BAB 2
TINJAUAN PUSTAKA

2.1 Gitar

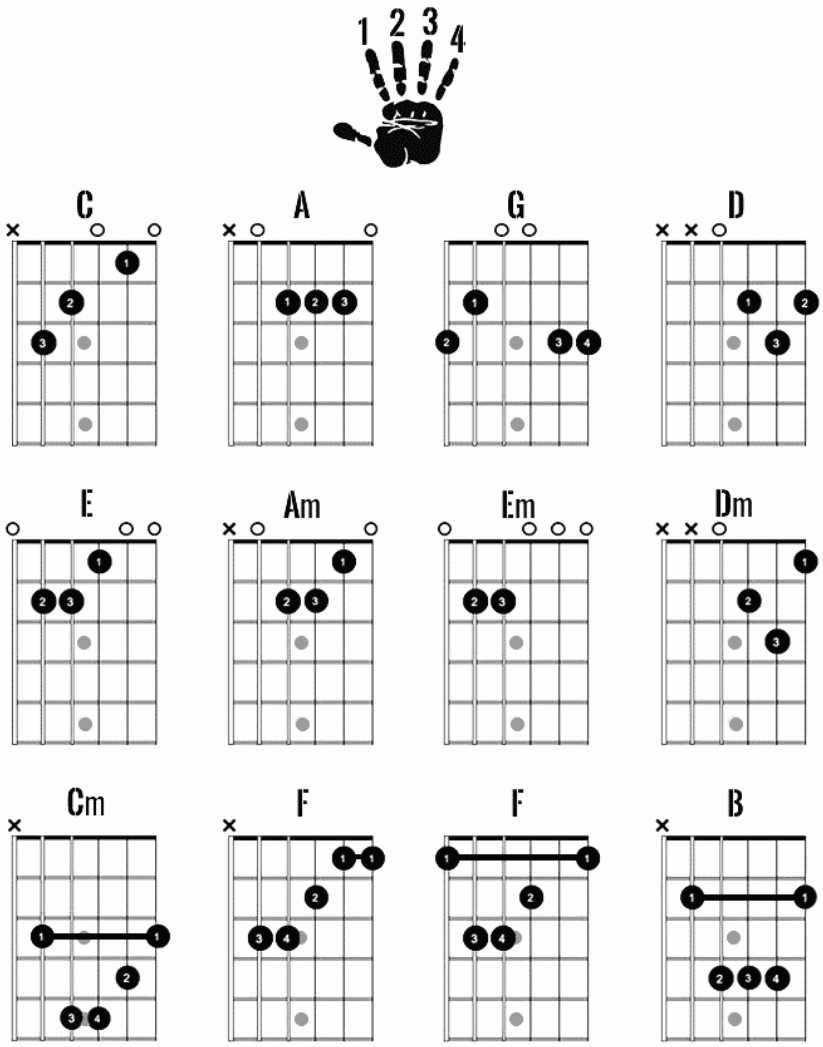
Gitar adalah sebuah alat musik berdawai yang dimainkan dengan cara dipetik, umumnya menggunakan jari. Gitar terbentuk atas sebuah bagian tubuh pokok dengan bagian leher yang padat sebagi tempat senar yang umumnya berjumlah enam didempetkan. Gitar secara tradisional dibentuk dari berbagai jenis kayu dengan senar yang terbuat dari nilon maupun baja. Gitar merupakan sebuah alat musik yang sering digunakan dalam musik modern. Bunyi yang dihasilkan oleh gitar elektrik pada pada prinsipnya adalah sama dengan gitar akustik, yaitu sumber bunyinya karena getaran string yang dipetik. Sehingga menyebabkan frekuensi bunyinya diproses oleh rongga-rongga *body* gitar. Sedangkan gitar elektrik, frekuensi getaran bunyi *string* gitar diproses oleh alat elektronik yang disebut *pickups*. Melalui *pickups* ke *amplifier* yang dapat didengarkan melalui speaker [10].

Gitar umumnya memiliki enam buah senar, namun ada beberapa gitar yang menggunakan kurang dari ataupun lebih dari enam senar [11]. Setiap senar memiliki ketebalan atau diameter yang berbeda. Senar pertama memiliki diameter paling kecil hingga senar keenam yang memiliki diameter paling besar. Pada gitar terdapat fret-fret yang membagi wilayah nada. Ketika sebuah senar ditekan maka akan membuat kontak dengan fret pada fingerboard. Fret sekarang betindak sebagai titik dukungan. Dua titik yang menunjang senar sekarang adalah fret dengan saddle, sehingga bagian bergetar dari senar lebih pendek. Posisi tangan saat menekan gitar pada fret tertentu akan mempengaruhi nada yang dihasilkan. Dimana panjang gelombang saat kita menekan pada fret diujung dan di tengah akan berbeda [11].

2.2 Chord Gitar

Chord adalah kumpulan tiga nada atau lebih yang bila dimainkan secara bersamaan terdengar harmonis. *chord* bisa dimainkan secara terputus-putus ataupun secara bersamaan. *Chord* ini digunakan untuk mengiringi suatu lagu.

Untuk memainkan alat musik gitar dibutuhkan pengetahuan tentang berbagai jenis *chord* dan juga mempunyai *feeling* yang baik saat memainkannya [4].



Gambar 2.1 Kunci Gitar Dasar

Sumber: https://google.com

Penjelasan dari Gambar 2.1 diatas yaitu:

a. Bulatan hitam menunjukan senar yang ditekan pada *fret*. *Fret* merupakan bagian dari gitar yang digunakan sebagai penampang senar saat ditekan.
b. Garis paling kanan atau garis pertama menunjukkan senar E tinggi
c. Garis ke 2 menunjukkan senar B
d. Garis ke 3 menunjukkan senar D

e. Garis ke 4 menunjukkan senar
f. Garis ke 5 menunjukkan senar A
g. Garis ke 6 atau garis paling kiri menunjukkan senar E rendah.

2.3 Frekuensi Pada Gitar

Frekuensi pada gitar merupakan hal penting yang menentukan tinggi rendahnya nada yang dihasilkan. Semakin tinggi frekuensi, semakin tinggi pula nadanya. Sebaliknya, semakin rendah frekuensi, semakin rendah pula nadanya. Adapun faktor faktor yang mempengaruhi frekuensi gitar:

1. Panjang senar

Semakin pendek senar, semakin tinggi frekuensi yang dihasilkan. Hal ini karena yang pendek bergetar lebih cepat.

2. Ketebalan senar

Semakin tipis senar, semakin tinggi frekuensi yang dihasilkan. Hal ini karena senar yang tipis lebih mudah bergetar.

3. Bahan senar

Bahan senar yang berbeda memiliki karakteristik getaran yang berbeda pula, sehingga menghasilkan frekuensi yang berbeda.

4. Penyetelan senar

Semakin kencang senar disetel, semakin tinggi frekuensi yang dihasilkan. Hal ini karena senar yang kencang bergetar lebih cepat.

5. Posisi fret

Menekan senar pada fret yang berbeda akan menghasilkan frekuensi yang berbeda. Hal ini karena bagian senar yang berbeda menjadi lebih pendek.

Tabel 2.1 Frekuensi Nada Gitar Standar [12].

Senar	Nada	Frekuensi (Hz)
E2	Mi	82.41
A2	La	110.0
D3	Re	146.83
G3	Sol	196.0
B3	Si	246.94
E4	Mi (Tinggi)	329.63

Huruf (A, B, C, D, E, F, G) menunjukkan nama not yang spesifik, dan angka (0, 1, 2, 3, 4, dst.) menunjukkan oktaf di mana not tersebut berada. Oktaf adalah rentang dari satu nada “C” ke nada “B” berikutnya, sebelum mencapai ke nada “C” lagi. Adapun Penjelasan frekuensi dan oktaf dari nada gitar standar:

1. E2: Nada E pada oktaf kedua, dengan frekuensi sekitar 82.41 Hz [12].
2. A2: Nada A pada oktaf kedua, dengan frekuensi sekitar 110.00 Hz [12].
3. D3: Nada D pada oktaf ketiga, dengan frekuensi sekitar 146.83 Hz [12].
4. G3: Nada G pada oktaf ketiga, dengan frekuensi sekitar 196.0 Hz [12].
5. B3: Nada B pada oktaf ketiga, dengan frekuensi sekitar 246.94 Hz [12].
6. E4: Nada E pada oktaf keempat, dengan frekuensi sekitar 329.63 Hz [12].

2.4 Jarak Frekuensi Tuning Gitar

Tuning gitar standar atau yang sering disebut dengan *standard tuning* merupakan metode penyetelan senar gitar yang paling umum digunakan oleh para pemain gitar di seluruh dunia. Dalam penyetelan ini, keenam senar gitar disetel pada frekuensi tertentu untuk menghasilkan nada-nada berikut: E2, A2, D3, G3, B3, dan E4, mulai dari senar paling tebal (senar ke-6) hingga senar paling tipis (senar ke-1) [11], [13].

Berikut tabel jarak frekuensi dari setiap senar gitar yang digunakan pada aplikasi:

Tabel 2.2 Jarak Frekuensi Tuning Gitar

Senar	Jarak Frekuensi (Hz)
E2	30.0 – 109.9
A2	110.0 – 145.9
D3	146.0 – 194.9
G3	195.0 – 245.9
B3	246.0 – 328.9

2.5 Tuner Gitar

Tuner atau Penyelaras nada gitar adalah rutinitas wajib yang harus dijalani seorang pemain gitar sebelum mulai memainkan gitarnya. Gitar yang tidak selaras tentu tidak bisa menghasilkan nada yang enak untuk didengar [14].

Bagi seorang pengguna gitar terutama pemula terkadang masih mengalami kesulitan dalam mengenali nada. Terutama dalam melakukan setem yang tepat pada gitar merupakan suatu hal yang sulit untuk dilakukan dengan cara pendengaran nada tanpa bantuan aplikasi untuk mencocokan setiap nadanya. Biasanya hal tersebut dikarenakan seorang pemain gitar pemula belum dapat mengingat nada pada masing-masing senar gitar tersebut. Kesulitan terjadi pada saat melakukan setem nada yang tepat pada masing-masing senar gitar [14].

Dengan semakin pesatnya perkembangan teknologi saat ini, khususnya diteknologi perangkat mobile berbasis Android maka penulis memanfaatkan fenomena ini dengan membuat suatu aplikasi smartphone berbasis mobile Android sebagai alat bantu penyelaras nada [14].

2.6 Computer Vision

Saat ini telah banyak berkembang sistem yang memamfaatkan fitur pengenalan wajah diantaranya yaitu sistem akses keamanan maupun sistem kontrol, pengolahan citra dan *computer vision* merupakan sebuah penemuan dibidang komputer yang digunakan untuk menghasilkan suatu sistem yang hampir mendekati dengan sistem visual manusia pada umumnya. Pengolahan citra adalah suatu jenis teknologi untuk menyelesaikan masalah mengenai pemrosesan gambar, sedangkan computer vision mempunyai tugas untuk membuat suatu keputusan tentang objek fisik nyata yang didapat dari perangkat atau sensor, *computer vision* membuat komputer dapat mengenali suatu citra layaknya manusia, salah satunya pengenalan gestur tangan. Penerapan pengolahan citra dan *computer vision* saat ini banyak di gunakan pada perusahaan atau lembaga untuk meningkatkan sistem keamanan berbasis data pada karakteristik tubuh atau perilaku yang disebut biometrik [15]. *Computer Vision* memiliki beberapa bidang, salah satunya yaitu *Object Detection.* Object Detection dapat mengidentifikasi objek yang berada pada citra dan letak objek yang dapat digunakan untuk berbagai keperluan seperti sistem keamanan dan pengawasan [16].

Saat ini tidak dapat dipungkiri bahwa perkembangan teknologi informasi sangat pesat. Selain perkembangan perangkat keras yang bertujuan untuk

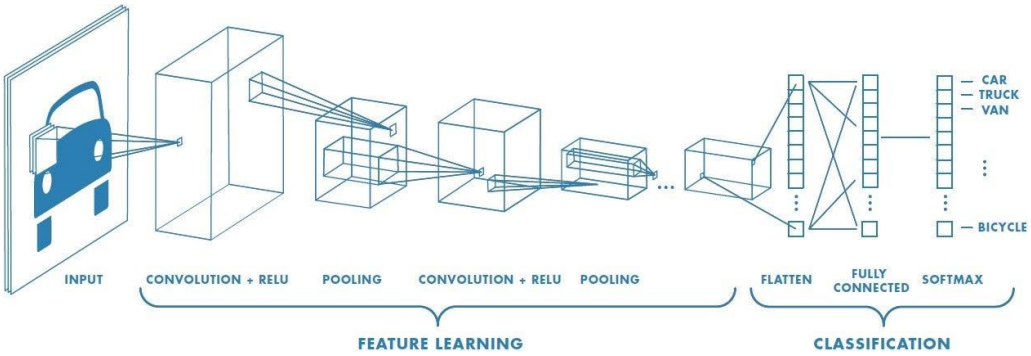
meningkatkan kinerja komputer, banyak juga pengembangan perangkat lunak yang dapat meniru kecerdasan manusia (kecerdasan buatan). Dengan berkembangnya dunia komputer dan meningkatnya kemampuan dan kecerdasan proses komputer, saat ini muncul ilmu-ilmu komputer yang memungkinkan komputer untuk secara otomatis mengambil informasi dari sebuah citra untuk tujuan pengenalan objek. Di masa depan, pengolahan citra ini diharapkan menjadi salah satu pilihan untuk menghitung jumlah manusia [16].

Baru-baru ini *Deep Learning* menjadi pusat perhatian dalam pengembangan Machine Learning, karena pada *Deep Learning* memberikan hasil yang optimal dalam *computer vision*. *Deep Learning* adalah metode pembelajaran otomatis yang dilakukan mesin untuk dapat memahami dan mengklasifikasikan suatu objek. Sebagian besar *Deep Learning* didominasi oleh pengenalan objek dan citra *(image recognition, face recognition, object detection, medical image classification* dan sebagainya). Penelitian ini menggunakan *Convolutional Neural Network* (CNN). *Convolutional Neural Network* merupakan salah satu model dari *Deep Learning* yang banyak digunakan untuk menganalisis citra/visual. Nama *Convolutional Neural Network* berarti bahwa pengoperasian jaringan tersebut menggunakan operasi matematika yang biasa dikenal sebagai konvolusi. Konvolusi adalah operasi antara dua function untuk menghasilkan function ketiga yang merupakan hasil modifikasi dari kedua function tersebut [16].

2.7 Convolutional Neural Network

Metode yang digunakan pada penelitian ini yaitu *Deep learning* yang memiliki kemampuan untuk memecahkan permasalahan dengan jumlah data yang besar. Dengan menggunakan *deep learning*, memungkinkan kita untuk membuat sistem yang mampu belajar dengan kecepatan dan akurasi yang diinginkan. Salah satu contoh metode *deep learning* yang digunakan pada penelitian ini yakni *Convolutional Neural Network*. Algoritma ini merupakan algoritma pengenalan yang efisien dan banyak digunakan pada pengolahan citra. CNN tidak berbeda dengan neural network lainnya seperti artificial neural network dimana mereka memiliki *bias*, *weight*, dan *activation function*. Namun yang membedakan CNN

dengan neural network lainnya adalah CNN memiliki layer khusus yang bernama *Convolutional Layer*. Pengolahan citra dilakukan dengan menggunakan kernel *filter*. *Filter* ini digunakan untuk mendapatkan pecahan-pecahan (*strides*) dari sebuah citra [17]. Proses ini dinamakan konvolusi. Berikut pada gambar 2 merupakan jaringan arsitektur Convolutional Neural Network:

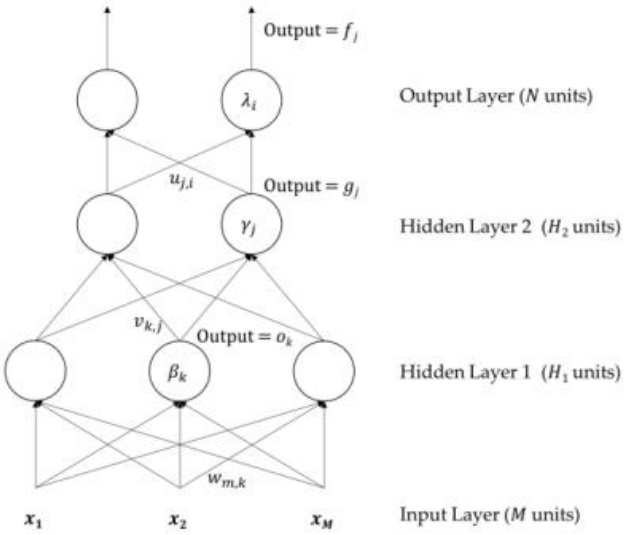


Gambar 2.2 Arsitektur Convolutional Neural Network

Sumber: https://medium.com/@kevinnickysetiawan/6-langkah-mudah-membuat-android-food-recognition-menggunakan-convolutional-neural-network-dengan-24443dbcd59d

2.8 *Deep Learning*

Deep Learning merupakan *artificial neural network* yang memiliki banyak lapisan dan *synapse weight*. *Deep learning* dapat menemukan relasi tersembunyi atau pola yang rumit antara masukan dan keluaran, yang tidak dapat diselesaikan menggunakan *multilayer perceptron* (3 lapisan). Keuntungan utama *deep learning* yaitu mampu merubah data dari *non-linearly separable* menjadi *linearly separable* melalui serangkaian transformasi (lapisan tersembunyi). Selain itu, *deep learning* juga mampu mencari *decision boundary* yang berbentuk non-linier, serta mengsimulasikan interaksi *non-linier* antar fitur. Jadi, input ditransformasikan secara non-linier sampai akhirnya pada output, berbentuk distribusi *class-assignment*. Pada pelatihan, deep learning menggunakan *back propagation*. Berikut contoh *Deep Learnin*g dengan 4 lapisan terdapat pada Gambar.



Gambar 2.3 Deep Learning dengan 4 Lapisan

2.9 OpenCV

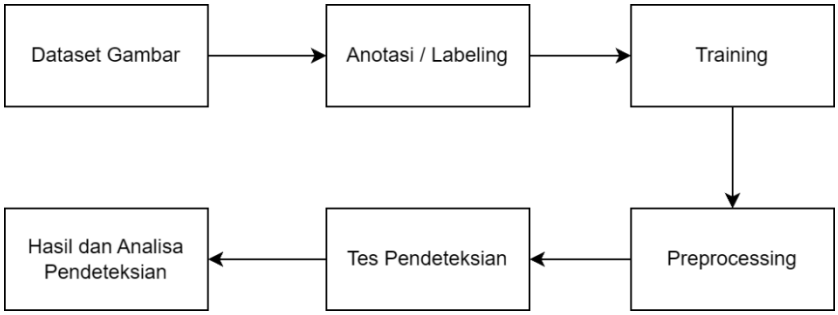
OpenCV (*Open-Source Computer Vision Library*) adalah sebuah *library open-source* python yang dikembangkan oleh intel yang berfokus untuk menyederhanakan programming terkait citra digital. Awalnya OpenCV merupakan library yang menggunakan bahasa pemrograman C/C++, dan sekarang telah dikembangkan ke dalam bahasa pemrograman python, java, matlab. OpenCV mempunyai banyak fitur, antara lain: pengenalan wajah, pelacakan wajah, deteksi wajah, kalman filtering dan berbagai jenis metode AI (Artificial Intelligence). OpenCV juga menyediakan berbagai algoritma sederhana terkait Computer Vision untuk low level API.

Selain itu, OpenCV juga menggunakan numpy. Numpy merupakan salah satu library python yang digunakan untuk mengimplementasi array dan matriks multidimensi bersama dengan operasi matematika tingkat tinggi. OpenCV terutama digunakan untuk menangkap data dari video langsung karena OpenCV berfokus pada pemrosesan gambar dan video yang ditangkap. Pada penelitian ini OpenCV difokuskan terutama dalam menangkap objek berupa video. OpenCV mencakup struktur data seperti skalar, point dan lain-lain yang digunakan untuk

membangun aplikasi OpenCV. Library OpenCV di import ke bahasa pemrograman python menggunakan kode ‘*import cv2* ‘[18].

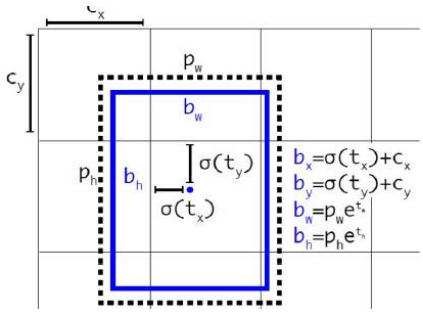
2.10 You Only Look Once (YOLO)

YOLO adalah sebuah metode untuk mendeteksi objek. YOLO memproses gambar secara real-time pada empat puluh lima (45) *frames per second* [19].



Gambar 2.4 Alur Pendeteksian Objek

YOLO mendeteksi objek dengan menggunakan unified model dimana sebuah single convolutional network memprediksi beberapa bounding boxes (kotak pembatas) serta probabilitas kelas di dalam kotak-kotak tersebut secara bersamaan. Pertama-tama, sistem YOLO membagi citra input ke dalam grid S × S. Jika pusat dari sebuah objek jatuh di dalam salah satu sel grid, maka sel grid itu bertanggung jawab untuk mendeteksi objek tersebut. Setiap sel grid memprediksi bounding boxes dan confidence score dari tiap bounding box tersebut. Confidence score merefleksikan seberapa yakin dan akurat model bahwa terdapat sebuah objek di dalam kotak tersebut. Setiap bounding box terdiri dari 5 prediksi: x, y, w, h, dan confidence. Koordinat (x, y) mewakili pusat dari kotak relatif ke batas sel grid. (w, h) atau lebar dan tinggi mewakili pusat dari kotak relatif ke gambar. Dan terakhir adalah confidence yang mewakili Intersection over Union (IoU) antara kotak prediksi dan kotak ground-truth. Setiap sel grid juga memprediksi probabilitas kelas. Probabilitas dikondisikan pada sel grid yang memuat objek dan hanya satu kelas probabilitas yang dideteksi per sel grid tanpa memperhitungkan jumlah bounding boxes [19].



Gambar 2.5 Prediksi Lokasi Bounding Box

2.11 Tensorflow

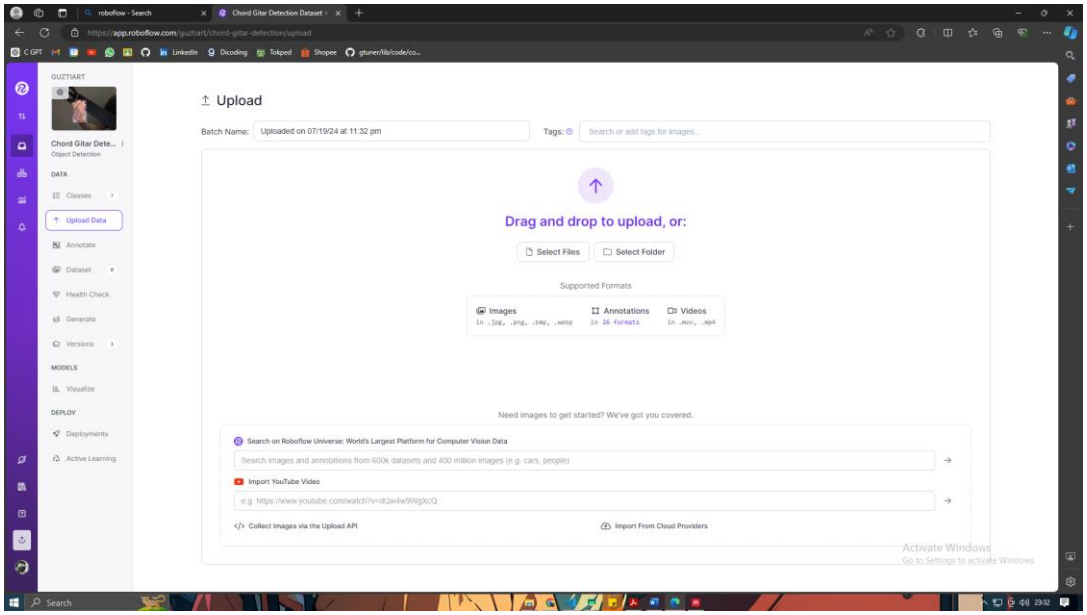
TensorFlow.js adalah pustaka pembelajaran mesin sumber terbuka yang dapat dijalankan di mana saja di mana JavaScript dapat dijalankan. Ini didasarkan pada TensorFlow asli yang ditulis dalam Python dan bertujuan untuk menemukan kembali pengalaman pengembang dengan serangkaian API pemanfaatan untuk ekosistem JavaScript.

Pembelajaran transfer melibatkan pembelajaran pengetahuan untuk membantu mempelajari sesuatu yang berbeda namun serupa. Demikian pula, jika Anda memiliki model pembelajaran mesin yang telah dilatih dan memiliki domain, seperti gambar, dapat digunakan kembali untuk melakukan tugas-tugas lain. Arsitektur Convolutional Neural Network (CNN) tradisional dapat dilihat bagaimana pembelajaran transfer dapat memanfaatkan jaringan yang telah dilatih untuk mempelajari objek baru [20].

2.12 Dataset

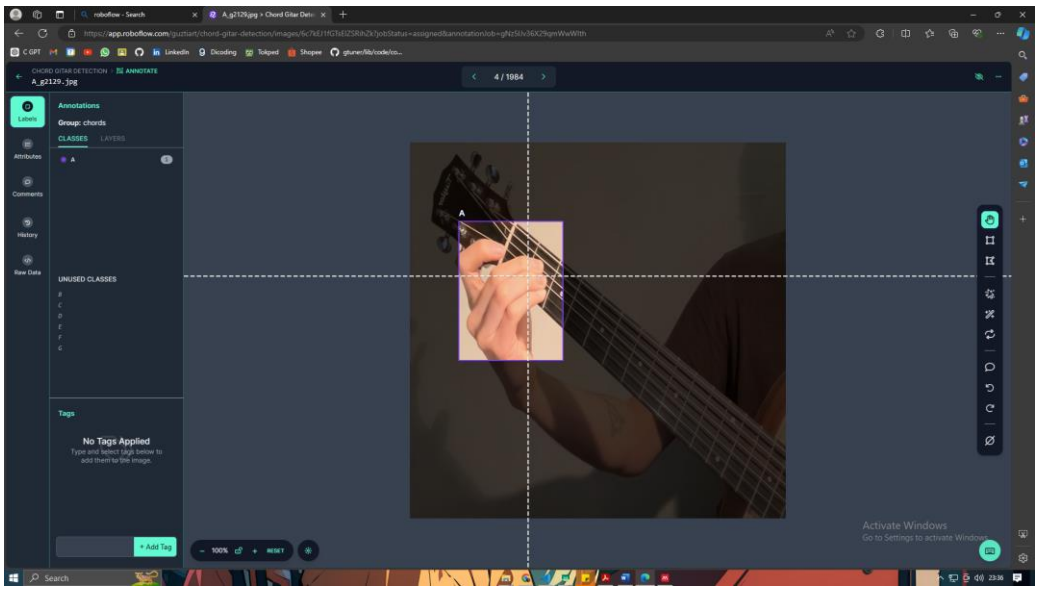
Dataset yang digunakan pada penelitian ini untuk identifikasi chord gitar melalui tahapan proses ekstraksi citra pada bentuk chord sebanyak 2684 citra terbagi menjadi 7 kelas. Peneliti membuat kelas dari chord major: A, B, C, D, E, F, G. Berikut merupakan tahapan pembuatan model *machine learning* dari dataset tersebut.

1. Pengumpulan dataset gambar dari setiap kelas
2. Membuat projek baru di situs website Roboflow, kemudian melakukan upload data citra



Gambar 2.6 Mengupload Dataset Gambar ke RoboFlow

3. Membuat anotasi atau labeling kelas dengan *bounding box* pada citra



Gambar 2.7 Pembuatan Label dan Bounding Box Pada Gambar

4. Setelah proses labeling, kemudian memasukkan gambar ke dalam dataset
5. Export dataset ke model YOLOv5.
6. Training model YOLOv5 dan diexport menjadi format PyTorch (.pt)
7. Konversi model PyTorch ke Tensorflow Lite

2.13 Aplikasi

Secara istilah aplikasi merupakan suatu program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Pada saat ini, aplikasi telah berkembang menjadi beberapa platform. Terdapat tiga *platform* utama pada pengembangan sebuah aplikasi, yaitu aplikasi berbasis mobile, web dan desktop. Secara umum, aplikasi dibagi menjadi tiga macam aplikasi yaitu [21].

1. Native Apps

Native apps adalah aplikasi yang dibangun secara spesifik untuk operating sistem tertentu. Jika aplikasi ini dibangun untuk sistem operasi iOS, maka aplikasi tersebut tidak akan dapat berjalan pada sistem operasi yang lain. Keuntungan utama dari jenis aplikasi ini adalah performanya yang tinggi serta memiliki user experience yang baik karena *developer* mengembangkan aplikasi ini menggunakan UI dari perangkat *native*. Secara keseluruhan, aplikasi ini menawarkan pengalaman pengguna yang lebih baik karena dapat bekerja dengan cepat, *responsive*, serta didistribusikan melalui *app store*.

2. Web Apps

Web apps berjalan menggunakan browser dan biasanya ditulis dalam HTML5, JavaScript atau CSS. Karena aplikasi menargetkan browser, maka web app dapat berjalan pada berbagai sistem operasi. Selain itu, aplikasi ini juga didesain secara responsive sehingga tampilan web dapat menyesuaikan ukuran layar pada perangkat yang digunakan oleh user.

3. Hybrid Apps

Hybrid apps dapat dikatakan seperti kombinasi dari dua macam aplikasi, yaitu native dan web. *Hybrid* apps memiliki dua bagian utama. Bagian pertama adalah kode back-end, dan yang kedua adalah *native shell* yang dapat diunduh dan memuat kode menggunakan tampilan web. *Hybrid* apps dinilai lebih mudah dan cepat untukdikembangkan dibanding dengan native apps. Namun kecepatan *hybrid* apps bekerja lebih lambat daripada aplikasi *native* karena bergantung pada kecepatan *browser user*.

2.14 Android

Android adalah sistem operasi berbasis linux yang dirancang untuk perangkat mobile, seperti telepon pintar (smartphone) dan Komputer Tablet (PDA). Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tanggal 5 November 2007, bersamaan dengan didirikannya *Open Handset Alliance* (OHA), konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Terdapat dua jenis distributor resmi dari sistem operasi android. Pertama yang mendapat dukungan penuh dari Google atau Google Mail Service (GMS) dan yang kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung dari Google atau dikenal sebagai *Open Handset Distribution* (OHD) [22].

Pada masa saat ini kebanyakan vendor-vendor smartphone sudah memproduksi smartphone berbasis android. Hal ini karena android itu adalah sistem operasi yang open source sehingga bebas didistribusikan dan dipakai oleh vendor manapun. Tidak hanya menjadi sistem operasi di smartphone, saat ini android menjadi pesaing Apple pada sistem operasi Tablet PC. Pesatnya pertumbuhan android adalah karena android itu sendiri merupakan platform yang lengkap, tersedianya tools pengembangan, adanya fasilitas market dan didukung oleh komunitas open source [23].

Dalam perkembangannya, android telah mengalami beberapa pembaruan sejak pertama kali rilis. Android saat tulisan ini dibuat sudah sampai pada generasi ke 14, yang diberi nama *Upside Down Cake*. Perkembangan versi android dari pertama kali rilis hingga sekarang dapat dilihat pada tabel berikut.

Tabel 2.3 Versi Android

No.	Versi Android	Nama	Tanggal Rilis
1	1.0	Astro/Alpha	23 September 2008
2	1.1	Bender/Beta	9 Februari 2009

3	1.5	Cupcake	30 April 2019
4	1.6	Donut	15 September 2009
5	2.0 – 2.1	Éclair	26 Oktober 2009
6	2.2 – 2.23	Froyo	20 Mei 2010
7	2.3 – 2.37	Gingerbread	6 Desember 2010
8	3.0 – 3.2.6	Honeycomb	22 Februari 2011
9	4.0 – 4.0.4	Ice Cream Sandwich	19 Oktober 2011
10	4.1 – 4.3.1	Jelly Bean	27 Juni 2012
11	4.4 – 4.4.4	Kitkat	31 Oktober 2013
12	5.1 – 5.1.1	Lollipop	12 November 2014
13	6.0 – 6.0.1	Marshmallow	5 Oktober 2015
14	7.1 – 7.1.2	Nougat	22 Agustus 2016
15	8.0 – 8.1	Oreo	21 Agustus 2017
16	9	Pie	6 Agustus 2018
17	10	Quince Tart/ Queen Cake	3 September 2019
18	11	Red Velvet Cake	8 September 2020
19	12	Snow Cone	4 Oktober 2021
20	13	Tiramisu	10 Februari 2022
22	14	Upside Down Cake	4 Oktober 2023

2.15 Arsitektur Android

Secara umum, arsitektur android terdiri dari lima lapisan, yaitu Applications and Widgets, Applications Framework, Libraries, Android Runtime dan Linux Kernel. Penjelasan dari tiap lapisan tersebut adalah sebagai berikut [23]:

1. Application and Widgets

Applications dan Widgets merupakan lapisan yang paling tampak pada pengguna ketika menjalankan program, pengguna hanya akan melihat program ketika digunakan tanpa mengetahui proses yang terjadi dibalik lapisan aplikasi.

Lapisan ini berjalan dalam android runtime dengan menggunakan kelas dan *service* yang tersedia pada *framework* aplikasi.

2. Application Framework

Application Framework merupakan lapisan dimana developer melakukan pengembangan/pembangunan aplikasi yang akan dijalankan pada sistem operasi android. Lapisan ini menyediakan berbagai class yang bisa digunakan olehdeveloper untuk mempermudah dalam proses pengembangan/pembangunan aplikasi. Termasuk juga berhubungan dengan akses ke hardware dan manajemen user-interface, memori dan sumber daya aplikasi. Komponen yang termasuk *Application Frameworks* antara lain *Views, Content Provider, Resource Manager, Notification,* dan *Activity Manager*.

3. Libraries

Libraries merupakan lapisan dimana fitur-fitur android berada. Developer dapat mengakses libraries untuk menjalankan aplikasinya, lapisan ini berjalan diatas kernel yang didalamnya terdapat kumpulan library C dan C++ seperti Libc dan SSL Selain itu terdapat juga beberapa library lainnya diantaranya:

a. Library Media untuk pemutaran media audio dan video.
b. Library Graphics untuk manajemen tampilan grafis 2D dan 3D.
c. Library SQLite untuk dukungan database.
d. Library LiveWebcore mencakup engine embedded web view.
e. Library SSL dan WebKit untuk integrasi web browser dan keamanan internet.

4. Android Runtime

Android *runtime* merupakan lapisan yang membuat aplikasi Android dapat dijalankan, dimana dalam prosesnya menggunakan implementasi Linux yang terdiri dari *Core Libraries* dan *Dalvik Virtual Machine* (DVM). *Core Libraries* merupakan serangakaian *library* Java yang terdiri dari berbagai macam fungsi pada bahasa pemrograman Java, sedangkan DVM merupakan *Java Virtual Machine* (semacam JVM pada Java ME) yang secara khusus dijalankan pada android.

5. Linux Kernel

Linux kernel merupakan lapisan paling bawah pada arsitektur android, dimana sistem operasi android itu berada. Google menggunakan kernel linux versi 2.6 untuk membangun sistem android, yang mencakup memory management, security setting, power managemnet dan beberapa driver hardware lainnya. Lapisan ini berperan sebagai abstraction layer antara hardware dan keseluruhan software.

2.16 Android Studio

Android Studio adalah Integrated Development Environment (IDE) resmi untuk sistem operasi android, pengganti Eclipse Android Development Tools (ADT) sebagai IDE utama untuk pengembangan aplikasi Android. IDE ini diluncurkan pada tanggal 16 Mei 2013. IDE ini dibangun di atas perangkat lunak IntelliJ IDEA JetBrains dan dirancang khusus untuk pengembangan Android. IDE ini tersedia untuk diunduh pada sistem operasi berbasis Windows, macOS dan Linux [24]

Android studio memilki fitur yang sangat membantu developer untuk membangun aplikasi android, berikut ini beberapa fitur dari Android Studio yang di ambil dari situs resmi di https://developer.android.com/studio.

2.17 Dart

Dart adalah bahasa pemrograman terstruktur open source untuk membuat aplikasi web berbasis browser yang kompleks. Pengguna dapat menjalankan aplikasi yang dibuat di Dart baik dengan menggunakan browser yang secara langsung mendukung kode Dart atau dengan mengkompilasi kode Dart pengguna ke JavaScript. Dart memiliki sintaks yang familiar, dan berbasis kelas, diketik secara opsional, dan *singlethreaded*. Ini memiliki model konkurensi yang disebut isolat yang memungkinkan eksekusi paralel. Selain menjalankankode Dart di *browser* web dan mengubahnya menjadi JavaScript, Pengguna juga dapat menjalankan kode Dart di baris perintah. dihosting di mesin virtual Dart, memungkinkan klien dan bagian server dari aplikasi pengguna dikodekan dalam bahasa yang sama. Sintaks bahasanya sangat mirip dengan Java, C#, dan

JavaScript. Salah satu tujuan utama Dart adalah agar bahasa itu tampak akrab. Ini adalah skrip Dart kecil, yang terdiri dari satu fungsi yang disebut main [25].

2.18 Flutter

Pengembangan dari aplikasi ini menggunakan framework Flutter dalam membuat aplikasi sehingga aplikasi dapat dijalankan pada sistem operasi Android dan iOS. Framework Flutter digunakan untuk pembuatan aplikasi *mobile* [26].

Flutter adalah sebuah SDK atau *framework open-source* yang dikembangkan oleh Google untuk membuat atau mengembangkan aplikasi yang dapat berjalan dalam sistem operasi Android dan iOS. Flutter menggunakan bahasa pemrograman Dart dalam pengkodean. Perbedaan framework Flutter dengan yang lainnya yaitu dalam build aplikasi, pada framework ini semua kodenya di compile dalam kode native-nya (Android NDK, LLVM, AOTcompiled) tanpa ada interpreter pada prosesnya sehingga proses compile-nya menjadi lebih cepat [26].

2.20 Pitch Detector Dart Library

'*pitch_detector_dart*' adalah sebuah *library* Dart yang digunakan untuk mendeteksi pitch atau nada dari audio input. *Library* ini memungkinkan pengembang untuk mengidentifikasi frekuensi dasar dari suara yang ditangkap, yang sangat berguna dalam berbagai aplikasi seperti tuning alat musik, analisis suara, dan pengenalan musik. Fitur utama dari library ini yaitu:

1. Mengidentifikasi frekuensi dasar dari sinyal audio.
2. Menggunakan algoritma yang efisien untuk memastikan deteksi pitch yang cepat dan akurat.
3. Mendukung pemrosesan audio secara *real-time*, memungkinkan aplikasi seperti tuner gitar langsung
4. Dapat digunakan di berbagai *platform* yang didukung oleh Dart, termasuk aplikasi web dan *mobile*.

Berikut merupakan contoh penggunaan library:

Tabel 2.4 Contoh Penggunaan Pitch Detector Dart Library

```
1. import 'package:pitch_detector_dart/pitch_detector.dart';
```

```
void main() {
  // Membuat instance dari PitchDetector
  final detector = PitchDetector();

  // Input data audio (biasanya berupa array of audio samples)
  List<double> audioSamples = [ /* Audio samples */ ];

  // Mendeteksi pitch dari audio samples
  PitchResult result = detector.getPitch(audioSamples);

  // Menampilkan hasil deteksi pitch
  if (result.pitched) {
    print('Detected pitch: ${result.pitch} Hz');
  } else {
    print('No pitch detected');
  }
}
```

2.19 Guitar Chord Dart Library

'guitar_chord_library' adalah sebuah library Dart yang digunakan untuk mengakses berbagai informasi tentang chord gitar, termasuk bentuk chord, posisi jari, dan variasi chord. Library ini sangat berguna untuk pengembang yang ingin membuat aplikasi terkait musik, terutama aplikasi pembelajaran gitar atau referensi chord. Fitur utama dari library ini yaitu:

1. Menyediakan akses ke berbagai bentuk chord gitar.
2. Menyediakan informasi mengenai penempatan jari pada fret gitar untuk memainkan *chord* tertentu.
3. Menyediakan variasi dari chord yang sama dalam posisi yang berbeda pada *fretboard*.
4. Memiliki kemampuan untuk mengubah kunci *chord* ke posisi lain di fretboard.

Berikut contoh penggunaan library:

Tabel 2.5 Contoh Pengguaan Guitar Chord Dart Library

```
import 'package:guitar_chord_library/guitar_chord_library.dart';

void main() {
  /// your favour instrument
  /// InstrumentType.guitar
  /// InstrumentType.ukulele
  var instrument = GuitarChordLibrary.instrument(InstrumentType.guitar);

  /// instrument string count
  print(instrument.stringCount);

  /// all marjor keys
  print(instrument.getKeys()); //use sharp
  print(instrument.getKeys(true)); //use flat
```

```
15.
16.   /// all chords by major key
17.   print(instrument.getChordsByKey('C#')); //use sharp
18.   print(instrument.getChordsByKey('C#', true)); //use flat
19.
20.   /// all positions of Cmajor chord
21.   final chordPostions = instrument.getChordPositions('C', 'major')!;
22.
23.   for (var position in chordPostions) {
24.     print(
25.         'baseFret: ${position.baseFret}\nfrets: ${position.frets}\nfingers:
${position.fingers}\n');
26.   }
27. }
```

2.21 Unified Modelling Language (UML)

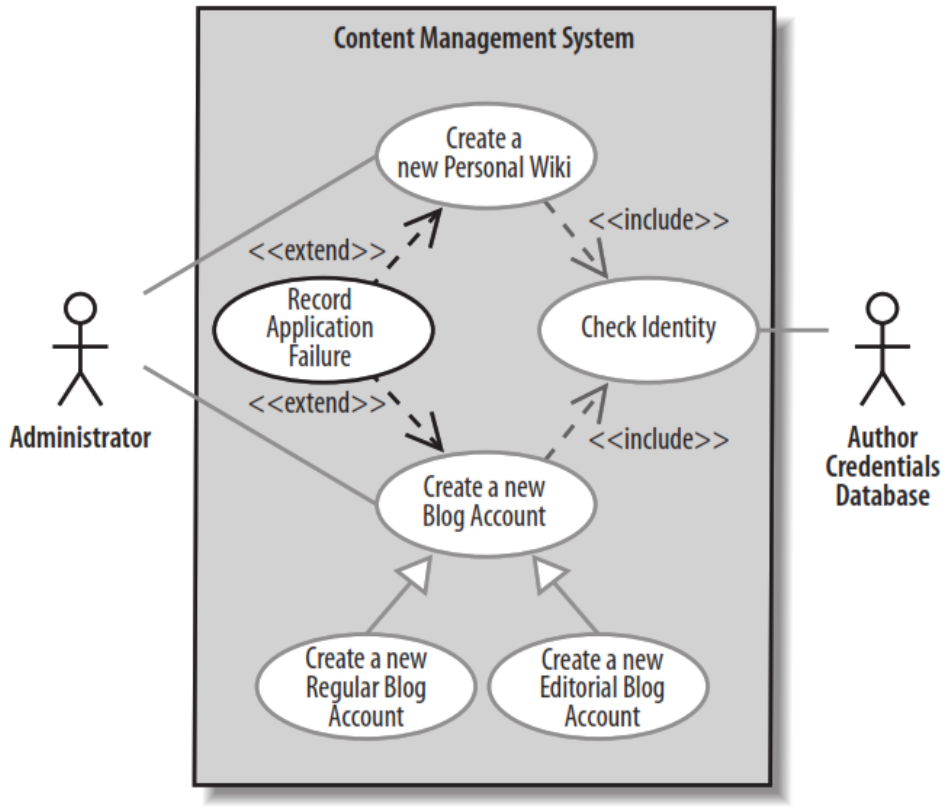
Pada tahap Desain, penggunaan diagram UML (Unified Modeling Language) dapat menggambarkan proses perancangan perangkat lunak secara lebih terstruktur. Unified Modeling Language UML) dalah sebuah bahasa pemodelan grafis yang digunakan untuk merancang, mendokumentasikan, dan memahami sistem perangkat lunak. UML memungkinkan para pengembang perangkat lunak untuk menggambarkan struktur, perilaku, dan interaksi sistem secara visual dengan menggunakan notasi yang konsisten [27].

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. UML merupakan bahasa pemodelan yang bertujuan untuk pengembangan dalam bidang rekayasa perangkat lunak yang dimaksudkan untuk memberikan cara standar untuk memvisualisasikan desain suatu sistem. Penciptaan UML pada awalnya dimotivasi oleh keinginan untuk membakukan sistem notasi yang berbeda dan pendekatan untuk desain perangkat lunak [28].

Berdasarkan pemaparan mengenai UML dari beberapa sumber referensi, maka dapat disimpulkan UML merupakan alat bantu dalam melakukan pemodelan yang saling berhubungan secara langsung dalam pembangunan sebuah sistem agar lebih efektif. Di dalam UML terdapat berbagai macam diagram, empat di antaranya yang akan digunakan pada penelitian ini adalah *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram*, dan *Class Diagram*.

2.22 Use Case Diagram

Use Case Diagram adalah uraian sekelompok yang saling terkait satu dengan lainnya dan membentuk sistem secara teratur yang dilakukan oleh sebuah aktor. Use case digunakan untuk membentuk tingkah laku suatu benda dalam sebuah model serta direalisasikan oleh sebuah kolaborasi. Hal yang ditekankan dalam diagram ini adalah "apa" yang dapat diperbuat oleh sistem bukan "bagaimana". Use case merepresentasikan interaksi antara *user* dengan sistem dan menyatakan sebuah aktivitas atas pekerjaan tertentu. Aktor merepresentasikan sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan–pekerjaan tertentu.



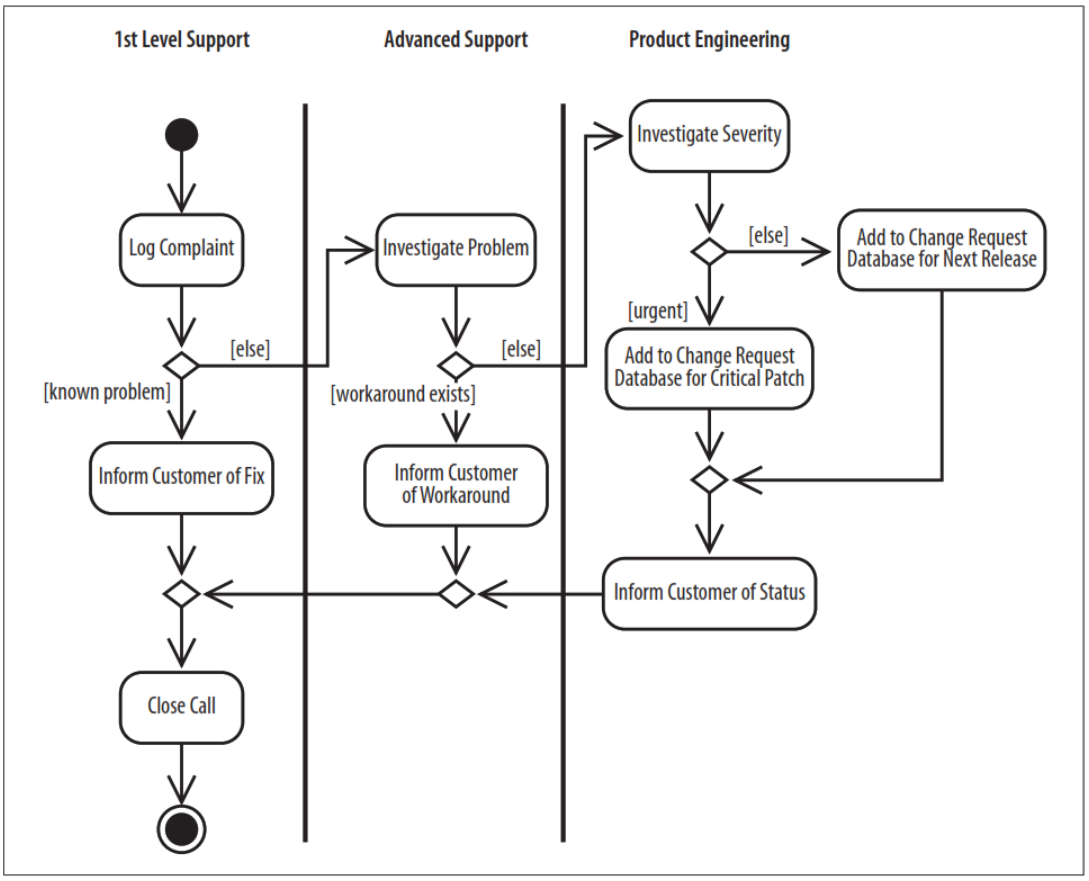
Gambar 2.8 Contoh Use Case Diagram

Sumber: Buku Learning UML 2.0, O'Reilly Media, 2006 [29].

2.23 Activity Diagram

Activity Diagram menggambarkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini mirip dengan *flowchart* karena

memodelkan *workflow* dari satu aktivitas ke aktivitas lainnya atau dari aktivitas ke status. Membuat activity diagram pada awal pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. Activity diagram juga bermanfaat untuk menggambarkan *parallel behaviou*r atau menggambarkan interaksi antara beberapa *use case*.



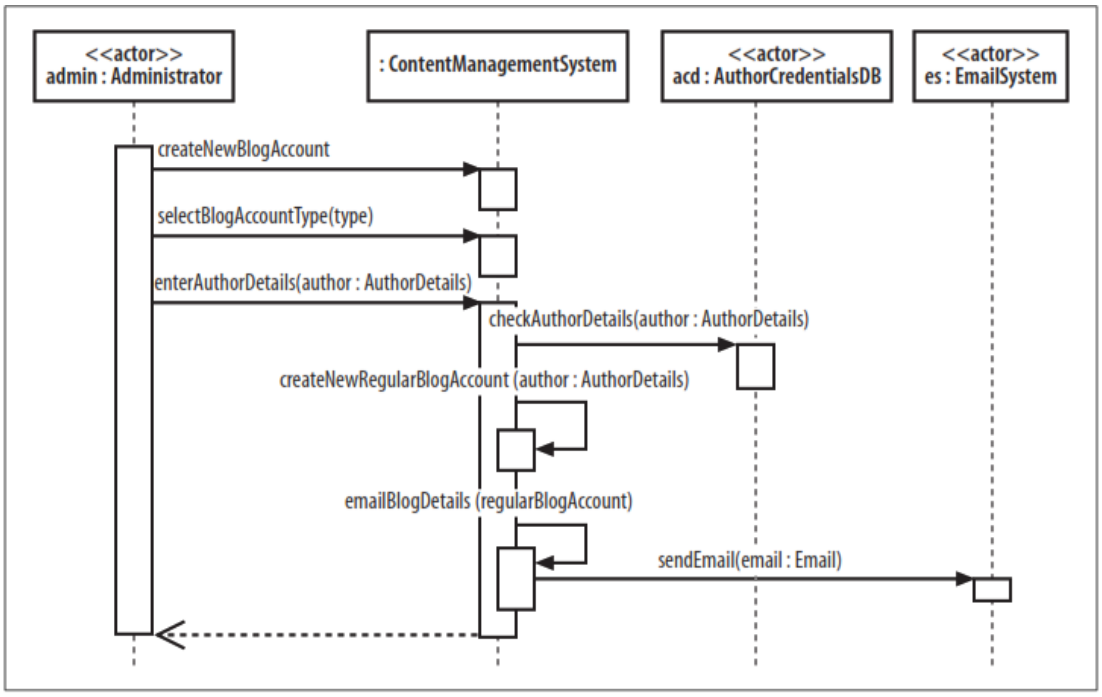
Gambar 2.9 Contoh Activity Diagram

Sumber: Buku Learning UML 2.0, O'Reilly Media, 2006 [29].

2.24 Sequence Diagram

Sequence Diagram menunjukkan interaksi objek yang diatur dalam urutan waktu. Ini menggambarkan objek dan kelas yang terlibat dalam skenario dan urutan pesan yang dipertukarkan antara objek yang diperlukan untuk menjalankan fungsionalitas skenario. Sequence diagram biasanya dikaitkan dengan realisasi use case dalam logical view dari sistem yang sedang dikembangkan. Sequence diagram kadang-kadang disebut diagram acara atau skenario acara. Sebuah

sequence diagram sebagai garis-garis paralel paralel menunjukkan berbagai proses atau objek yang hidup secara bersamaan, dan, sebagai panah horizontal, pesan-pesan dipertukarkan di antara mereka, dalam urutan terjadinya. Ini memungkinkan spesifikasi skenario runtime sederhana secara grafis [28].

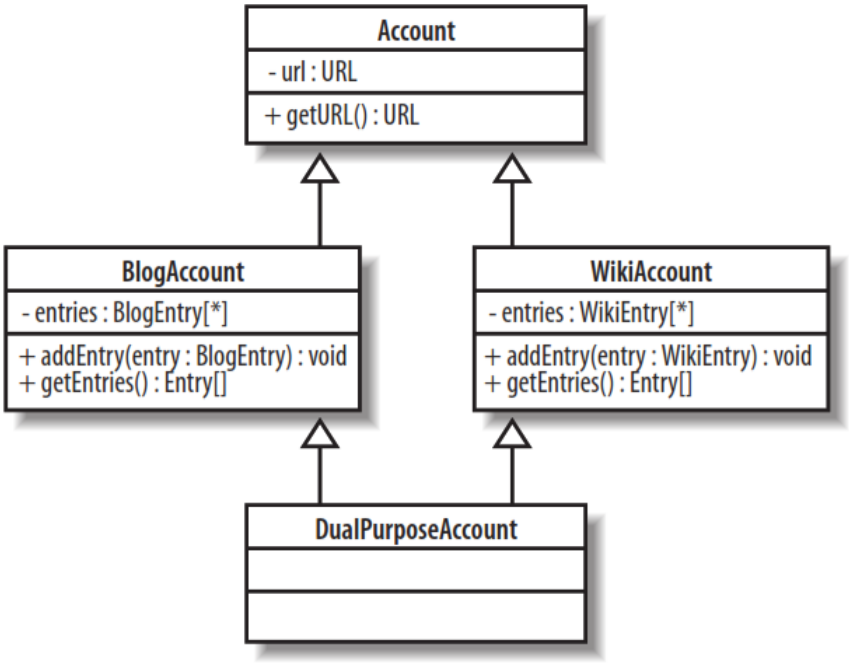


Gambar 2.10 Contoh Sequence Diagram

Sumber: Buku Learning UML 2.0, O'Reilly Media, 2006 [29].

2.25 Class Diagram

Dalam rekayasa perangkat lunak, class diagram dalam UML adalah jenis diagram struktur statis yang menggambarkan struktur sistem dengan menunjukkan kelas sistem, atributnya, operasi (atau metode), dan hubungan antar objek. Class diagram adalah blok bangunan utama pemodelan berorientasi objek. Ini digunakan terperinci menerjemahkan model ke dalam kode pemrograman. Class diagram juga dapat digunakan untuk pemodelan data. Kelas-kelas dalam class diagram mewakili elemen utama, interaksi dalam aplikasi, dan kelas yang akan diprogram [28].



Gambar 2.11 Contoh Class Diagram

Sumber: Buku Learning UML 2.0, O'Reilly Media, 2006 [29].