BAB I. PENDAHULUAN

1.1 Latar Belakang

Pada tahun 2020, Kementerian Pendidikan dan Kebudayaan, melalui Direktorat Jenderal Pendidikan Tinggi, memperkenalkan program Kampus Merdeka. Program Kampus Merdeka merupakan inisiatif yang diprakarsai oleh Nadiem Makarim, Menteri Pendidikan dan Kebudayaan Indonesia. Tujuannya adalah memberikan kesempatan belajar kepada seluruh mahasiswa perguruan tinggi untuk mendapatkan pengalaman belajar di dunia industri di luar kampus. (Sopiansyah & Masruroh, 2021). Ide pembelajaran ini merupakan perkembangan dari konsep Merdeka Belajar sebelumnya. Perancangan konsep Kampus Merdeka pada dasarnya adalah inovasi pembelajaran untuk memastikan kualitas pembelajaran yang unggul. Landasan hukum pelaksanaan kurikulum MBKM (Merdeka Belajar Kampus Merdeka) meliputi berbagai peraturan, seperti Permendikbud Nomor 3 Tahun 2020 tentang standar Pendidikan Tinggi; Permendikbud Nomor 4 tahun 2020 tentang Transformasi Perguruan Tinggi Negeri menjadi Perguruan Tinggi Berbadan Hukum; Permendikbud Nomor 5 Tahun 2020 tentang Akreditasi Program Studi dan Perguruan Tinggi; Permendikbud Nomor 6 Tahun 2020 Tentang Penerimaan Mahasiswa Baru Program Studi pada Perguruan Tinggi Negeri; Permendikbud Nomor 7 Tahun 2020 tentang Pendirian, Perubahan, Pembubaran Perguruan Tinggi Negeri, dan Pendirian, Perubahan, Pencabutan Izin Perguruan Tinggi Swasta. Tujuan dari kebijakan Merdeka Belajar Kampus Merdeka adalah mendorong mahasiswa untuk menguasai berbagai bidang ilmu pengetahuan sesuai dengan spesialisasinya, sehingga mereka siap untuk bersaing dalam lingkup global. (Baharuddin, 2021). Kebijakan ini memberikan kesempatan kepada mahasiswa untuk memilih mata kuliah yang akan mereka tempuh berdasarkan keinginan sendiri.

Beberapa program dan kegiatan yang ditawarkan oleh Kampus Merdeka seperti kegiatan Magang dan Studi Independen Bersertifikat (MSIB), Kampus Mengajar, Pertukaran Mahasiswa Merdeka (PMM), Wirausaha Merdeka, Indoesian *International Student Mobility Award* (IISMA), Praktisi Mengajar, Bangkit *by* Google, GoTo *and* Traveloka dan Kementerian ESDM-GERILYA. Setiap program kegiatan yang ditawarkan memiliki keunggulan masing-masing

dan memberikan kesempatan kepada seluruh mahasiswa untuk mendapatkan ilmu dan pengalaman langsung dari industri terkait. Dengan mengikuti Program Kampus Merdeka mahasiswa akan mendapatkan banyak manfaat dan keuntungan seperti kegiatan yang dapat dikonversi menjadi maksimal 20 SKS, memperluas jaringan hingga ke luar program studi dan universitas, eksplorasi pengetahuan dan kemampuan di lapangan selama lebih dari satu semester, dan menimba ilmu secara langsung dari mitra berkualitas dan terkemuka.

Bangkit *Academy* adalah inisiatif kesiapan karier yang dirancang oleh Google untuk memberikan pengalaman langsung kepada mahasiswa Indonesia dengan praktisi industri. Tujuan utama program ini adalah mempersiapkan mahasiswa dengan keterampilan relevan yang diperlukan untuk mencapai kesuksesan karier di perusahaan teknologi terkemuka. Program ini didirikan bersama oleh Google, GoTo, dan Traveloka, diperkenalkan pada tahun 2020, dan berlanjut hingga 2023. Bangkit menawarkan tiga lintas pembelajaran utama: *Machine Learning*, *Mobile Development*, dan *Cloud Computing*. Selain mengajarkan keterampilan teknologi, Bangkit juga menitikberatkan pada pengembangan kemampuan Bahasa Inggris dan *soft skills* yang sangat penting untuk mempersiapkan diri menghadapi dunia kerja. Program Bangkit memberikan banyak manfaat bagi pesertanya, termasuk peluang untuk mendapatkan Sertifikasi Global dari Google, kurikulum dan pengajaran yang diarahkan oleh praktisi industri, peningkatan keterampilan di bidang teknologi, pengembangan *soft skills* dan Bahasa Inggris, konversi hingga 20 SKS (Sistem Kredit Semester), kesempatan karier eksklusif bagi lulusan Bangkit, dan dukungan finansial hingga 140 juta rupiah untuk 20 tim *capstone project* terbaik yang ingin membangun *startup* mereka.

Alur pembelajaran *Mobile Development* di Bangkit *Academy* bertujuan untuk melatih peserta agar menjadi android *developer* yang mampu menciptakan beragam aplikasi *mobile* berbasis android dan berinovasi dengan teknologi terkini untuk menangani permasalahan industri. Setiap peserta akan mendapatkan akses selama 5 bulan di sistem manajemen pembelajaran (LMS) yang dimiliki oleh Dicoding *Academy* dengan *learning path* khusus untuk pengembangan aplikasi Android. Peserta diwajibkan menyelesaikan setiap kelas dan tugas yang diberikan,

termasuk kelas-kelas seperti Memulai Pemrograman Dengan Kotlin, Belajar Membuat Aplikasi Android Untuk Pemula, Belajar Fundamental Aplikasi Android, Belajar Pengembangan Aplikasi Android *Intermediate*, Belajar Membuat Aplikasi Android dengan Jetpack Compose, Belajar Prinsip Pemrograman SOLID, dan Belajar Dasar UX *Design*. Untuk menyelesaikan setiap kelas, peserta akan diuji dengan membuat tugas atau *submission* yang memenuhi ketentuan yang telah ditetapkan, serta mengikuti ujian akhir untuk setiap kelas. Setiap *submission* akan *direview* oleh *reviewer* dari tim Dicoding yang merupakan ahli dan memiliki kemahiran di bidang pengembangan aplikasi Android.

Diakhir pembelajaran para peserta Bangkit *Academy* akan diberikan sebuah tugas besar yang bernama *Capstone Project* yang merupakan tugas besar atau proyek akhir yang melibatkan seluruh unsur penjuruan atau *learning path* untuk membuat sebuah proyek yang memiliki unsur *Mobile Development, Machine Learning* dan *Cloud Computing*. Tugas besar ini dikerjakan secara per tim dan setiap tim akan beranggotakan 6 orang dengan rincian 2 orang dari *Mobile Development,* 2 orang dari *Machine Learning* dan 2 orang dari *Cloud Computing.*

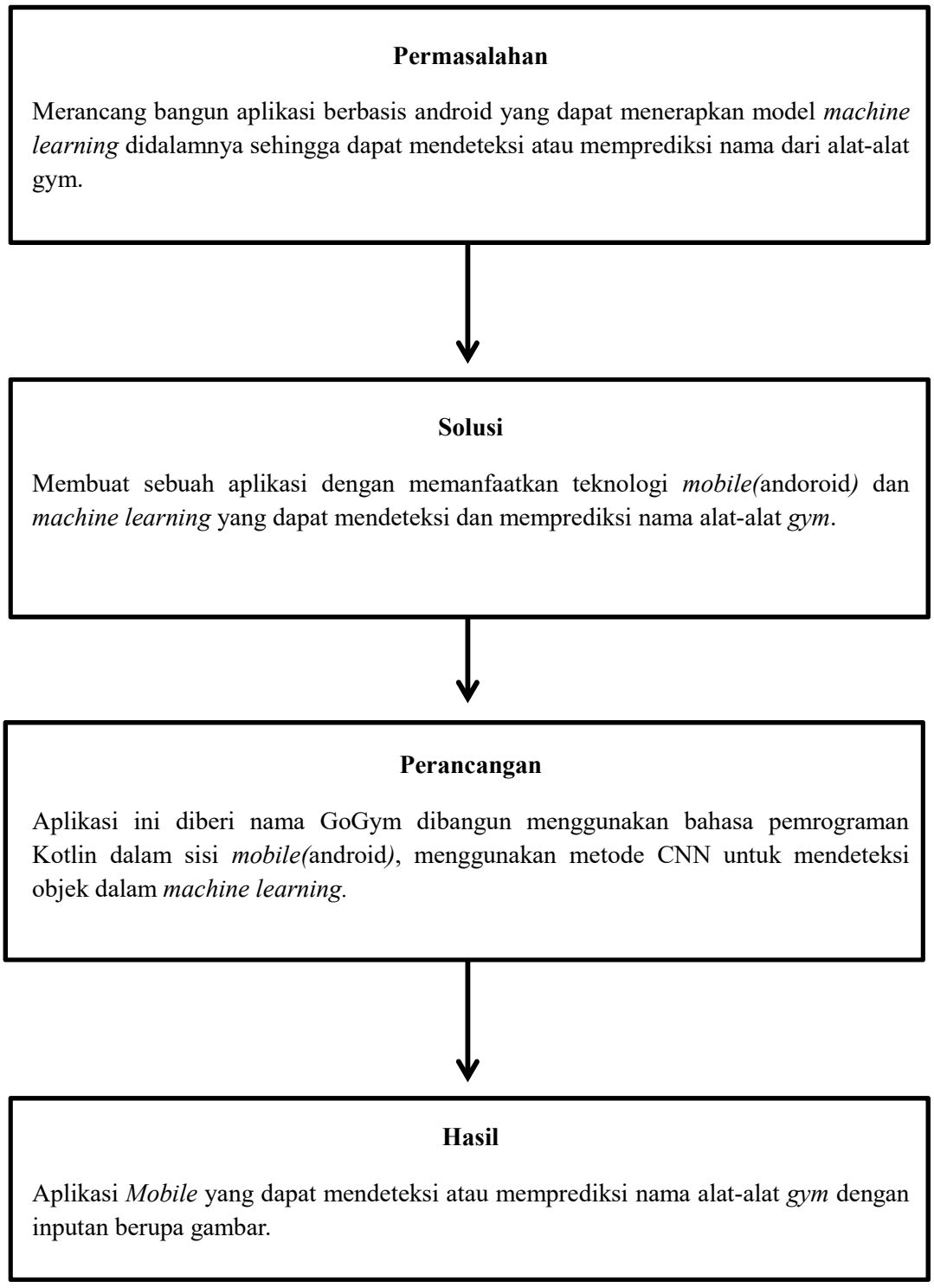
Dalam pembuatan proyek akhir ini penulis tergabung dalam sebuah tim bernama GoGym dan berperan sebagai android *developer*. Tim GoGym mendapatkan tantangan dari Bangkit untuk menciptakan sebuah inovasi dalam bentuk digital untuk dunia olahraga yaitu merancang sebuah aplikasi android berbasis *cloud computing* dan teknologi *machine learning* yang dapat mengenali atau memprediksi nama alat-alat *gym* berdasarkan gambar. Peran android *developer* pada pengembangan aplikasi ini adalah membangun *user interface* pada aplikasi *figma* lalu melakukan *slicing* desain tersebut kedalam aplikasi android menggunakan android studio dengan menggunakan bahasa pemrograman kotlin. Kemudian mengintegrasikan aplikasi *mobile(*android) tersebut dengan *Back-End* yang dibuat oleh anggota *cloud compting* serta mengintgrasikan *Machine Learning* kedalam aplikasi android dengan menggunakan file TfLite. Diharapkan aplikasi ini menjadi terobosan dan inovasi dalam pemanfaatan teknologi *machine learning* dan android dalam bidang olahraga serta dapat membantu masyarakat luas untuk mengetahui berbagai macam alat-alat gym dan cara penggunaannya.

1.2 Tujuan

Tujuan tugas akhir adalah merancang bangun aplikasi berbasis android yang menerapkan teknologi *machine learning* didalamnya, sehingga aplikasi ini dapat mendeteksi nama alat-alat *gym* dengan inputan data berupa gambar.

1.3 Kerangka Pemikiran

Setelah mendapat tugas dari Tim Bangkit untuk merancang sebuah aplikasi yang dapat menjadi terobosan dan inovasi pemanfaatan teknologi *machine learning* dan android dalam dunia olahraga, tim GoGym melakukan disukusi bersama semua anggota serta *advisor* (pembimbing) yang diberikan oleh Tim Bangkit. Akhirnya kami mendapatkan tugas yaitu membuat sebuah terobosan dan inovasi dengan memanfaatkan teknologi android dan *machine learning* untuk membuat sebuah aplikasi *mobile*(android*)* yang dapat mengenali atau memprediksi nama alat-alat gym berdasarkan inputan gambar. Untuk lebih jelasnya kerangka pemikiran disajikan dalam Gambar 1 berikut.



Gambar 1. Kerangka Pemikiran

1.4 Kontribusi

Kontribusi yang diberikan dengan adanya *prototype* aplikasi *mobile* pendeteksi nama alat-alat *gym* berbasis android dengan *machine learning*. Yang diberikan diantara:

1. Memberikan gambaran dan terobosan inovasi dengan hadirnya *protype* aplikasi GoGym ini diharapkan menjadi inovasi kedepannya dalam pengembangan aplikasi *mobile(*android*)* berbasis *machine learning* dalam pemanfaatan bidang olaharaga.
2. Menjawab tantangan dan tugas yang diberikan oleh Tim Bangkit dengan melahirkan ide dan inovasi dalam pemanfaatan teknologi android dan *machine learning.*
3. Menyumbangkan karya serta ilmu pengetahuan kepada Politeknik Negeri Lampung sehingga dapat memberikan manfaat dan referensi kepada adik tingkat dalam contoh perancangan aplikasi android yang menerapkan model *machine learning.*

BAB II. TINJAUAN PUSTAKA

2.1 Rancang Bangun

Rancang Bangun merupakan proses yang melibatkan interpretasi hasil dari suatu sistem ke dalam bahasa pemrograman. Tujuannya adalah untuk memberikan penjelasan terperinci mengenai implementasi komponen-komponen sistem. Perancangan sistem mencakup aktivitas menciptakan sistem baru, menggantikan, atau memperbaiki sistem yang sudah ada, baik secara keseluruhan maupun sebagian. Secara keseluruhan, perancangan dan pengembangan sistem terkait dengan perencanaan dan pembangunan sistem secara komprehensif (A. Surahman et al., 2022).

2.2 Aplikasi Mobile

Menurut Hasan Abdurahman dan Asep Ririh Riswaya (2014), aplikasi adalah program yang telah siap digunakan dan dapat dieksekusi oleh pengguna untuk mencapai hasil yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut. Aplikasi pada dasarnya adalah solusi untuk mengatasi masalah dengan menggunakan teknik pemrosesan data tertentu. Pemrosesan data aplikasi ini terkait dengan komputasi yang diinginkan atau diharapkan. Secara umum, aplikasi merupakan alat terapan yang memiliki fungsi dan kemampuan khusus, berjalan pada suatu sistem tertentu, dan membantu berbagai kegiatan yang dilakukan oleh manusia.(Simargolang & Nasution, 2018).

Aplikasi *mobile* adalah jenis aplikasi yang memungkinkan mobilitas menggunakan perangkat seperti telepon seluler atau *handphone*. Dengan aplikasi *mobile*, pengguna dapat dengan mudah melakukan berbagai kegiatan, termasuk hiburan, berjualan, belajar, pekerjaan kantor, *browsing*, dan lain-lain. Banyak penelitian telah menggunakan aplikasi *mobile* untuk berbagai tujuan, seperti hiburan dan memudahkan layanan komunikasi data.(S. Surahman & Setiawan, 2017).

2.3 Pendeteksi

Pendeteksi adalah suatu perangkat yang secara otomatis merekam perubahan tertentu, seperti suhu atau tekanan udara di suatu lokasi. Alat atau instrumen (*tools*) adalah objek yang membantu mempermudah tugas sehari-hari kita. Pendeteksi adalah perangkat yang digunakan untuk merekam atau

menghitung informasi mengenai objek yang sedang diselidiki secara otomatis, seperti perangkat yang mendeteksi korsleting listrik. Berdasarkan definisi tersebut, detektor atau pendeteksi adalah perangkat otomatis yang digunakan untuk menghitung atau merekam informasi tentang suatu objek.

2.4 Alat-alat *Gym*

Peralatan gym adalah instrumen yang diciptakan khusus untuk aktivitas latihan fisik dan kesehatan. *Range* peralatan ini mencakup beragam jenis yang digunakan untuk latihan kekuatan otot dan meningkatkan keseimbangan tubuh. Contoh dari peralatan *gym meliputi barbell, dumbbell, stepper aerobik*, dan masih banyak lagi.

2.5 *Gym*

Gym adalah salah satu olahraga popular untuk melatih kebugaran tubuh dengan menggunakan beban dan alat-alat *gym* untuk membentuk tubuh. Di tempat *gym* orang dapat melakukan melakukan berbagai jenis latihan fiisk, seperti angkat beban, *kardio*, berlari di *treadmill*, bersepeda statis dan masih banyak lainnya. Tujuan utama *gym* adalah untuk membantu orang meningkatkan kebugaran fisik, membangun kekuatan otot, meningkatkan daya tahan *kardiovaskular*, dan mencapai tujuan kebugaran pribadi mereka. *Gym* biasanya dilengkapi dengan berbagai peralatan kebugaran, seperti *dumbbell*, mesin angkat beban, alat *kardio*, dan lebih banyak lagi.

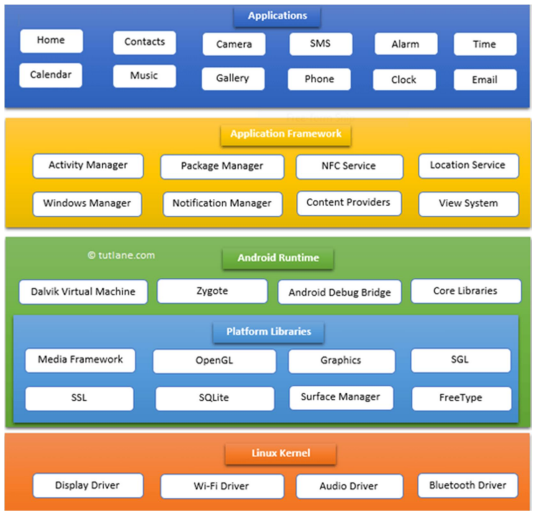
2.6 Android

Menurut Satyaputra & Aritonang (2016: 2), android adalah suatu sistem operasi yang digunakan pada *smartphone* dan *tablet*. Sistem operasi ini berperan sebagai penghubung antara perangkat (*device*) dan penggunanya, memungkinkan interaksi pengguna dengan perangkatnya serta menjalankan berbagai aplikasi yang ada pada perangkat tersebut. Nazrudin Safaat H dalam M. Ichwan, Fifin Hakiky (2011: 15) mendefinisikan android sebagai sistem operasi mobile yang berbasis linux, mencakup sistem operasi, *middleware*, dan aplikasi. Android, sebagai sistem operasi berbasis linux untuk telepon seluler, memberikan platform terbuka bagi para pengembang untuk mengembangkan aplikasi mereka sendiri. Selanjutnya, Murtiwiyati & Glenn Lauren (2013: 2) menjelaskan bahwa android adalah sistem operasi untuk perangkat *mobile* yang menggunakan basis linux,

meliputi sistem operasi, *middleware*, dan aplikasi. (Kuswanto & Radiansah, 2018).

Android merupakan sistem operasi yang populer, dikembangkan oleh Google, khusus untuk perangkat *mobile* seperti *smartphone* dan *tablet*. Namun, kini juga digunakan dalam perangkat lain seperti televisi pintar, konsol *game*, dan jam tangan pintar. Android memiliki berbagai karakteristik unggul yang membuatnya diminati di berbagai kalangan. Beberapa karakteristik tersebut antara lain, sifat *open source* yang memungkinkan akses kode sumber oleh siapa saja, ekosistem aplikasi yang kuat melalui Google *Play Store*, kemudahan penyesuaian tampilan (*Customizability*), kemampuan untuk menjalankan beberapa aplikasi secara bersamaan (*Multi-Tasking*), dan dukungan untuk beragam perangkat keras. Komunitas pengembang yang besar dan aktif juga memfasilitasi banyak sumber daya dan dukungan bagi pengembang yang ingin menciptakan aplikasi Android. Selain itu, Android terintegrasi dengan Layanan Google seperti Gmail, Google Maps, Google Drive, memudahkan sinkronisasi data dan akses ke layanan online.

Sistem operasi Android memiliki struktur yang terdiri dari beberapa tingkatan yang umumnya disebut sebagai arsitektur Android. Arsitektur Android adalah struktur komponen perangkat lunak yang dirancang untuk memenuhi kebutuhan perangkat *mobile*. Struktur ini terdiri dari lima tingkatan utama, yaitu aplikasi (*application*), kerangka kerja Android (Android *framework*), *runtime* Android (Android *runtime*), perpustakaan *platform* (*platform library*), dan kernel Linux. Berikut adalah lapisan arsitektur android yang disajikan dalam Gambar 2.



Gambar 2. Arsitektur Android

Berikut adalah penjelsan terkait masing-masing lapisa dari arsitektur sistem operasi android:

1. *Linux Kernel*

   Linux kernel adalah elemen sentral dalam sistem operasi Android yang bertanggung jawab menyediakan fungsionalitas sistem operasi ke perangkat *mobile*. Tugas utama kernel meliputi mengelola berbagai *driver* perangkat seperti *driver* kamera, *driver* tampilan, *driver* Bluetooth, *driver keypad*, manajemen memori, manajemen proses, manajemen daya, dan sebagainya. Selain itu, kernel menangani aspek Linux terkait jaringan, driver perangkat, dan memastikan interaksi antar-muka berjalan dengan lancar.

2. *Platform Libraries*

   Platform libraries meliputi beragam pustaka inti C/C++ dan pustaka berbasis Java seperti SSL, libc, Grafik, SQLite, Webkit, Media, Manajer Permukaan, OpenGL. Perpustakaan ini juga memiliki tanggung jawab dalam memutar dan merekam format audio dan video, *FreeType* untuk mendukung *font*, WebKit untuk mendukung peramban, SQLite untuk basis data, dan SSL untuk keamanan internet.

3. *Amdroid Runtime*

   Android Runtime, yang merupakan lapisan ketiga dari android *architecture*, adalah mesin yang menggerakkan aplikasi bersama dengan perpustakaan dan membentuk fondasi untuk kerangka aplikasi. Di dalam android *runtime*, terdapat komponen kunci yaitu Dalvik *Virtual Machine* (DVM), yang merupakan suatu jenis Mesin Virtual Java yang didesain dan dioptimalkan secara khusus untuk Android.

4. *Android Framework*

   Layer Android *Framework* menyediakan kelas-kelas dan antarmuka untuk mengembangkan aplikasi Android serta layanan-layanan ke aplikasi dalam bentuk kelas-kelas Java. Layanan-layanan seperti telepon, lokasi, manajemen notifikasi, NFC, dan manajemen tampilan adalah contoh dari layanan-layanan yang terdapat di dalam Android.

5. *Application*

   Layer teratas pada Android *Architecture* adalah *Application*. Pengguna dapat mengembangkan aplikasi mereka untuk diinstal pada lapisan ini saja. Layer *Application* berinteraksi dengan Android *runtime* yang menggunakan kelas-kelas dan layanan yang tersedia dari *framework* aplikasi.

2.7 Machine Learning

Machine learning adalah bidang ilmu komputer yang mempelajari cara suatu mesin menyelesaikan masalah tanpa memerlukan deprogram secara eksplisit. Mitchell mendefinisikan *machine learning* sebagai kemampuan komputer untuk belajar dari pengalaman terhadap tugas yang diberikan dengan kinerja yang dapat diukur (Mitchell, 1997). Proses *machine learning* melibatkan pengembangan struktur di mana setiap tahap membangun versi mesin yang lebih baik. (Ula ,Ulva, Mauliza, 2021).

Machine Learning (ML), juga dikenal sebagai pembelajaran mesin, merupakan pendekatan dalam AI yang sering digunakan untuk meniru atau menggantikan perilaku manusia dalam menyelesaikan masalah atau melakukan otomatisasi. Sesuai dengan namanya, ML mencoba untuk mereplikasi bagaimana manusia atau makhluk cerdas memproses pembelajaran dan membuat generalisasi. Terdapat dua aplikasi utama dalam ML, yaitu klasifikasi dan prediksi. ML memiliki ciri khas, yaitu adanya proses pelatihan atau pembelajaran yang memerlukan data untuk dipelajari, yang disebut sebagai data *training*. Klasifikasi adalah metode di ML yang memungkinkan mesin untuk mengelompokkan atau mengkategorikan objek berdasarkan ciri-ciri tertentu, mirip dengan cara manusia membedakan antara berbagai objek. Sementara itu, prediksi atau regresi memungkinkan mesin untuk memprediksi keluaran dari data masukan berdasarkan pembelajaran sebelumnya pada tahapan training. Beberapa metode ML yang populer meliputi Sistem Pengambil Keputusan, *Support Vector Machine* (SVM), dan Jaringan Saraf Tiruan (*Neural Network*).

Dalam bidang *machine learning*, kita akan mempelajari konsep yang disebut *Deep Learning. Deep Learning* adalah salah satu cabang dari *Machine Learning* yang menggunakan struktur jaringan neural yang kompleks dengan

banyak lapisan (disebut *deep neural networks*) untuk menginterpretasi dan menganalisis data. Kehadiran *Deep Learning* dikenal karena kemampuannya untuk memproses data yang kompleks seperti gambar, suara, dan teks. *Neural network*, di sisi lain, adalah sebuah metode dalam bidang kecerdasan buatan yang mengajarkan komputer untuk memproses data dengan cara yang terinspirasi dari kerja otak manusia.

Deep Learning adalah suatu pendekatan pembelajaran yang menggunakan jaringan *neural* buatan dengan banyak lapisan (*multi-layer*). Jaringan *neural* buatan ini dirancang menyerupai struktur otak manusia, di mana *neuron-neuron* saling terhubung membentuk jaringan *neuron* yang kompleks. *Deep Learning* juga dikenal dengan sebutan *deep structured learning, hierarchical learning*, atau *deep neural learning,* karena memanfaatkan transformasi *non-linier* yang lebih dari satu tingkat. Metode *Deep Learning* dapat dianggap sebagai penggabungan antara *machine learning* dan kecerdasan buatan (*artificial neural network*). Beberapa algoritma yang termasuk kedalam *Deep Learning* antara lain:

1. *Convolutional Networks (CNN)*

   Convolutional Neural Networks (CNNs) merupakan salah satu varian dari *Deep Learning* yang sangat efektif dalam mengolah data grid seperti gambar dan video. CNNs memiliki lapisan konvolusi yang mampu memperoleh informasi tentang pola lokal pada area kecil dari input, yang memfasilitasi pengenalan pola kompleks tingkat tinggi.

2. *Recurrent Neural Networks (RNNs)*

   Recurrent Neural Networks (RNNs) adalah jenis arsitektur jaringan neural yang memiliki *loop*, memungkinkan aliran informasi dari satu langkah waktu ke langkah waktu berikutnya. RNNs terbukti sangat efisien dalam mengolah data yang memiliki urutan, seperti teks, suara, dan video.

3. *Long Short-Term Memory (LSTM)*

   Long Short-Term Memory (LSTM) merupakan adaptasi khusus dari *Recurrent Neural Networks* (RNNs) yang dirancang untuk mengatasi kendala "menghilang atau meledaknya *gradien*" yang umumnya muncul pada RNN saat memproses data dengan panjang yang bervariasi.

4. *Generative Adversarial Networks (GANs)*

   Generative Adversarial Networks (GANs) merupakan struktur yang terdiri dari dua jaringan *neural*, yaitu generator dan *discriminator*, yang saling bersaing. Inti tujuannya adalah untuk menciptakan data baru yang memiliki distribusi yang serupa dengan data latih.

5. *Deep Belief Networks (DBNs)*

   Deep Belief Networks (DBNs) adalah struktur jaringan yang terdiri dari sejumlah besar lapisan dengan fungsi pembelajaran yang bervariasi. DBNs utamanya digunakan dalam tugas pembelajaran yang tidak terawasi.

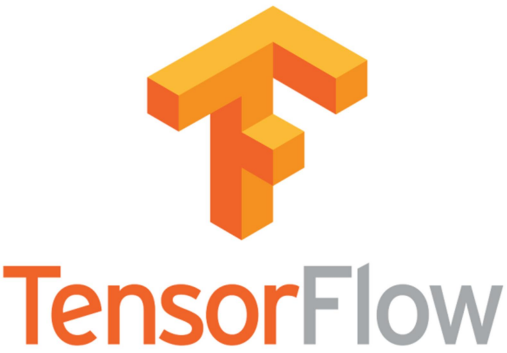
6. *Autoencoders*

   Autoencoders adalah tipe jaringan yang mempelajari representasi data input secara efisien melalui pendekatan kompresi dan dekompresi. Penggunaan umum meliputi reduksi dimensi dan rekonstruksi data.

7. *Deep Q Networks (DQNs)*

   Deep Q-Networks (DQNs) adalah tipe arsitektur jaringan *neural* yang diterapkan pada pembelajaran penguatan, terutama di bidang permainan video. DQNs memiliki kemampuan untuk mempelajari strategi yang kompleks dan mencapai kinerja tinggi dalam permainan.

Dalam pengembangan model *machine learning*, kita akan memanfaatkan suatu perangkat lunak yang dikenal dengan sebutan TensorFlow. TensorFlow merupakan suatu kerangka kerja komputasi yang digunakan untuk membangun model pembelajaran mesin. Kerangka kerja ini menyediakan berbagai *toolkit* yang memungkinkan pembuatan model pada tingkat abstraksi yang diinginkan dan dapat menjalankan grafik pada berbagai *platform hardware*, termasuk CPU, GPU, dan TPU (Hikmatia A.E & Ihsan Zul, 2021). Salah satu penjelasan lainnya adalah bahwa TensorFlow merupakan perangkat lunak dengan kode terbuka yang dikembangkan oleh tim Google Brain. Kerangka kerja ini sangat terkenal dalam *domain Machine Learning* dan *Deep Learning*. TensorFlow memungkinkan para pengembang untuk efisien membangun dan melatih berbagai model *Machine Learning* dan *Deep Learning*. Representasi model ini terwakili dalam bentuk grafik komputasi yang terdiri dari serangkaian operasi pada tensor, yaitu *array multidimensi*. Berikut logo TensorFlow disajikan dalam Gambar 3.



Gambar 3. Logo TensorFlow

Dengan menggunakan TensorFlow model *machine learning* yang telah di latih dapat dikonversi menjadi format ".tflite", dengan format file ini model machine learning dapat dijalankan atau di gunakan di perangkat seluler dengan sumber daya terbatas. Selain untuk perangkat seluler bisa juga digunakan pada perangkat yang berbasis *Internet of Things*(IoT)

2.8 CNN (Convolutional Neural Networks)

CNN merupakan jenis arsitektur jaringan saraf tiruan yang sangat efektif dalam memproses data grid, terutama citra. Arsitektur ini memiliki lapisan konvolusi yang memungkinkan deteksi pola dan fitur dalam gambar dengan tingkat efisiensi tinggi. Oleh karena itu, CNN sering digunakan untuk tugas-tugas pengenalan pola dalam gambar, khususnya klasifikasi objek. Dalam penjelasan lainnya, CNN merupakan perkembangan dari *Multilayer Perceptron* (MLP) yang termasuk dalam jenis *neural network feed forward* (non-berulang). *Convolutional Neural Network* dirancang khusus untuk mengolah data dua dimensi. Klasifikasi CNN sebagai *Deep Neural Network* didasari oleh kedalaman jaringannya yang tinggi, dan aplikasinya yang luas pada data citra. CNN berfokus pada analisis gambar visual, mendeteksi, dan mengenali objek pada citra, yang diwakili sebagai vektor berdimensi tinggi yang memerlukan banyak parameter untuk mendeskripsikan jaringannya. Secara umum, CNN tidak jauh berbeda dengan *neural network* biasa, terdiri dari *neuron* dengan *weight*, bias, dan *activation function*.

CNN memiliki peran krusial dalam algoritma *deep learning* dan sering dimanfaatkan untuk mengidentifikasi atau mengenali informasi yang bersifat visual, seperti pengenalan objek dalam gambar. Pada intinya, algoritma CNN

merupakan suatu arsitektur jaringan saraf tiruan yang sangat efisien untuk mengklasifikasikan citra. Konsep sentral dari CNN terfokus pada operasi konvolusi, di mana setiap fitur dari citra diekstraksi sehingga membentuk pola-pola yang mempermudah proses klasifikasi. Pendekatan ini secara signifikan meningkatkan efisiensi pembelajaran citra untuk implementasi yang lebih baik (Arsal et al., 2020) .

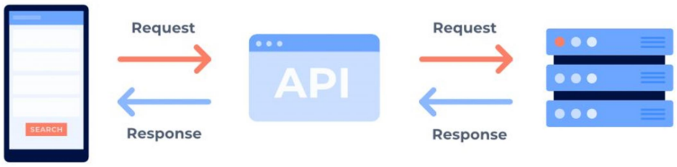
2.9 Aplikasi Berbasis Android

Berdasarkan Asropudin (2013:6), aplikasi merupakan perangkat lunak yang diciptakan oleh perusahaan teknologi untuk melaksanakan tugas-tugas spesifik, contohnya seperti Ms.Word dan Ms.Excel. Android adalah suatu sistem operasi yang dirancang khusus untuk perangkat *mobile* yang berbasis linux. Awalnya, sistem operasi ini dikembangkan oleh Android Inc. dan kemudian diakuisisi oleh Google pada tahun 2005. (Maiyana, 2018). Jadi, simpulannya, aplikasi Android adalah perangkat lunak yang dirancang untuk beroperasi di perangkat mobile atau smartphone, mampu mendukung beragam program dan memenuhi kebutuhan pengguna.

2.10 Aplication Programming Interface (API)

API adalah antarmuka yang diperlukan untuk mengakses layanan atau program dari sebuah aplikasi. Fungsi-fungsi yang telah tersedia dari aplikasi lain disediakan melalui API, memungkinkan pengembang untuk memanfaatkan fungsionalitas tersebut tanpa perlu membangunnya dari awal. Dalam konteks situs web, API memungkinkan pemanggilan fungsi melalui HTTP dan menerima *respons* dalam bentuk *Extensible Markup Language* (XML) atau *JavaScript Object Notation* (JSON). (Hasanuddin, 2022). Penggunaan fasilitas API bertujuan untuk mempercepat dan menyederhanakan proses pengembangan aplikasi dengan menyediakan fungsi-fungsi terpisah yang tidak perlu dirancang ulang, menghindari duplikasi fitur yang serupa.

API menyediakan fungsionalitas yang memudahkan pengembangan aplikasi dan situs web. Hal ini juga memungkinkan perluasan dan penyederhanaan pembangunan aplikasi dengan memanfaatkan API untuk menyimpan data di server, yang pada gilirannya mempercepat transmisi data. Berikut alur kerja dari API disajikan dalam Gambar 4.



Gambar 4. Alur Kerja *Aplication Programming Interface* (API)

Dalam melakukan tugasnya, API melewati serangkaian langkah dari awal hingga menghasilkan output. Berikut adalah mekanisme kerja API:

1. Aplikasi mengakses API. Awalnya, API akan memulai proses saat pengguna membuka aplikasi. Contohnya, pengguna membuka aplikasi pemesanan tiket *online* dan ingin mengakses tujuan spesifik. Aplikasi akan berinteraksi dengan API yang telah terhubung.
2. API membuat permintaan ke server. Setelah aplikasi berhasil mengakses alamat API, permintan akan diteruskan ke *server*. Jadi, API akan menyampaikan bahwa aplikasi membutuhkan data untuk tanggal dan tujuan yang diminta.
3. Server merespons API. Setelah data ditemukan sesuai permintaan, server akan kembali ke API, lalu memberikan data berupa ketersediaan tempat duduk, waktu keberangkatan, dan lainnya.
4. API memberikan hasil ke Aplikasi. Terakhir, informasi akan diberikan ke aplikasi yang diakses pengguna. Proses ini terjadi bersama permintaan ke maskapai yang lain; jadi, terkadang aplikasi bisa menampilkan jadwal dari berbagai maskapai sekaligus dalam satu kali permintaan.

2.11 Google Map API

Google menyediakan pelayanan populer dan gratis, yaitu Google Maps, yang menawarkan fitur lokasi seperti koordinat, rute, tampilan jalan, dan lain-lain. Google Maps adalah sebuah peta yang memungkinkan tampilan dan navigasi dunia, memungkinkan akses dan penelusuran Negara, Provinsi, Kota, Daerah, bahkan Desa(Ariyanti, 2015). Dengan kata lain, Google Maps adalah sebuah peta global yang bisa diakses dan dilihat melalui web browser. Google Maps API adalah sebuah perpustakaan (library) yang menggunakan bahasa pemrograman JavaScript. Dalam mengembangkan aplikasi atau situs web, kita dapat

menggunakan fasilitas ini untuk mengintegrasikan fitur dari Google Maps ke dalam aplikasi atau situs web kita menggunakan Google Maps API.

2.12 Representational State Transfer (REST)

REST, singkatan dari *Representational State Transfer*, adalah sebuah arsitektur yang digunakan untuk mentransmisikan data melalui *interface* yang menggunakan protokol HTTP. Cara kerja REST mirip dengan aplikasi website pada umumnya. Pada REST, client dapat mengirimkan *request* kepada *server* menggunakan protokol HTTP. Setelah menerima *request*, *server* akan memproses data tersebut dan memberikan *response* kepada *client*(Hasanuddin, 2022).

Fungsi REST pada API adalah mendukung penggunaan metode HTTP (seperti *GET*, *POST, PUT,* dan *DELETE*) untuk melakukan operasi CRUD (Buat, Baca, Perbarui, Hapus) pada sumber daya. Ini memungkinkan pengembang untuk membangun aplikasi yang mudah digunakan dan mudah diintegrasikan dengan sistem lain. Berikut adalah beberapa contoh jenis method HTTP:

1. *GET*

   Digunakan untuk mendapatkan atau membaca informasi dari *server*. Contohnya adalah mendapatkan daftar pengguna atau mendapatkan detail suatu entitas.

2. *POST*

   Digunakan untuk membuat entitas baru di *server*. Contohnya adalah membuat pengguna baru atau mengirim pesan baru.

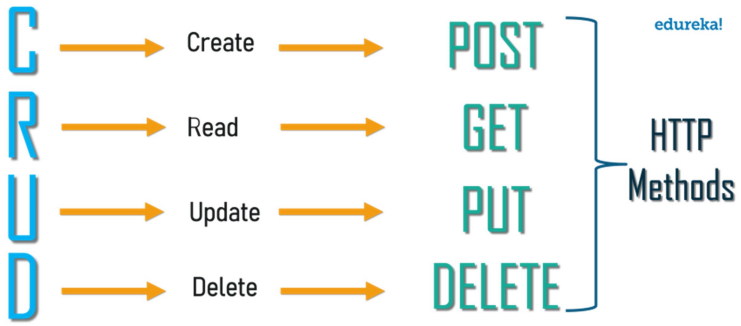
3. *PUT*

   Digunakan untuk memperbarui informasi atau entitas yang ada di *server*.

   Contoh penggunaannya adalah memperbarui informasi pengguna atau memperbarui detail.

4. *DELETE*

   Digunakan untuk menghapus entitas dari *server*. Contoh penggunaannya menghapus pengguna atau menghapus entri dari suatu *database*. Berikut adalah *method* yang ada pada *protocol* HTTP disajikan dalam Gambar 5.



Gambar 5. HTTP *Method*

2.13 JavaScript Object Notation (JSON)

JSON adalah sebuah format untuk berbagi data. Seperti namanya, bahasa pemrograman Javascript adalah yang melingkari bahasa JSON, akan tetapi format tersedia bahasa lain yang dapat JSON gunakan, seperti PHP, Java, python, dan Ruby. JSON dapat digunakan dengan cara disisipkan didalam file dokumen HTML. Namun, harus memerlukan beberapa karakter seperti tanda petik untuk tampilan JSON string, atau bisa juga dimasukan didalam sebuah *Variable*. Dengan format seperti itu maka sangat mudah untuk ditransferkan antar server web dengan client atau melalui browser (Hasanuddin, 2022).

2.14 Bahasa Pemrograman

Bahasa pemrograman, sering juga disebut bahasa komputer, adalah kumpulan aturan sintaks dan semantik yang digunakan untuk memberikan instruksi standar kepada komputer dalam pembuatan program komputer. (Premana et al., 2022). Bahasa pemrograman dipakai untuk menciptakan solusi yang sesuai dengan berbagai kebutuhan, termasuk di bidang pendidikan, ekonomi, bisnis, dan aspek sosial budaya.

2.15 Kotlin

Kotlin adalah bahasa pemrograman yang berasal dari *Java Virtual Machine* (JVM). Bahasa pemrograman ini dirancang khusus untuk Android, menggabungkan paradigma *object-oriented* (OO) dan fungsional. Selain itu, Kotlin menonjolkan *interoperabilitas*, memungkinkan integrasi dengan proyek-proyek yang menggunakan bahasa pemrograman Java. Selain digunakan untuk pengembangan aplikasi Android, Kotlin dapat diaplikasikan dalam pengembangan aplikasi desktop, web, dan backend. Awalnya dikembangkan oleh JetBrains,

pencipta IntelliJ IDEA, Kotlin kemudian menjadi *open source* dan mendapat dukungan kuat dari Google untuk pengembangan aplikasi Android. (Febriandirza, 2020). Berikut adalah logo dari bahasa kotlin disajikan dalam Gambar 6.

Gambar 6. Logo Kotlin

Kotlin diakui sebagai bahasa pemrograman utama dalam pengembangan aplikasi Android. Pada Google I/O pada bulan Mei 2017, Google secara resmi mendukung penggunaan Kotlin dalam pembuatan aplikasi Android, dan hal ini membawa dampak besar. Hampir 60% pengembang Android di seluruh dunia beralih dari bahasa Java ke Kotlin. Google terus meningkatkan Kotlin, menghadirkan Jetpack Compose, sebuah UI Toolkit yang mempermudah pengembang Android dalam membangun antarmuka pengguna aplikasi. Selain itu, Kotlin sekarang juga hadir dalam versi Kotlin Multiplatform, memungkinkan pengembang untuk membuat aplikasi berbasis mobile (Android) dan situs web dengan satu bahasa pemrograman, yaitu Kotlin. Berikut adalah kebelihan dari bahasa Kotlin:

1. *Open Suorce*
2. Mudah Dipelajari
3. Perusahaan Banyak yang Memakai Kotlin
4. Bahasanya Simpel dan Ringkas
5. Kotlin Lebih Ringan daripada Java
6. Lebih Aman dari Kesalahan *Error*
7. Terintegrasi dengan Android Studio
8. *Interoperable*
9. Kotlin Tidak Hanya untuk Android
10. Selalu diperbaharui oleh google

2.16 Android Studio

Android Studio adalah *Integrated Development Environment* (IDE) yang digunakan untuk mengembangkan aplikasi Android, dan ini merupakan proyek yang dikembangkan oleh Google. Android Studio merupakan perkembangan dari

IDE Eclipse dan didasarkan pada IntelliJ IDEA, sebuah IDE Java yang terkenal. Android Studio dijadwalkan untuk menggantikan Eclipse sebagai IDE resmi untuk pengembangan aplikasi Android. Sebagai pengembangan dari Eclipse, Android Studio memiliki sejumlah fitur baru yang membedakannya dari IDE Eclipse. Selain itu, berbeda dengan Eclipse yang menggunakan ADT, Android Studio menggunakan Gradle sebagai lingkungan pembangunan (build environment). (Makiolor et al., 2017). Google merekomendasikan Android Studio sebagai *Integrated Development Environment* (IDE) untuk mengembangkan aplikasi Android. Android Studio memiliki berbagai keunggulan dibandingkan dengan pendahulunya, yaitu Eclipse. Beberapa keunggulannya meliputi integrasi yang kuat, editor kode yang canggih, editor tata letak visual, emulator yang cepat, dukungan penuh untuk bahasa pemrograman Kotlin, *debugger* yang handal, dukungan untuk *Gradle*, dokumen yang kaya, dan komunitas pengembang yang besar. Berikut adalah beberapa kelibhan Android Studio dalam pengembangan aplikasi berbasis android dibandingkan IDE lainnya sebagai berikut:

1. Spesifik untuk Pengembangan Android

   Android Studio adalah *Integrated Development Environment* (IDE) yang diakui sebagai platform resmi untuk merancang aplikasi Android. Spesifikasinya dirancang dan dioptimalkan khusus untuk memfasilitasi pembuatan aplikasi Android, memberikan beragam fitur penting dan relevan yang mendukung pengembangan Android.

2. Integrasi Penuh dengan Android SDK

   Android Studio memiliki integrasi sempurna dengan Android SDK, mempermudah pengaksesan ke alat dan perpustakaan yang diperlukan dalam pengembangan aplikasi Android.

3. *Rich Code Editor*

   Android Studio menyediakan editor kode yang lengkap dengan penyorotan sintaksis, fungsi *auto-complete*, dan kemampuan refactor yang andal. Hal ini meningkatkan efisiensi dan kecepatan saat menulis kode.

4. *Preview Layout*

   Dalam IDE ini, terdapat fitur pratinjau yang memungkinkan pengembang untuk melihat tampilan visual dari tata letak (layout) aplikasi yang sedang mereka kembangkan.

5. *Emulator dan Debugging*

   Android Studio menyertakan Android Emulator yang dapat mensimulasikan beragam perangkat Android. Tambahan, terdapat alat debug yang sangat efektif untuk membantu pengembang dalam mendeteksi dan memperbaiki bug dalam aplikasi mereka.

6. Pustaka Bawaan dan Dukungan *Gradle*

   Dalam Android Studio, pengelolaan dependensi menggunakan *Gradle* didukung, sehingga mempermudah penggunaan pustaka dan alat tambahan saat mengembangkan aplikasi yang lebih kompleks.

7. Desain dan *Layout Tools*

   Dalam IDE ini, tersedia beragam alat dan editor yang memungkinkan desain tata letak antarmuka pengguna secara cepat dan efektif.

8. Dukungan Bahasa dan *Localization*

   Android Studio memfasilitasi pengembangan aplikasi dalam beragam bahasa dan mempermudah proses lokalitas, sehingga aplikasi dapat diadaptasi untuk berbagai pasar global.

9. Dokumentasi dan Bantuan

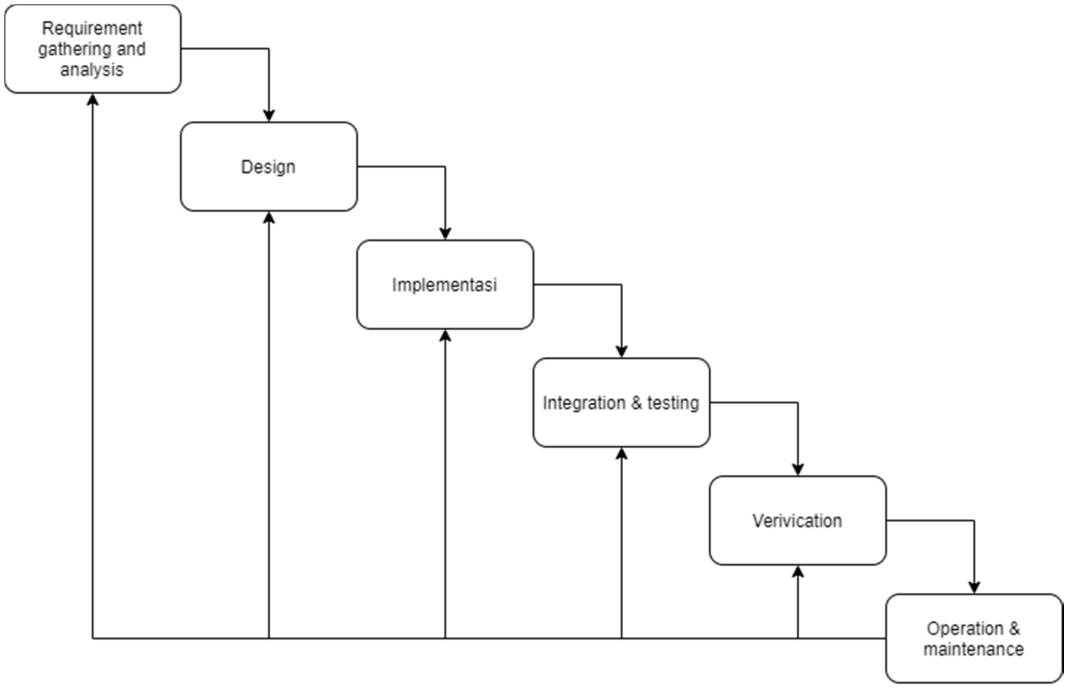
   Android Studio memberikan kemudahan akses ke dokumentasi resmi Android dan berbagai sumber daya pengembangan. Ini memungkinkan pengembang untuk dengan cepat mencari solusi dan informasi yang diperlukan.

10. Komunitas dan Ekosistem

    Sebagai IDE resmi untuk pengembangan Android, Android Studio memiliki dukungan dari komunitas yang besar, tutorial, dan banyak plugin yang dapat meningkatkan fungsionalitas dan efisiensi pengembangan.

2.17 Metode *Waterfall*

Metode *Waterfall* adalah siklus hidup pengembangan perangkat lunak yang terdiri dari serangkaian tahapan penting yang krusial dalam proses pembuatan perangkat lunak, dilihat dari sudut pandang pengembangan. (Febriani et al., 2020). Metode ini memvisualisasikan proses pengembangan perangkat lunak dengan serangkaian langkah yang linear dan jelas, seperti aliran air yang mengalir ke bawah (waterfall). Perhatian utama dari model *waterfall* adalah pada perencanaan dan penjadwalan dengan tenggat waktu dan anggaran yang telah ditetapkan. (Firdaus et al., 2022). Berikut adalah tahapan dalam metode *waterfall* disajikan dalam Gambar 7.



Gambar 7. Metode *Waterfall*

Terdapat lima tahapan dalam melakukan metode *waterfall*, dimulai dari analisis kebutuhan pengguna, desain hingga implementasi. Berikut tahapan pengembangan aplikasi dari setiap tahapan metode *waterfall*:

1. *Requirement Analysis*

   Pada tahap ini ,dilakukan analisis melaui pertemuan anara pengguna dan analis sistem untuk menentukan kebutuhan organisasi atau perusahaan serta tujuan yang ingin dicapai dengan aplikasi.

2. *Sistem and Software Design*

   Perancangan desisn sistem dilakukan untuk memberikan gambaran kepada pengguna mengenai sistem yang akan dibangun serta menyipakan kebutuhan perangkat keras untuk perancangan dari keseluruhan arsitektur sistem perangkat lunka yang akan dibuat.

3. *Implementation and Unit Testing*

   Pada tahap ini yaitu melakukan pemrograman dari desain sistem yang sudah dibuat sebelumnya yang dimana perangkat lunak di bagi menjadi modul-modul kecil yang akan digabungkan nantinya dalam tahapan berikutnya.

4. *Integration and Sistem Testing*

   Setalah semua modul yang dikembangkan dan di uji maka akan dilakukan pemeriksaan secara umum.

5. *Operation and Maintenance*

   Pada tahapan terakhir perangkat lunak dilakukan pemeliharaan yang di mana sudah di operasikan oleh pengguna,

2.18 UML (Unified Modelling Language)

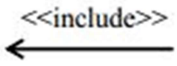
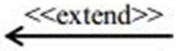
UML merupakan standar bahasa yang sering digunakan di industri untuk menguraikan kebutuhan, melakukan analisis dan desain, serta mengilustrasikan arsitektur dalam pengembangan perangkat lunak berorientasi objek.(Putra & Andriani, 2019). Selain digunakan untuk merancang dan menganalisis arsitektur, UML juga berguna untuk mendokumentasikan dan memodelkan sistem perangkat lunak. UML menyediakan notasi grafis yang membantu pengembang dan pemangku kepentingan dalam memahami, merencanakan, merancang, dan mendokumentasikan sistem yang akan dibangun. UML memberikan pendekatan konsisten dan standar untuk menggambarkan berbagai aspek dari sistem perangkat lunak. Ada beberapa jenis UML yang termasuk di dalamnya adalah sebagai berikut:

1. *Use Case Diagram*

   Use case diagram adalah representasi visual dari perilaku sistem informasi yang akan dikembangkan. *Use case* berfungsi dengan menerangkan interaksi umum antara pengguna sistem dengan sistem itu

sendiri melalui narasi yang menggambarkan bagaimana sistem tersebut digunakan. Berikut adalah simbol-simbol yang ada pada *use case diagram* disajikan pada Tabel 1 berikut.

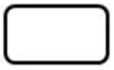
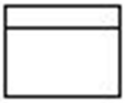
Tabel 1. Simbol-Simbol *Use Case Diagram*

Nama	Simbol	Keterangan
Aktor		Untuk mewakili peran orang atau alat ketika berkomunikasi dengan sistem
Use Case		Abstraksi dan interaksi antara system dan aktor
Association		Abstraksi dari penghubung antara actor dengan use case
Generalisasi		Menjukkan spesialisasi actor untuk dapat berpartisipasi dengan use case
Include		Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
Extend		Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika kondisi terpenuhi

2. *Activity Diagram*

Activity diagram adalah representasi grafis yang mengilustrasikan alur kerja atau aktivitas sistem yang terdapat dalam perangkat lunak. Tujuannya adalah untuk menjelaskan urutan aktivitas dalam suatu proses serta untuk mengetahui aktivitas *actor/*pengguna berdasarkan *use case* diagram yang telah dibuat sebelumnya. Berikut simbol-simbol yang ada pada *activity diagram* berserta nama dan fungsinya disajikan dalam Tabel 2 berikut.

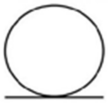
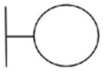
Tabel 2. Simbol-Simbol *Activity Diagram*

Nama	Simbol	Keterangan
Status Awal		Sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas		Aktivitas yang dilakukan oleh sistem
Percabangan		Percabangan dimana ada pilihan beberapa aktivitas
Pengabungan		Penggabungan dari beberapa aktivitas menjadi satu kesatuan
Status Akhir		Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane		*Swimlane* memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

3. *Sequence Diagram*

Sequence diagram menggambarkan perilaku objek pada skenario *use case* dengan menunjukkan periode hidup objek dan pesan yang ditransmisikan serta diterima antar objek.. Berikut adalah simbol-simbol yang ada pada *seuqnce diagram* disajikan pada Tabel 3 berikut.

Tabel 3. Simbol-Simbol *Sequence Diagram*

Nama	Simbol	Keterangan
Entity Class		Untuk mewakili peran orang atau alat ketika berkomunikasi dengan sistem
Boundary Class		Abstraksi dan interaksi antara system dan aktor

Tabel 3. Lanjutan

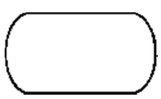
Nama	Simbol	Keterangan
Control Class		Abstraksi dari penghubung antara actor dengan use case
Recursive		Menjukkan spesialisasi actor untuk dapat berpartisipasi dengan use case
Activation		Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
Life Line		Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika kondisi terpenuhi

2.19 Flowchart

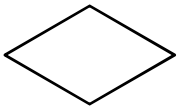
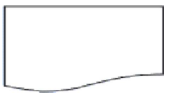
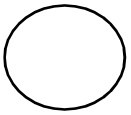
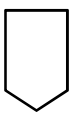
Flowchart sering dikenal sebagai diagram alur yang memanfaatkan simbol-simbol khusus untuk menguraikan suatu proses secara terperinci dan jelas. Simbol-simbol tersebut digunakan untuk menggambarkan hubungan antara satu proses dengan proses lainnya (Santoso & Luthfi, 2012).

Bagan alir (*flowchart*) adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi. Menurut (Indrajani, 2015) . Flowchart adalah representasi visual secara grafis dari langkah-langkah dan urutan prosedur suatu program.. Berikut simbol-simbol dari *flowchart* disajikan dalam Tabel 4.

Tabel 4. Simbol-Simbol *Flowchart*

Nama	Simbol	Keterangan
Terminal		Terminal biasanya di gunakan untuk memulai atau mengakhiri diagram

Tabel 4. Lanjutan

Nama	Simbol	Keterangan
Input/Output		*Input/Output* di mana digunakan menyatakan masukan dan keluaran yang terlepas dari jenisnya.
Process		Digunakan untuk menggambarkan proses atau alur pemrosesan yang dilakukan oleh komputer.
Decision		Berfungsi untuk melakukan pemilihan atau pengambilan keputusan terhadap proses yangakan dijalankan berdasarkan kondisi tertentu.
Flow Direction Symbol/Connecti ngLine		Digunakan untuk menghubungkan satu simboldengan simbol lainnya, yangmenunjukkan aliran atau arus suatu proses.
Document		Digunakan untuk mengacupada input dan output dari dokumen.
Connector		Digunakan untuk mendeklarasikan koneksi antara satu proses ke proses lainnya pada halaman yang sama.
Offline Connector		Digunakan untuk menampilkan koneksi dari satu proses ke proses lainnya di halaman yang berbeda.
Manual Operation		Digunakan untuk merujuk pada pemrosesan yang tidak dilakukan oleh *komputer*.

2.20 Artikel Ilmiah Terkait

Berikut adalah beberapa artikrl ilmiah yang diterapkan sebagai acuan dalam penyusunan tugas akhir dan dalam pembangunan aplikasi ini. Berikut artikel ilmiah terkait disajikan dalam Tabel 5 berikut.

Tabel 5. Artikel Ilmiah Terkait

No	Nama Penulis	Judul Artikel Ilmiah Terkait	Hasil Artikel Ilmiah Terkait
1	(Mahmud, Kunang, 2020)	Pengembangan Aplikasi Pengenalan Aksara Komering Menggunakan Metode Deep Learning Berbasis Android	Aplikasi ini bertujuan untuk mengelai berbagai macam aksara komering deng
2	(Ambarak, Falani, 2023)	Pengembangan aplikasi bahasa isyarat indonesia berbasis realtime video menggunakan model machine learning	Menghasilkan aplikasi android yang dapat mengenali bahsa isyarat secara realtime
3	(Asrianto, Bakhri, 2021)	Pengembangan Aplikasi Pengenalan Tokoh Pahlawan dan Kebudayaan Melalui Patung di Pantai Losari Makassar Menggunakan Teknologi Machine Learning Kit Berbasis Android	Bertujuan utnuk mengenalkan tokoh pahlawan melalui aplikasi android berbassis machine learning dengan objek deteksi berupa patung
4	(Hikmah, Santoso, Soetrisno, 2021)	Perancangan antarmuka sistem pendeteksi uang kertas rupiah bagi penyandang low vision berbasis android	Menghasilkan Aplikasi yang dapat mengenali jenis uang kertas rupiah
5	(Alfikri, Utomo, Februariyanti, Nurwahyudi, 2022)	Pembangunan aplikasi penerjemah bahasa isyarat dengan metode cnn berbasis android	Menghasilkan aplikasi yang dapat menerjemahkan basaha isyarat berbasis android