya yang hierarkis: satu aktivitas besar dapat tersusun dari serangkaian tindakan yang lebih kecil, atau bahkan dari sub-diagram aktivitas lain yang saling terhubung melalui aliran objek (Wayahdi & Ruziq, 2023). Diagram ini juga cocok digunakan baik untuk desain tingkat tinggi, seperti pemodelan proses bisnis secara umum, maupun untuk desain tingkat rendah yang mendeskripsikan algoritma secara lebih rinci (Wayahdi & Ruziq, 2023). Pada penelitian ini, *activity diagram* digunakan untuk menggambarkan alur konsultasi mahasiswa dengan sistem, mulai dari menjawab pertanyaan karakteristik belajar, diproses melalui mekanisme *Forward Chaining* dan *Certainty Factor*, hingga akhirnya rekomendasi strategi belajar ditampilkan kembali kepada mahasiswa.

Tabel 2. 4 Activity Diagram (Ramdany et al., 2024)

Simbol	Deskripsi
<i>Activity</i> 	Menggambarkan satu aktivitas/proses
<i>Control Flow</i> 	Menggambarkan aliran/urutan aktivitas
<i>Object Flow</i> 	Menggambarkan aliran objek dari sebuah <i>action</i> atau <i>activity</i> ke <i>action</i>
<i>Start Point</i> 	Menggambarkan aktivitas yang memiliki sebuah status awal
<i>End Point</i> 	Menandakan akhir aktivitas
Penggabungan/ <i>join</i> 	Untuk menggabungkan kembali <i>activity</i> atau <i>action</i> yang paralel

Pencabangan/ <i>Fork</i> 	Untuk memecah <i>behavior</i> menjadi <i>activity</i> atau <i>action</i> yang paralel
<i>Decision</i> 	Menggambarkan titik percabangan Keputusan di mana ada kemungkinan perbedaan transisi

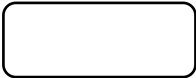
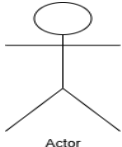
2.5.3. *Sequence Diagram*

Sequence diagram merupakan diagram UML paling umum digunakan setelah *use case diagram*, karena kemampuannya merepresentasikan bagaimana objek-objek dalam suatu sistem saling berinteraksi dan bertukar pesan berdasarkan urutan waktu (*over time*) (Wayahdi & Ruziq, 2023). Melalui diagram ini, dapat terlihat bagaimana suatu peristiwa yang telah didefinisikan pada *use case diagram* dipetakan menjadi operasi-operasi konkret pada kelas objek yang ada di *class diagram*, sehingga *sequence diagram* pada dasarnya menjembatani gambaran fungsional (*use case*) dengan gambaran struktural (*class diagram*) ke dalam satu alur waktu yang runtut.

Penerimaan yang luas terhadap *sequence diagram* dalam praktik pemodelan sistem tidak lepas dari sifatnya yang relatif intuitif untuk dibaca, serta kemampuannya menggambarkan perilaku sistem secara parsial tanpa harus menampilkan keseluruhan logika sekaligus (Wayahdi & Ruziq, 2023). Inti dari diagram ini terletak pada penggambaran interaksi antar objek beserta indikasi hubungan yang terjadi di antaranya (Ramdany et al., 2024) dengan tujuan utamanya

adalah mengungkap bagaimana objek-objek tersebut saling berkolaborasi untuk mewujudkan satu fungsi tertentu, sebagaimana yang telah dibayangkan oleh perancang sistem sejak tahap desain (Wayahdi & Ruziq, 2023). Pada penelitian ini, *sequence diagram* digunakan untuk menggambarkan interaksi antara mahasiswa, antarmuka, mesin inferensi, dan basis data selama satu sesi konsultasi berlangsung, sehingga proses inferensi yang telah dijelaskan secara naratif pada bab sebelumnya dapat ditelusuri secara lebih runtut berdasarkan urutan waktunya.

Tabel 2. 5 Sequence Diagram

Simbol	Deskripsi
Objek 	Menyatakan objek yang berinteraksi (nama objek ditempatkan di dalamnya)
Aktor 	Dapat berupa manusia, sistem lain, atau perangkat yang berinteraksi dengan sistem yang dibuat.
<i>Lifeline</i> 	Menggambarkan objek dan masa hidupnya selama proses

<i>Activation</i> 	Menandakan objek sedang aktif memproses
Pesan 	Menggambarkan komunikasi antar objek dengan objek

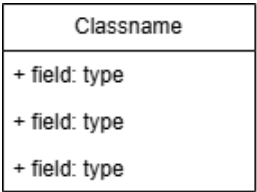
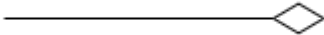
2.5.4. Class Diagram

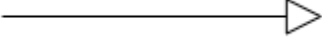
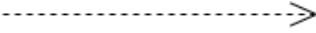
Class diagram menggambarkan struktur serta penjelasan rinci dari setiap kelas yang terdapat dalam model desain suatu sistem, termasuk aturan dan tanggung jawab entitas yang menentukan perilaku sistem tersebut (Ramdany et al., 2024). Dapat dikatakan bahwa *class diagram* adalah representasi visual dari struktur program pada jenis-jenis objek yang dibentuknya dan juga menggambarkan alur data yang akan dibangun dalam sistem sekaligus menjadi kumpulan dari beberapa kelas beserta relasinya satu sama lain. Setiap kelas pada dasarnya identik dengan sebuah entitas, yang digambarkan dalam bentuk persegi dengan tiga bagian: nama kelas pada bagian atas, atribut yang dimiliki kelas tersebut di tengah, dan metode atau operasi yang dapat dijalankannya di bagian bawah.

Peran *class diagram* berbeda pada tiap tahap pengembangan sistem. Pada tahap analisis, diagram ini menunjukkan aturan dan tanggung jawab entitas yang membentuk perilaku sistem; sedangkan pada tahap desain, ia berperan menangkap struktur keseluruhan kelas yang membentuk arsitektur sistem, sehingga membantu visualisasi struktur kelas secara menyeluruh (Ramdany et al., 2024). Karena

sifatnya yang mampu memberikan pandangan global atas suatu sistem dan tercermin dari kelas-kelas yang ada beserta relasinya, *class diagram* sering disebut sebagai jenis diagram UML yang paling banyak digunakan dan menjadi dasar penting bagi pengembangan sistem perangkat lunak pada tahap selanjutnya. Pada penelitian ini, *class diagram* digunakan untuk menggambarkan struktur data Pertanyaan, Strategi, Kaidah, dan Mesin Inferensi, beserta relasi yang menghubungkan keempatnya dalam basis pengetahuan sistem pakar.

Tabel 2. 6 Class Diagram (Ramdany et al., 2024)

Simbol	Deskripsi
<i>Class</i> 	Menampilkan nama kelas, atribut, dan operasi
<i>Association</i> 	Menghubungkan antar kelas yang saling berelasi
<i>Aggregation</i> 	Hubungan keseluruhan-bagian antar kelas
<i>Composition</i> 	Kelas yang tidak bisa berdiri sendiri tanpa kelas induknya

<i>Generalization</i> 	Hubungan umum-khusus antar kelas
<i>Depency</i> 	Satu kelas menggunakan/bergantung pada kelas lain