BAB 2
TINJAUAN PUSTAKA

2.1 Tidur

Tidur merupakan salah satu kebutuhan dasar manusia yang esensial untuk kesehatan fisik dan mental. Tidur memiliki peran penting dalam proses pemulihan tubuh, pembentukan memori, dan regulasi emosi. Menurut Max Hirshkowitz, tidur adalah keadaan perilaku yang ditandai oleh inaktivitas relatif, responsivitas yang lebih rendah terhadap rangsangan eksternal, dan pola gelombang otak tertentu [10].

2.2 Fase-Fase Tidur

Tidur dibagi menjadi dua kategori utama, yaitu tidur non-REM (Non-Rapid Eye Movement) dan tidur REM (Rapid Eye Movement). Tidur non-REM terdiri dari tiga tahap:

1. Tahap 1: Periode transisi antara terjaga dan tidur, yang berlangsung sekitar 5-10 menit.
2. Tahap 2: Tidur yang lebih dalam, ditandai dengan penurunan suhu tubuh dan detak jantung, berlangsung sekitar 20 menit.
3. Tahap 3: Tidur paling dalam, dikenal sebagai tidur gelombang lambat atau tidur delta, yang sangat penting untuk pemulihan fisik.

Tidur REM, yang terjadi sekitar 90 menit setelah tertidur, ditandai dengan mimpi intens dan aktivitas otak yang mirip dengan keadaan terjaga. Tahap ini penting untuk pemulihan mental dan emosional.

2.3 Gangguan Tidur

Gangguan tidur, seperti insomnia, sleep apnea, dan restless legs syndrome, dapat berdampak negatif pada kesehatan dan kualitas hidup. Sleep apnea, yang ditandai oleh berhenti bernapas secara berulang selama tidur, dapat menyebabkan penurunan saturasi oksigen dan gangguan tidur yang signifikan. Menurut T Young, prevalensi sleep apnea obstruktif di antara populasi dewasa adalah sekitar 2-4% [11].

2.4 Kualitas Tidur

Kualitas tidur adalah suatu keadaan yang menunjukkan adanya kemampuan masyarakat untuk tidur dan memperoleh jumlah istirahat sesuai dengan kebutuhannya. Kualitas tidur adalah suatu keadaan tidur yang dijalani seseorang individu menghasilkan kesegaran dan kebugaran saat terbangun [12].

Kualitas tidur dipengaruhi oleh durasi tidur, kontinuitas tidur, dan struktur tidur. Durasi tidur yang ideal bervariasi berdasarkan usia dan kebutuhan individu, tetapi rata-rata orang dewasa membutuhkan sekitar 7-8 jam tidur per malam [13]. Kontinuitas tidur mengacu pada seberapa sering tidur terganggu sepanjang malam. Struktur tidur mencakup proporsi waktu yang dihabiskan dalam setiap tahap tidur.

2.4.1 Faktor Yang Mempengaruhi Kualitas Tidur

Menurut banyak sumber, kualitas tidur memiliki dampak besar pada kesejahteraan fisik dan mental kita sehari-hari. Tidur yang baik mempengaruhi energi, suasana hati, kinerja kognitif, dan kesehatan secara keseluruhan. Namun, berbagai faktor dapat memengaruhi kemampuan kita untuk tidur dengan nyenyak. Berikut merupakan faktor-faktor yang mempengaruhi kualitas tidur:

1. Lingkungan Tidur

Secara khusus, lingkungan tidur memiliki peran penting dalam menentukan kualitas tidur seseorang. Faktor seperti kebisingan dapat mengganggu tidur, baik itu berasal dari luar seperti lalu lintas atau suara tetangga, maupun dari dalam rumah seperti suara elektronik atau suara mendengkur. Paparan cahaya terang, terutama cahaya biru dari layar elektronik, juga dapat mengganggu ritme tidur alami tubuh dan mengganggu kualitas tidur. Selain itu, suhu dan kelembaban ruangan yang tidak nyaman juga dapat mempengaruhi kenyamanan tidur seseorang.

2. Kebiasaan Tidur

Kebiasaan tidur juga memainkan peran penting dalam kualitas tidur seseorang. Jadwal tidur yang tidak teratur dapat mengganggu jam biologis tubuh dan mengurangi kualitas tidur. Konsumsi kafein dan alkohol dapat mengganggu proses tidur dan mempengaruhi kualitas tidur seseorang. Aktivitas fisik yang teratur dapat membantu

meningkatkan kualitas tidur, tetapi olahraga yang terlalu intensif sebelum tidur dapat mengganggu proses tidur. Pola makan juga dapat mempengaruhi kualitas tidur, dengan konsumsi makanan berat atau pedas sebelum tidur yang dapat memengaruhi pencernaan dan kenyamanan tidur seseorang.

3. Kesehatan Mental dan Fisik

Kesehatan mental dan fisik seseorang juga berperan dalam kualitas tidur. Stres dan kecemasan dapat menyebabkan gangguan tidur, sementara gangguan mental seperti depresi, kecemasan, atau insomnia dapat mempengaruhi kualitas tidur. Kondisi fisik seperti sleep apnea atau nyeri fisik juga dapat membuat seseorang sulit tidur dengan nyenyak.

2.4.2 Manfaat Kualitas Tidur

Terdapat beberapa manfaat dari meningkatkan kualitas tidur. Manfaat-manfaat tersebut antara lain:

1. Meningkatkan Konsentrasi dan Produktivitas

Manfaat istirahat dan tidur yang bisa langsung anda rasakan yaitu tubuh terasa lebih benergi dan fit keesokan harinya. Selain itu, cukup tidur juga baik untuk menjaga fungsi otak. Hal ini tentu akan membantu anda lebih produktif, lebih fokus, dan lebih konsentrasi dalam menyelesaikan pekerjaan sehari-hari.

2. Meningkatkan Suasana Hati

Istirahat dan tidur yang berkualitas juga dapat menyegarkan kembali pikiran dan suasana hati, sehingga lebih siap menjalani aktivitas dan menyelesaikan pekerjaan. Saat tidur, otak memproses emosi dan pikiran agar dapat bereaksi dengan baik. Itulah sebabnya bila tidak cukup tidur, emosi atau suasana hati cenderung memberikan reaksi negatif. Misalnya, akan lebih mudah murung, mudah lelah, atau merasa cemas.

3. Meningkatkan Daya Ingat

Manfaat istirahat dan tidur selanjutnya adalah memudahkan mengingat hal-hal secara detail. Tanpa istirahat dan tidur yang cukup, akan sulit fokus dalam menerima informasi baru dan otak pun tidak memiliki cukup waktu untuk mengingat sesuatu dengan baik.

4. Memperkuat Sistem Kekebalan Tubuh

Tidur dan istirahat yang cukup juga berperan dalam meningkatkan daya tahan tubuh. Dengan imunitas yang kuat, tubuh mampu melawan berbagai infeksi kuman secara maksimal sehingga tidak mudah terserang penyakit.

5. Mengontrol Kadar Gula Darah

Kurang istirahat atau tidur dapat memiliki efek kesehatan yang berkaitan dengan metabolisme tubuh dan salah satunya adalah peningkatan kadar gula darah. Hal ini tentunya dapat meningkatkan risiko terkena diabetes.

6. Menjaga Berat Badan Tetap Ideal

Tidur dan istirahat yang cukup juga bermanfaat untuk menjaga berat badan tetap ideal. Berbagai penelitian membuktikan bahwa kurang tidur bisa mengganggu keseimbangan hormon yang mengatur nafsu makan dalam tubuh.

7. Menjaga Kesehatan Jantung

Saat tidur, tubuh melepaskan hormon yang menjaga jantung dan pembuluh darah tetap sehat. Selain itu, tekanan darah juga menjadi lebih stabil karena jantung dan pembuluh darah dapat beristirahat selama tidur.

2.4.3 Meningkatkan Kualitas Tidur Dengan Menciptakan Lingkungan Tidur Yang Mendukung

Kualitas tidur sangat dipengaruhi oleh lingkungan tidur. Lingkungan tidur yang nyaman dan mendukung dapat membantu seseorang mendapatkan tidur yang lebih nyenyak dan berkualitas. Penelitian menunjukkan bahwa beberapa faktor lingkungan, seperti suhu kamar, kebisingan, pencahayaan, dan kualitas tempat tidur, memainkan peran penting dalam menentukan seberapa baik seseorang tidur. Berikut adalah beberapa aspek lingkungan tidur yang perlu diperhatikan meliputi:

1. Suhu Kamar

Suhu kamar yang terlalu panas atau terlalu dingin dapat mengganggu tidur. National Sleep Foundation merekomendasikan suhu kamar antara 15-19°C sebagai suhu yang ideal untuk tidur yang nyenyak [14]. Suhu yang nyaman membantu tubuh mencapai kondisi homeostasis yang optimal untuk tidur.

2. Kebisingan

Kebisingan adalah salah satu faktor utama yang dapat mengganggu tidur. Penelitian menunjukkan bahwa paparan kebisingan selama tidur dapat mengurangi waktu tidur nyenyak dan meningkatkan risiko gangguan tidur [15]. Menggunakan penutup telinga atau mesin white noise dapat membantu mengurangi dampak kebisingan terhadap tidur.

3. Pencahayaan

Paparan cahaya, terutama cahaya biru dari perangkat elektronik, dapat mengganggu produksi melatonin, hormon yang mengatur tidur. Paparan cahaya biru sebelum tidur dapat memperpanjang waktu yang dibutuhkan untuk tertidur dan mengurangi durasi tidur REM [16]. Oleh karena itu, disarankan untuk mengurangi paparan cahaya dan menggunakan lampu redup di malam hari untuk menciptakan lingkungan tidur yang lebih kondusif.

4. Kualitas Tempat Tidur

Tempat tidur yang nyaman, termasuk kasur dan bantal yang mendukung, sangat penting untuk tidur yang berkualitas. Mengganti kasur yang lama dengan yang baru dapat meningkatkan kualitas tidur dan mengurangi nyeri punggung. Pemilihan kasur yang sesuai dengan preferensi masyarakat dan postur tidur dapat membantu mengurangi gangguan tidur [17].

2.4.4 Tingkat Kebisingan Terhadap Kualitas Tidur

Kebisingan lingkungan merupakan salah satu faktor yang sering kali diabaikan namun memiliki dampak signifikan terhadap kesehatan dan kualitas hidup kita. Menurut World Health Organization (WHO), dijelaskan bahwa kebisingan yang berlebihan bukan hanya gangguan sehari-hari, tetapi juga ancaman serius bagi kesehatan fisik dan mental [15]. Untuk mengetahui pengaruh tingkat kebisingan terhadap kualitas tidur akan ditunjukkan pada Tabel 2.1 Tingkat Kebisingan Terhadap Kualitas Tidur:

Tabel 2.1 Tingkat Kebisingan Terhadap Kualitas Tidur

Tingkat Kebisingan (dB)	Sumber Bunyi	Skala Intensitas	Efek Kualitas Tidur
0-30	Bisikan lembut, suara kipas angin kecil, gemerisik daun	Sangat tenang	Tidak ada efek yang signifikan. Suara pada tingkat ini biasanya tidak mengganggu tidur dan dapat menciptakan lingkungan yang nyaman untuk tidur.
30-40	Suara AC, lalu lintas yang sangat jauh, suara percakapan pelan	Tenang	Umumnya tidak mengganggu tidur, meskipun beberapa orang yang sangat sensitif terhadap suara mungkin sedikit terganggu. Suara pada tingkat ini biasanya masih dapat diterima dan tidak menyebabkan gangguan tidur yang signifikan.
40-55	Lalu lintas kota, percakapan biasa, suara dari tetangga	Bising	Mulai mengganggu tidur, terutama jika suara bersifat tiba-tiba atau berulang. Kebisingan pada tingkat ini dapat menyebabkan gangguan tidur bagi sebagian orang, seperti kesulitan untuk tertidur atau terbangun di malam hari.
≥ 55	Mesin berat, suara pesawat terbang yang lewat, konser	Sangat Bising	Mengganggu tidur secara signifikan. Kebisingan pada tingkat ini dapat menyebabkan terbangun berulang kali di malam hari dan

	musik, kembang api, lalu lintas kota yang berat		kesulitan untuk tertidur kembali. Dampak ini lebih terasa bagi individu yang sensitif terhadap suara.

2.4.5 Tingkat Suara Mendengkur Terhadap Kualitas Tidur

Mendengkur atau bisa dikenal dengan ngorok merupakan fenomena yang umum terjadi pada banyak orang dan dapat mengganggu tidur baik bagi individu yang mendengkur maupun bagi pasangan tidurnya. Menurut Maimon, Patrick dan Hanly Pada jurnal yang berjudul "Does Snoring Intensity Correlate with the Severity of Obstructive Sleep Apnea?", menjelaskan bahwa suara ngorok tidak hanya merupakan gangguan yang mengganggu kenyamanan tidur, tetapi juga dapat menjadi indikator masalah kesehatan yang lebih serius, seperti sleep apnea [18]. Untuk mengetahui pengaruh tingkat suara mendengkur terhadap kualitas tidur akan ditunjukkan pada Tabel 2.2 Tingkat Suara Mendengkur Terhadap Kualitas Tidur:

Tabel 2.2 Tingkat Suara Mendengkur Terhadap Kualitas Tidur

Tingkat Desibel Mendengkur (dB)	Skala Intensitas	Efek Kualitas Tidur
40-50	Ringan	Umumnya tidak mengganggu tidur, meskipun beberapa orang yang sangat sensitif terhadap suara mungkin sedikit terganggu.
50-60	Sedang	Mulai mengganggu tidur, terutama jika suara bersifat tiba-tiba atau berulang. Kebisingan pada tingkat ini dapat menyebabkan gangguan tidur bagi sebagian orang.

≥ 60	Berat	Sangat mengganggu tidur. Kebisingan pada tingkat ini hampir pasti menyebabkan gangguan tidur yang serius, termasuk kesulitan tertidur, terbangun berulang kali, dan tidur yang tidak nyenyak.

2.5 Mendengkur

Mendengkur adalah bunyi yang keluar akibat adanya gangguan pada saluran udara yang melewati hidung dan faring (sepanjang jalan nafas bagian atas) [19]. Mendengkur terjadi karena masuknya aliran udara pernapasan ke paru-paru terhalang. Halangan bisa berada di rongga hidung, mulut dan tenggorok. Halangan tersebut menyebabkan terjadinya penyempitan aliran udara pernapasan yang menetap atau hanya sementara.

Mendengkur merupakan masalah sosial dan kesehatan karena dapat mengganggu pasangan tidur, pergaulan, menurunnya produktivitas, meningkatnya biaya kesehatan dalam mengatasinya [20]. Mendengkur bukanlah gejala normal, terlebih jika gejala muncul secara tiba-tiba dan berlangsung terus menerus. Sebagian kasus perlu diwaspadai karena dapat menjadi petunjuk adanya masalah kesehatan [21].

2.6 Monitoring

Monitoring adalah proses menjaga atau pengawasan terhadap keberadaan dan besarnya perubahan keadaan dan arus data dalam sebuah sistem. *Monitoring* bertujuan untuk mengidentifikasi kesalahan dan kemajuan dalam menentukan keputusan selanjutnya. Teknik yang digunakan dalam monitoring informasi sistem memotong bidang pengolahan real-time, statistik, dan analisis data. Satu set komponen perangkat lunak yang digunakan untuk pengumpulan data, pengolahan, dan presentasi disebut sistem *monitoring*.

Monitoring juga merupakan elemen kunci dalam pemantauan kesehatan, termasuk *monitoring* kualitas tidur. Dengan menggunakan teknologi yang tepat, *monitoring* kualitas tidur dapat dilakukan secara efisien dan non-invasif, memberikan informasi

yang berharga bagi individu maupun profesional kesehatan untuk menjaga kesehatan dan kesejahteraan.

2.7 Kebisingan

Bising adalah suara yang tidak diinginkan dari usaha atau kegiatan dalam tingkat dan waktu tertentu yang dapat menimbulkan gangguan kesehatan manusia dan kenyamanan lingkungan. Pada umumnya kebisingan sangat berkaitan dengan ketergangguan (*annoyance*). Kebisingan ada di mana-mana dan ketergangguan adalah salah satu reaksi yang paling umum terhadap bising [22].

2.8 Gelombang Suara

Bunyi atau suara merupakan gejala fisis yang dihasilkan karena adanya suatu getaran. Bunyi termasuk gelombang longitudinal yang merambat melalui suatu medium. Energi yang terkandung dalam bunyi dapat meningkat secara cepat dan menempuh jarak yang sangat jauh [22]. Bunyi diidentikkan sebagai pergerakan gelombang di udara yang terjadi jika sumber bunyi mengubah partikel terdekat dan posisi menjadi partikel yang bergerak [22]. Beberapa karakteristik utama dari gelombang suara meliputi:

1. Amplitudo

Amplitudo merupakan tingkat kekuatan gelombang suara, yang berkaitan dengan volume atau intensitas suara. Amplitudo yang lebih tinggi menghasilkan suara yang lebih keras.

2. Frekuensi

Frekuensi merupakan jumlah siklus gelombang suara yang terjadi dalam satu detik, diukur dalam *hertz* (Hz). Frekuensi gelombang suara menentukan nada atau *pitch* suara. Frekuensi yang lebih tinggi menghasilkan suara yang lebih tinggi.

3. Panjang Gelombang

Panjang gelombang merupakan jarak antara dua titik yang berdekatan pada gelombang suara yang identik, diukur dalam meter. Panjang gelombang invers terhadap frekuensi, artinya gelombang dengan frekuensi tinggi memiliki panjang gelombang yang lebih pendek.

2.9 Smartphone

Smartphone adalah sebuah alat elektronik interaktif yang mengerti perintah sederhana yang dikirim oleh pengguna dan membantu dalam aktivitas sehari-hari. Beberapa perangkat pintar yang paling umum digunakan adalah *smartphone*, *tablet*, *smartwatch*, kacamata pintar dan peralatan elektronik pribadi lainnya. Sementara banyak perangkat pintar kecil, elektronik pribadi portabel, perangkat tersebut sebenarnya ditentukan oleh kemampuan mereka untuk terhubung ke jaringan untuk berbagi dan berinteraksi dari jarak jauh. Banyak TV dan kulkas juga dianggap perangkat cerdas.

2.10 Sensor Smartphone

Sensor *smartphone* adalah komponen elektronik yang terintegrasi dalam perangkat *smartphone* untuk mendeteksi, mengukur, dan memonitor berbagai parameter fisik dan lingkungan di sekitarnya. Sensor-sensor ini memungkinkan perangkat untuk mengumpulkan data tentang gerakan, orientasi, posisi, dan lingkungan sekitar, yang kemudian dapat digunakan untuk berbagai aplikasi, termasuk pemantauan kesehatan.

2.10.1 Sensor Mikrofon

Sensor mikrofon pada *smartphone* adalah perangkat keras yang berfungsi untuk menangkap suara dan mengubahnya menjadi sinyal listrik yang daipat diproses oleh perangkat. Sensor mikrofon pada *smartphone* umumnya menggunakan teknologi elektroakustik untuk mengonversi gelombang suara menjadi sinyal listrik.

2.11 Aplikasi

Aplikasi merujuk pada program komputer yang dirancang untuk menjalankan suatu tugas atau menyediakan layanan tertentu bagi pengguna. Aplikasi dapat berupa perangkat lunak *desktop*, web, atau *mobile*, tergantung pada *platform* tempat aplikasi tersebut dijalankan. Aplikasi memiliki berbagai tujuan, mulai dari produktivitas, hiburan, pendidikan, hingga kesehatan dan kesejahteraan.

2.11.1 Aplikasi Mobile

Aplikasi *mobile* adalah jenis aplikasi yang dirancang khusus untuk dijalankan pada perangkat *mobile*, seperti *smartphone* dan *tablet*. Aplikasi *mobile* memiliki

karakteristik yang berbeda dengan aplikasi *desktop* atau web, terutama dalam hal ukuran layar, interaksi pengguna, dan kemampuan akses ke berbagai fitur perangkat *mobile*, seperti GPS, kamera, dan sensor-sensor lainnya.

2.12 Android

Android merupakan sebuah sistem operasi berbasis *Linux* dengan bersumber kode terbuka dibawah lisensi *APACHE* 2.0 yang dibuat beragam perangkat yang berbeda. *Android, Inc*. didirikan oleh Andy Rubin, Rich Miner, Nick Sears, dan Chris White yang kemudian pada tahun 2005 dibeli oleh *Google* [23].

Sifatnya sendiri adalah *open source* yang berarti bisa digunakan bebas, diperbaiki, dimodifikasi, hingga didistribusikan oleh siapa saja yang membuat atau mengembangkan perangkat lunak. Dengan sifatnya seperti itu, operasi Sistem ini bebas digunakan oleh perusahaan teknologi untuk perangkatnya secara gratis atau tanpa lisensi.

2.12.1 Arsitektur Android

Arsitektur *android* terdiri dari beberapa komponen utama yang bekerja bersama-sama untuk menyediakan lingkungan yang stabil dan aman bagi pengguna serta pengembang aplikasi. Komponen-komponen utama dalam arsitektur *android* antara lain:

1. *Linux Kernel*

   Linux Kernel merupakan bagian terbawah dari arsitektur *android* yang menyediakan layanan dasar seperti manajemen memori, manajemen proses, dan keamanan.

2. *Libraries*

   Libraries merupakan kumpulan perpustakaan (*libraries*) yang disediakan oleh *android* untuk memungkinkan pengembang untuk menggunakan berbagai fitur seperti grafika, database, jaringan, dan UI (*User Interface*).

3. *Android Runtime* (*ART*)

   Android Runtime merupakan lingkungan *runtime* yang menjalankan dan mengelola kode aplikasi.

4. *Application Framework*

Application Framework merupakan kerangka kerja (*framework*) yang menyediakan berbagai layanan yang diperlukan oleh aplikasi, seperti manajemen aktivitas (*Activity*), manajemen tampilan (*View*), manajemen notifikasi (*Notification*), dan manajemen jaringan (*Networking*).

5. *Applications*

Applications merupakan aplikasi yang dibuat oleh pengembang dan dijalankan pada sistem *android*. Aplikasi dapat diunduh dan diinstal oleh pengguna melalui *Google Play Store* atau sumber lainnya.

2.12.2 Android Lifecycle

Android Lifecycle mengacu pada serangkaian peristiwa yang terjadi selama aplikasi *android* berjalan, dari saat aplikasi dimulai hingga dihentikan atau dihancurkan. Setiap komponen dalam aplikasi *android*, seperti *Activity*, *Service*, *Broadcast Receiver*, dan *Content Provider*, memiliki siklus hidup (*lifecycle*) yang terdefinisi dengan baik.

Komponen utama *Android Lifecycle* antara lain sebagai berikut:

1. *Activity Lifecycle*

Activity adalah komponen yang menyediakan antarmuka pengguna (UI) dalam aplikasi *android*. Siklus hidup *Activity* terdiri dari sejumlah metode yang dipanggil oleh sistem *android* sesuai dengan peristiwa-peristiwa yang terjadi, seperti pembuatan, dimulainya, berhentinya, dan penghancuran *Activity*. Beberapa metode penting dalam *Activity Lifecycle* antara lain onCreate(), onStart(), onResume(), onPause(), onStop(), dan onDestroy().

2. *Service Lifecycle*

Service adalah komponen yang berjalan di latar belakang (*background*) untuk melakukan tugas-tugas yang tidak memerlukan interaksi langsung dengan pengguna. Siklus hidup *Service* mencakup metode-metode seperti onCreate(), onStartCommand(), dan onDestroy(), yang dipanggil oleh sistem *android* saat *Service* dibuat, dimulai, dan dihentikan.

3. *Broadcast Receiver Lifecycle*

Broadcast Receiver adalah komponen yang mendengarkan dan menanggapi pesan-pesan (*broadcast*) yang dikirim oleh sistem atau aplikasi lain. Siklus hidup *Broadcast Receiver* terdiri dari metode onReceive(), yang dipanggil oleh sistem *android* saat pesan broadcast diterima.

4. *Content Provider Lifecycle*

Content Provider adalah komponen yang menyediakan akses ke data dalam aplikasi kepada aplikasi lain atau sistem *android*. *Content Provider* tidak memiliki metode siklus hidup yang terdefinisi secara eksplisit, namun *Content Provider* biasanya dibuat saat aplikasi dimulai dan dihancurkan saat aplikasi dihentikan.

2.12.3 Android Studio

Android Studio merupakan *Integrated Development Environtment* atau (IDE) resmi untuk *Android* [23]. Android Studio diluncurkan resmi oleh *Google* pada tahun 2013 yang dilengkapi *Android System Development Kit* atau (SDK) yang terdiri dari *library* dan *tools* untuk *build* atau membangun aplikasi *android*. *Android Studio* memberi fitur-fitur yang bisa meningkatkan fungsionalitas dan produktifitas Dalam membuat aplikasi *android* dan juga dapat diunduh disistem operasi *Linux*, *Windows*, ataupun *Mac OS*.

2.13 Bahasa Pemrograman Kotlin

Kotlin merupakan salah satu bahasa pemrograman yang digunakan untuk membuat aplikasi berbasis *android*. *Kotlin* membawa segala keuntungan dari bahasa pemrograman modern untuk *platform asndroid* tanpa memberikan batasan baru [23]. *Kotlin* bersifat *statis-typed*, modern, dan berorientasi objek. Dikembangkan oleh perusahaan JetBrains, *Kotlin* dirancang untuk berjalan di atas mesin virtual *Java* (JVM), membuatnya kompatibel dengan aplikasi *Java*. *Kotlin* juga dapat digunakan untuk pengembangan aplikasi *android*.

2.14 Firebase

Firebase Realtime Database merupakan database realtime yang tersimpan di cloud dan support multiplatform seperti Android, iOS dan Web. Data pada firebase akan disimpan dalam struktur JSON (Java Script Object Notation). Database firebase akan melakukan sinkronisasi secara otomatis terhadap aplikasi client yang terhubung kepadanya. Aplikasi multiplatform yang menggunakan SDK Android, iOS dan JavaScript akan menerima update data terbaru secara otomatis pada saat aplikasi terhubung ke server firebase [24].

Firebase Realtime Database merupakan platform Database yang digunakan pada aplikasi realtime. Ketika terjadi perubahan data, maka aplikasi yang terhubung dengan firebase akan memperbaharui secara otomatis melalui setiap device(perangkat) baik websitea taupun mobile. Firebase mempunyai library (pustaka) yang lengkap untuk sebagian besar platform web dan mobile. Firebase dapat digabungkan dengan framework lain seperti node, java, javascript, dan lain-lain [24].

2.15 Teachable Machine

Teachable Machine adalah alat berbasis web yang dikembangkan oleh Google untuk membuat model pembelajaran mesin dengan mudah tanpa perlu pengetahuan mendalam tentang coding atau machine learning. Alat ini memungkinkan pengguna untuk melatih model menggunakan data yang mereka miliki dengan cara yang intuitif dan interaktif.

Menurut Google AI (2020), Teachable Machine dirancang untuk mempermudah proses pembuatan model pembelajaran mesin, sehingga dapat digunakan oleh berbagai kalangan, mulai dari pendidik, siswa, hingga pengembang aplikasi. Pengguna dapat membuat model untuk pengenalan gambar, suara, dan pose hanya dengan mengunggah contoh data dan melatih model secara langsung di browser.

Teachable Machine menyediakan antarmuka yang ramah pengguna dan mendukung ekspor model ke berbagai format, termasuk TensorFlow dan TensorFlow Lite, sehingga model yang telah dibuat dapat diimplementasikan dalam aplikasi web, mobile, atau embedded system.

2.16 TensorFlow

TensorFlow adalah platform open-source yang dikembangkan oleh Google Brain Team untuk pengembangan dan pelatihan model machine learning. Diperkenalkan pada tahun 2015, TensorFlow telah menjadi salah satu framework paling populer untuk pengembangan aplikasi kecerdasan buatan [25].

TensorFlow menawarkan berbagai alat dan library yang mendukung pemodelan, pelatihan, dan deployment model machine learning. Dengan arsitektur yang fleksibel, TensorFlow memungkinkan pengguna untuk menjalankan model pada berbagai perangkat, mulai dari server hingga perangkat edge.

Beberapa fitur utama TensorFlow meliputi:

1. High-Level APIs: Seperti Keras, yang memudahkan pengguna dalam membangun dan melatih model deep learning.
2. Distribusi Pelatihan: Mendukung pelatihan model secara terdistribusi pada cluster komputasi.
3. TensorBoard: Alat untuk visualisasi dan debugging pelatihan model.
4. Model Hub: Repository yang menyediakan model pra-latih yang dapat digunakan kembali oleh pengguna.

2.16.1 TensorFlow Lite

TensorFlow Lite adalah versi ringan dari TensorFlow yang dirancang khusus untuk menjalankan model machine learning pada perangkat dengan sumber daya terbatas, seperti smartphone, embedded device, dan IoT [26].

TensorFlow Lite memungkinkan deployment model machine learning dengan efisiensi tinggi dalam hal penggunaan memori dan performa, tanpa mengorbankan akurasi model. Ini dicapai melalui teknik seperti quantization dan optimization.

Beberapa fitur utama TensorFlow Lite meliputi:

1. Model Conversion: Mendukung konversi model TensorFlow menjadi format TensorFlow Lite.
2. Interpreter: Mesin runtime yang efisien untuk mengeksekusi model TensorFlow Lite pada perangkat mobile dan embedded.

3. Delegasi: Mendukung delegasi untuk akselerasi hardware menggunakan GPU dan NNAPI.

TensorFlow Lite juga menyediakan API yang memudahkan integrasi model dalam aplikasi mobile, sehingga pengembang dapat membuat aplikasi dengan kemampuan AI secara native.

2.17 Unified Modeling Language (UML)

UML adalah sebuah bahasa yang berdasar pada grafik/gambar untuk memvisualisasi, mengspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (*Object-Oriented*).

Salah satu dari tahapan SDLC atau *software development life cycle*, adalah desain. Desain bertujuan agar *software* yang akan dibuat dapat memenuhi kebutuhan user dan handal. Oleh karena itu, desain menjadi tahapan penting dalam proses pembuatan *software*. Dalam mendesain *software*, kita perlu mentransformasikan kebutuhan user, baik secara fungsional maupun non fungsional ke dalam model.

Model adalah sebuah abstraksi dari hal nyata. Model merupakan penyederhanaan dari sistem yang sebenarnya sehingga desain dari sebuah sistem dapat dimengerti oleh pihak lain. Untuk memodelkan sesuatu, tentu diperlukan bahasa pemodelan. Bahasa pemodelan dapat berupa pseudo-code, code, gambar, diagram, atau deskripsi yang menggambarkan sebuah sistem. Di sinilah UML berperan sebagai bahasa pemodelan.

2.17.1 Use Case Diagram

Use Case Diagram menggambarkan hubungan interaksi antara sistem dan aktor. *Use Case* dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya.

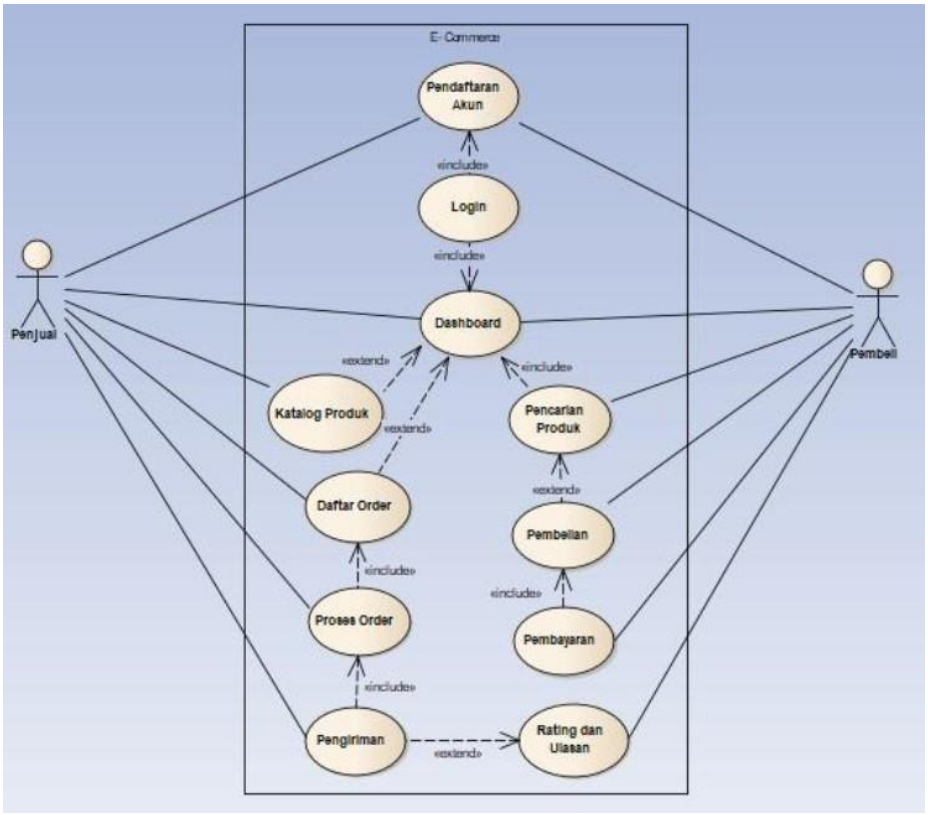
Manfaat *use case diagram* yaitu membantu kita dalam menyusun kebutuhan-kebutuhan (*requirement* analisis) sebuah sistem. *Use case diagram* akan di gunakan untuk mengkomunikasikan rancangan sistem dengan klien, dan merancang *test case* untuk semua fitur yang ada pada sistem. Berikut merupakan komponen use case diagram:

1. Aktor

Pada dasarnya aktor bukanlah bagian dari use case diagram, namun untuk dapat terciptanya suatu use case diagram diperlukan beberapa aktor. Aktor tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem.

2. Use Case

Use case adalah gambaran fungsionalitas dari suatu sistem, sehingga customer atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun. Berikut merupakan contoh *Use Case* Model yang ditunjukkan pada Gambar 2.1 Contoh Use Case Model [27].



Gambar 2.1 Contoh Use Case Model

3. Relasi dalam Use Case

Ada beberapa relasi yang terdapat pada *use case diagram*:

a. *Association*, menghubungkan link antar element
b. *Generalization*, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
c. *Dependency*, sebuah element bergantung dalam beberapa cara ke elemen lainnya.
d. *Aggregation*, bentuk *assosiation* dimana sebuah elemen berisi elemen lainnya.

Sedangkan tipe relasi/ *stereotype* yang mungkin terjadi pada use case diagram:

a. <<include>> , yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari use case lainnya.
b. <<extends>>, kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm
c. <<communicate>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association* . Ini merupakan pilihan selama asosiasi hanya tipe *relationship* yang dibolehkan antara aktor dan *use case*.

4. Use Case Diagram

Use Case Diagram adalah gambaran *graphical* dari beberapa atau semua aktor, *use case*, dan interaksi diantaranya yang memperkenalkan suatu sistem.

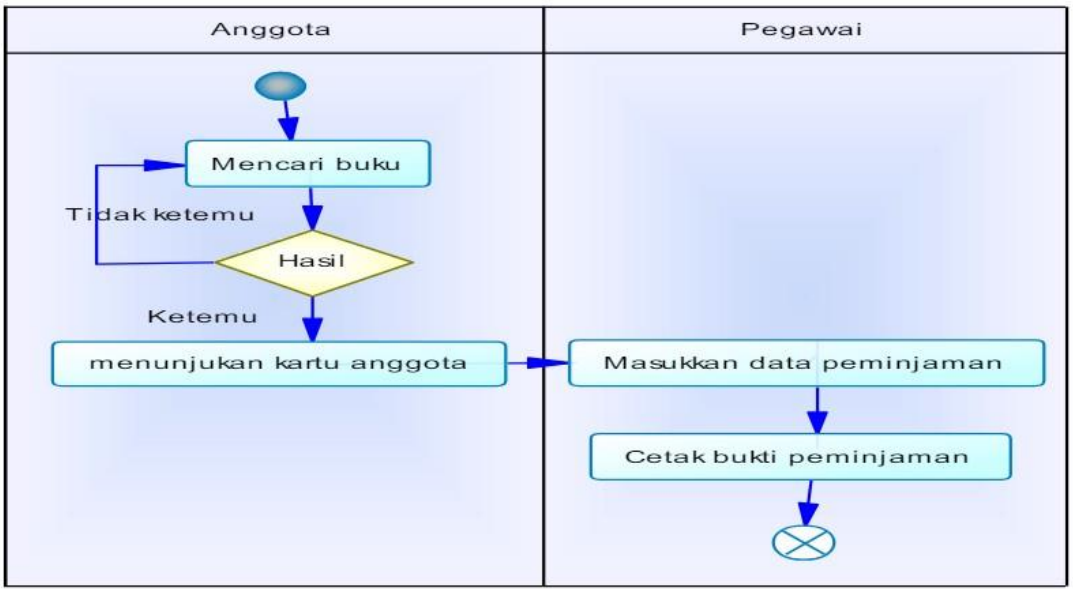
5. Batasan Sistem (*System Boundary*)

Dalam batasan sistem hal-hal yang penting yaitu:

a. *System boundary*.
b. Mendefinisikan batas antara aktor dan sistem.
c. Dinotasikan bujur sangkar/persegi.
d. Semua *use case* tercakup di dalamnya.

2.17.2 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses *parallel* yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu activity diagram tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Berikut merupakan contoh *Activity Diagram* yang ditunjukkan pada Gambar 2.2 Contoh Activity Diagram [28].



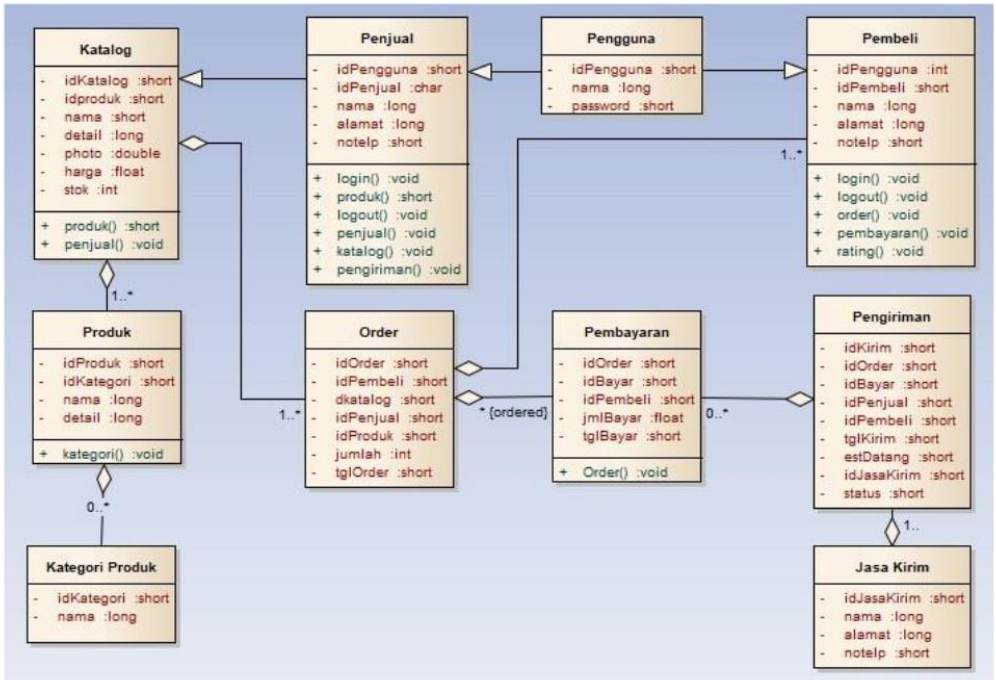
Gambar 2.2 Contoh Activity Diagram

2.17.3 Class Diagram

Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi *class*, atribut, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi.

Diagram kelas ini sesuai jika diimplementasikan ke proyek yang menggunakan konsep *object-oriented* karena gambaran dari *class diagram* cukup mudah untuk digunakan.

Desain model dari diagram kelas ini sendiri dibagi menjadi dua bagian. Bagian pertama merupakan penjabaran dari database. Bagian kedua merupakan bagian dari modul MVC, yang memiliki class interface, class control, dan class entity. Berikut merupakan contoh Class Diagram yang ditunjukkan pada Gambar 2.3 Contoh Class Diagram [27].



Gambar 2.3 Contoh Class Diagram

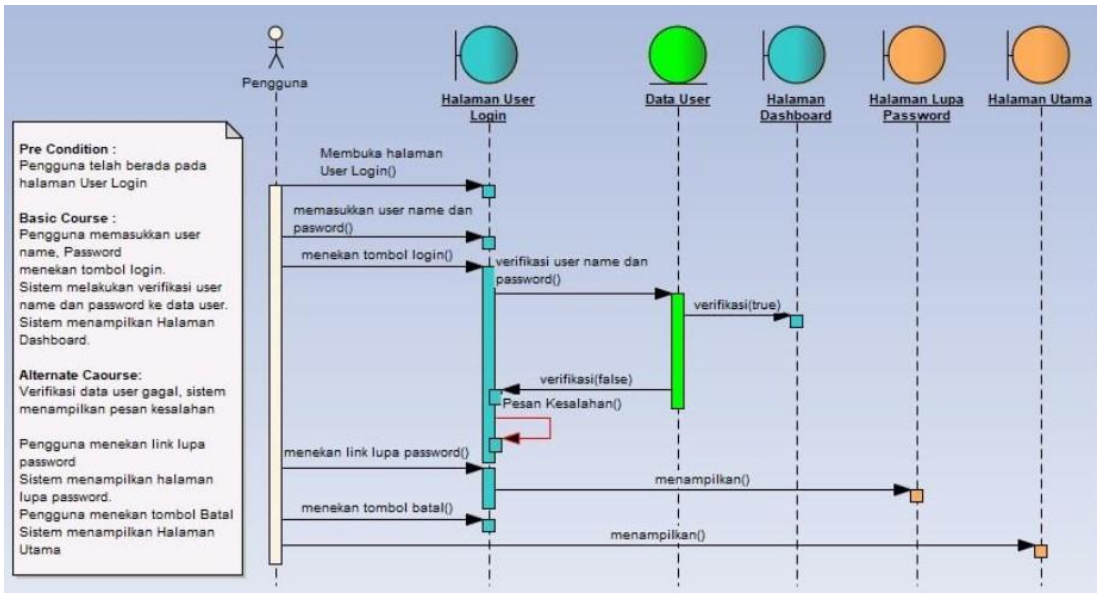
2.17.4 Sequence Diagram

Sequence diagram atau diagram urutan adalah sebuah diagram yang digunakan untuk menjelaskan dan menampilkan interaksi antar objek-objek dalam sebuah sistem secara terperinci. Selain itu *sequence diagram* juga akan menampilkan pesan atau perintah yang dikirim, beserta waktu pelaksanaannya. Objek-objek yang berhubungan dengan berjalannya proses operasi biasanya diurutkan dari kiri ke kanan.

Diagram sequence terdiri dari dua bagian, yaitu bagian vertikal yang menunjukkan waktu dan bagian horizontal yang menunjukkan objek-objek. Setiap objek, termasuk aktor, memiliki waktu aktif yang digambarkan dengan kolom vertikal yang disebut dengan *lifeline*. Sedangkan pesan atau perintah digambarkan sebagai garis panah dari satu *lifeline* ke *lifeline* yang lain.

Diagram sequence dapat digunakan untuk menggambarkan serangkaian langkah yang dilakukan sebagai respon dari sebuah peristiwa untuk menghasilkan suatu output tertentu. *Sequence diagram* berhubungan dan berkaitan erat dengan use case diagram, di mana satu use case diagram akan menjadi satu *diagram sequence*.

Tujuan utama pembuatan diagram sequence adalah untuk mengetahui urutan kejadian yang dapat menghasilkan output yang diinginkan. Selain itu, tujuan dari diagram sequence ini mirip dengan *activity diagram*, seperti menggambarkan alur kerja dari sebuah aktivitas, serta dapat menggambarkan aliran data dengan lebih detail, termasuk data atau perilaku yang diterima atau dikirimkan. Berikut merupakan contoh *Sequence Diagram* yang ditunjukkan pada Gambar 2.4 Contoh Sequence Diagram [27].



Gambar 2.4 Contoh Sequence Diagram