PENERAPAN METODE EDGE DETECTOR PADA *MACHINE LEARNING* UNTUK MENGANALISIS MASA GENERATIF TUMBUHAN

SKRIPSI

Diajukan sebagai salah satu syarat untuk kelulusan
Jenjang Strata Satu (S1)
Pada Program Studi Teknik Informatika

Disusun Oleh:

Muhammad Helmi Rafsanjani
361701012

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER INDONESIA MANDIRI

BANDUNG

2020

LEMBAR PERSETUJUAN

PENERAPAN METODE EDGE DETECTOR PADA *MACHINE LEARNING* UNTUK MENGANALISIS MASA GENERATIF TUMBUHAN

EDGE DETECTOR IMPLEMENTATION WITH MACHINE LEARNING FOR PLANTS GENERATIVE PHASE ANALYSIS

Oleh:

Muhammad Helmi Rafsanjani
361701012

Skripsi ini telah di terima dan disahkan

Untuk memenuhi persyaratan mencapai gelar

SARJANA TEKNIK INFORMATIKA

Pada

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
INDONESIA MANDIRI

Bandung, 15 Januari 2020

Disahkan Oleh

Ketua Program Studi Teknik Informatika,	Dosen Pembimbing,
Chalifa Chazar, S.T., M.T.	Chalifa Chazar, S.T., M.T.
NIDN: 437300079	NIDN: 437300079

ii

LEMBAR PERSETUJUAN REVISI TUGAS AKHIR

PENERAPAN METODE EDGE DETECTOR PADA *MACHINE LEARNING* UNTUK MENGANALISIS MASA GENERATIF TUMBUHAN

EDGE DETECTOR IMPLEMENTATION WITH MACHINE LEARNING FOR PLANTS GENERATIVE PHASE ANALYSIS

Telah melakukan sidang Skripsi pada hari Selasa, 26 Januari 2021 dan telah melakukan revisi sesuai dengan masukan pada saat sidang Tugas Akhir

Menyetujui,

No	Nama	Keterangan	Tanda Tangan
1	Chalifa Chazar, S.T., M.T.	Pembimbing	
2	Yudhi W. Arthana R., S.T., M.Kom.	Penguji 1	
3	DR. Chairuddin, IR., M.M., M.T.	Penguji 2	

Bandung, 29 Juni 2021

Mengetahui,

Ketua Program Studi

Chalifa Chazar, S.T., M.T.

NIDN. 0421098704

iii

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa:

1. Skripsi ini adalah Asli dan belum pernah diajukan untuk mendapatkan gelar Akademik, baik di Sekolah Tinggi Manajemen & Komputer Indonesia Mandiri maupun Perguruan Tinggi lainnya.
2. Skripsi ini murni merupakan karya penelitian saya sendiri dan tidak menjiplak karya lain. Dalam hal adabantuan atau arahan dari pihak lain maka telah saya sebutkan identitas dan jenis bantuannya di dalam lembar ucapan terima kasih.
3. Seandainya ada karya pihak lain yang ternyata memiliki kemiripan dengan karya saya ini, maka hal ini adalah diluar pengetahuan saya dan terjadi tanpa kesengajaan dari pihak saya.

Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terbukti adanya kebohongan dalam pernyataan ini, makan saya bersedia menerima sanksi akademik sesuai norma yang berlaku di Sekolah Tinggi Manajemen Informatika dan Komputer Indonesia Mandiri.

Bandung, 15 Januari 2020
Yang Membuat Pernyataan

Muhammad Helmi Rafsanjani

361701012

PENERAPAN MACHINE LEARNING DENGAN SISTEM INTELIJENSI VISUAL APLIKASI ANDROID UNTUK ANALISIS PADA MASA GENERATIF TUMBUHAN

ABSTRAK

Pengenalan objek yang dilakukan manusia merupakan bagian penting yang selalu terpakai oleh manusia pada umunya, lalu dengan kecerdasan buatan yang semakin mendominasi membuat teknologi semakin mendekati kemampuan atau bahkan melebihi kemampuan indera manusia. Sistem intelijensi visual merupakan bidang yang memperdalam cara pandang teknologi dalam memberikan informasi dan kalkulasi secara diskrit dengan dukungan kecerdasan buatan. Lain halnya dengan vegetasi yang ada di bumi, vegetasi tidak mengalami perubahan ataupun berbedaan yang signifikan dalam waktu yang singkat bahkan sejak zaman dahulu pun tanaman tetap seperti itu adanya, yang membedakan adalah cara manusia mengelola tanaman agar bisa bermanfaat dalam memenuhi kebutuhan primer (Sandang, Pangan Papan). Orang orang di bidang pertanian mulai menggunakan teknologi untuk membantu proses pertumbuhan vegetasi, tujuannya tentu saja agar lebih mangkus, efisien, mengurangi resiko, dan hemat biaya. Penulis bertujuan untuk membuat program yang dapat membantu menganalisa setiap vegetasi dengan sistem intelijensi visual dan memprosesnya dengan metode *machine learning* agar dapat membantu petani mengambil keputusan yang tepat saat mengelola pertanian.

Kata Kunci: Sistem intelijensi visual, *Machine Learning*, pertanian, vegetasi

MACHINE LEARNING IMPLEMENTATION WITH VISUAL INTELLIGENCE SYSTEM ANDROID APPLICATION FOR ANALYSIS IN PLANTS GENERATIVE PHASE

ABSTRACT

Object recognition from human being is an important role that always been used commonly, and then artificial inteligence increasing dominating lately make technology closer or even far more better than human sense. Visual Intelligence System is a field that deepens the perspective of technology in providing discrete information and calculations with the support of artificial intelligence. Different from the vegetation on earth, vegetation does not experience changes or significant differences in a short time even since ancient times even plants remain as it is, the difference is the way humans manage plants to be useful in aquiring primary needs (Clothing, Food, Board). People in agriculture have begun to use technology to help the process of vegetation growth, the aim is of course to be more efficient, efficient, reduce risk and save costs. The author aims to create a program that can help analyze each vegetation with a visual intelligence system and process it with the method of machine learning in order to help farmers make the right decision when managing agriculture.

Keyword*: Visual Intelligence System, Machine Learning, Agriculture, Vegetation*

UCAPAN TERIMAKASIH

Dalam penyusunan skripsi ini tidak terlepas dukungan dari berbagai pihak. Peneliti secara khusus mengucapkan terima kasih yang sebesar besarnya kepada semua pihak yang telah membantu. Peneliti banyak menerima bimbingan, petunjuk dan bantuan serta dorongan dari berbagai pihak yang bersifat moral maupun materil. Pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Allah SWT dengan segala rahmat dan karunia-Nya yang memberikan kekuatan dan kelancaran bagi peneliti dalam menyelesaikan skripsi ini.
2. Kedua orang tua tercinta yang selama ini telah membantu penelitian dalm bentuk perhatian, kasih sayang, semangat, serta doa yang tidak henti-hentinya mengalir demi kelancaran dan kesuksesan peneliti dalam menyelesaikan skripsi ini. Dan juga Saudara-Saudari kandung penulis terima kasih atas bantuan doa serta dukungannya.
3. Bapak Dr. Chairuddin, Ir., M.M., M. T ., selaku ketua Sekolah Tinggi Manajemen Informatika dan Komputer Indonesia Mandiri.
4. Ibu Chalifa Chazar, S,T., M.M., M.T., selaku Ketua Program Jurusan Teknik Informatika dan juga selaku dosen pembimbing. Terima kasih atas kesabaran, bimbingan, arahan, semangat, ilmu, dan doa yang terus tercurahkan agar peneliti bisa menyelesaikan skripsi.

5. Bapak Dr. Ichsan Ibrahim, S.SI, M.SI., yang telah membantu membimbing peneliti dalam menyelesaikan skripsi ini.
6. Segenap dosen dan seluruh staf akademik yang selalu membantu dalam memberikan ilmu, fasilitas, serta pendidikan pada peneliti hingga dapat menunjang dalam penyelesaian skripsi ini.
7. Seluruh teman-teman penulis dari kelas reguler dan juga kelas karyawan yang sama sama berjuang menyusun skripsi dan saling membantu memberi semangat satu sama lain.
8. Seluruh teman unit kegiatan mahasiswa Imforsil yang tidak henti hentinya memberikan bantuan doa agar peneliti bisa menyelesaikan skripsi ini.
9. Serta masih banyak lagi pihak-pihak yang sangat berpengaruh dalam proses penyelesaian skripsi yang tidak bisa peneliti sebutkan satu persatu.

Semoga Allah SWT senantiasa membalas semua kebaikan yang telah diberikan. Semoga penelitian ini dapat bermanfaat bagi peneliti sendiri dan juga kepada para pembaca pada umumnya.

KATA PENGANTAR

Bismillahirrahmaanirrahim, Puji dan Syukur penulis panjatkan kepada Allah SWT karena atas taufiq dan hidayah-Nya, penulis dapat menyelesaikan skripsi yang berjudul "**PENERAPAN *MACHINE LEARNING* DENGAN SISTEM INTELIJENSI VISUAL APLIKASI ANDROID UNTUK ANALISIS PADA MASA GENERATIF TUMBUHAN"**. Skripsi ini disusun sebagai syarat untuk menyelesaikan jenjang pendidikan Strata Satu (S1) di Program Studi Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer Indonesia Mandiri. Penulis menyadari dalam penulisan Laporan Tugas Akhir ini, baik bentuk maupun isinya masih terdapat banyak kekurangan karena keterbatasan pengetahuan dan kemampuan serta pengalaman yang penulis miliki. Untuk itu dengan kerendahan hati penulis mengharapkan kritik dan saran yang membangun untuk kemajuan penulis dikemudian hari. Penulis berharap tugas akhir ini bisa menjadi salah satu sumber yang dapat bermanfaat.

Bandung, 29 Juni 2021
Penulis

Muhammad Helmi Rafsanjani
361701012

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Kecerdasan buatan dan manusia kini menjadi sebuah hubungan yang sudah biasa ditemukan dalam kehidupan sehari hari. Bidang apapun pasti menggunakan teknologi yang semakin maju untuk berbagai manfaat yang dapat diperoleh, salah satunya ialah bidang pertanian. FAO memprediksi bahwa pada tahun 2020, sekitar 75% penduduk di negara-negara berkembang di Afrika, Asia, dan Amerika Latin tinggal di kawasan perkotaan (*Food and Agriculture* Organization, 2020) . Kondisi ini mendorong masyarakat untuk mulai mencoba memenuhi kebutuhan pangan secara mandiri serta memperbaiki kondisi lingkungan agar tercipta lingkungan yang sehat dan berkualitas. Salah satu solusinya adalah dengan menerapkan pertanian perkotaan atau disebut juga pertanian urban (Noorsya dan Kustiwan, 2013).

Pertanian urban kini banyak digeluti masyarakat yang tinggal di daerah perkotaan. Pada praktiknya pertanian urban di kota kota besar saat ini mengarah pada pembangunan pertanian yang mempunyai nilai manfaat lebih luas untuk psikologi dan lingkungan. Kebutuhan pangan kota Bandung yang dipasok oleh daerah luar kota Bandung seperti Kabupaten hingga luar Provinsi mencapai 97%. Hal ini tentu menjadi masalah serius ketika terbatasnya ketersediaan dan akses pangan tidak terpenuhi (Puriandi dan Indrajati, 2013).

Masalah terbatasnya ketersediaan pangan yang ada dikarenakan kurang mendukungnya kebutuhan informasi dan alat bantu di lingkungan masyarakat perkotaan dalam melaksanakan pertanian urban menjadi salah satu alasan dalam penelitian ini membuat solusi yang bisa menyelesaikan permasalahan masyarakat dalam melaksanakan pertanian urban sehingga masyarakat bisa memenuhi kebutuhan pangan secara mandiri, dan membantu memberi informasi terhadap masyarakat mengenai *urban farming*.

Pemanfaatan teknologi yang terus berkembang khususnya kecerdasan buatan dan juga sistem intelijensi visual dapat digunakan untuk mendukung *urban farming*. Kecerdasan buatan atau disebut juga *Artificial Intelligence* (AI) menyebar luas menjadi lebih kuat, AI banyak digunakan hingga meningkatnya jumlah dari produk *digital* baru, publik servis, militer dan bahkan aplikasi manusia sehari hari sudah menjadi hal yang tidak terbantahkan lagi (Europa,eu, 2018). *Machine Learning* dikolaborasikan dengan sistem intelijensi visual menjadi aplikasi yang dapat menganalisis permasalahan suatu tanaman. Metode yang penulis gunakan pada sistem intelijensi visual aplikasi ini adalah metode deteksi tepi (*edge detector*) untuk pencarian detector, yaitu deskripsi dari fitur visual suatu citra yang menerangkan suatu aspek pembeda suatu citra dengan citra yang lain, bisa dari histogram (arah *gradien* warna), menggunakan *clustering* yang sama, menghitung jarak histogram, lalu menghasilkan *keypoint descriptor*.

Metode pada *Machine Learning* dalam aplikasi ini menggunakan metode *Unsupervised Learning*. Penelitian ini bertujuan untuk mengkolaborasikan metode *Edge Detector* dengan *Unsupervised Learning* untuk mengetahui seberapa efektif

hasil yang diperoleh untuk mendeteksi suatu objek berupa tanaman. Penelitian ini diharapkan dapat membantu masyarakat dalam merawat dan mengetahui kendalanya dalam bercocok tanam pada pertanian urban.

1.2 Identifikasi Masalah

Sesuai dengan latar belakang yang telah diuraikan sebelumnya, maka identifikasi masalah dari penelitian ini adalah:

1. Kurang dan sulitnya memperoleh informasi mengenai kebutuhan dan alat yang digunakan secara efektif oleh masyarakat dalam melakukan *urban farming*.
2. Bagaimana caranya agar user dapat memperoleh hasil yang efektif dalam mendeteksi masa pertumbuhan tanaman dengan menggunakan sistem inteljensi visual dan *Machine Learning*?

1.3 Tujuan Penelitian

Sesuai dengan identifikasi masalah yang telah di uraikan sebelumnya, maka tujuan penelitian ini adalah:

1. Membantu *user* mendapatkan informasi mengenai pertanian urban dan memberikan solusi cara tanam yang efektif dengan menerapkan Sistem Intelijensi Visual dan *Machine Learning*.
2. Menggunakan metode *edge detector* yang mempunyai kelebihan dalam mendeteksi objek khususnya dalam pendeteksian warna sehingga memberikan hasil akurasi yang lebih optimal.

1.4 Batasan Masalah

Di dalam melakukan suatu penelitian di perlukan adanya batasan suatu masalah agar penelitian tersebut lebih terarah dan memudahkan dalam pembahasan sehingga tujuan penelitian akan tercapai. Beberapa batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penelitian ini tidak dilakukan pada masa vegetatif tanaman, dimana masa vegetatif merupakan masa dimana tanaman sudah menjadi pohon seutuhnya.
2. Program hanya dapat menganalisis tanaman pada masa generatif, yaitu saat tanaman baru memiliki batang dan tunas daun pertama.
3. Program hanya bisa mendeteksi objek tanaman apabila sudah memiliki bentuk tunas daun dan batang.
4. Tahapan *operation* & *maintenance* tidak dilakukan dalam penelitian ini.

1.5 Metode Penelitian

1.5.1 Metode Pengumpulan Data

Adapun metode pengumpulan data yang dilakukan untuk mendapatkan data data dan referensi yang dibutuhkan dalam menyusun penelitian ini, meliputi:

1. Studi Kepustakaan

Dilakukan dengan cara membaca dan mempelajari buku-buku, referensi yang mendukung dengan topik dengan menghimpun informasi yang relevan pada topik atau masalah yang menjadi obyek penelitian. Informasi tersebut dapat

diperoleh dari buku-buku, karya ilmiah, tesis, disertasi, ensiklopedia, internet, dan sumber-sumber lain yang akan dibahas dalam penyusunan skripsi ini.

2. Studi Lapangan

a. Observasi

Melakukan observasi ke lahan pertanian untuk mendapapatkan data-data yang diperlukan.

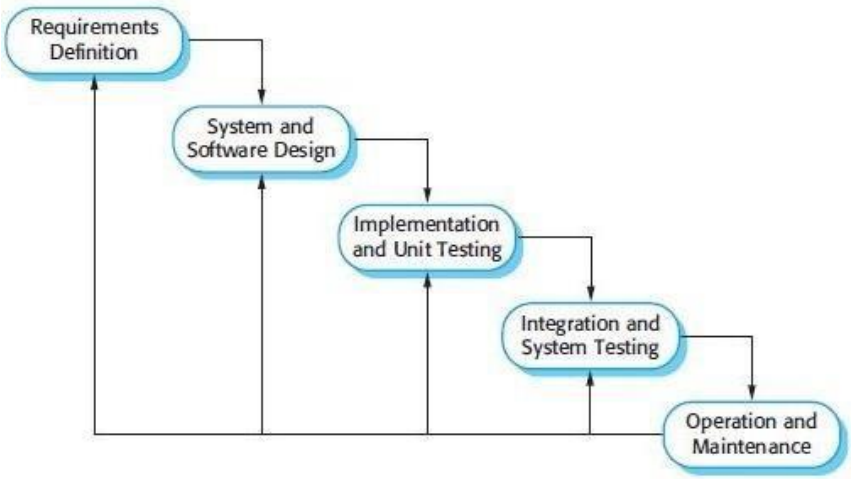
b. Wawancara

Melakukan wawancara kepada pihak petani untuk mendapatkan data yang diperlukan dalam penelitian.

1.5.2 Metode Pengembangan Sistem

Dalam penelitian ini peneliti menggunakan metode penelitian kuantitatif. Penelitian kuantitatif digunakan dalam meneliti status kelompok manusia, suatu 6 kondisi, suatu sistem pemikiran atau kelas peristiwa pada waktu tertentu. Sehingga melalui metode ini akan diperoleh data dan informasi tentang gambaran suatu fenomena, fakta, sifat serta hubungan fenomena tertentu secara komperehensif dan integral (Silalahi, 2015: 5). Dengan demikian pengulangan dalam penelitian kuantitatif dilakukan dalam rangka mendapatkan konsistensi atau reliabilitas data penelitian yang ada.

Kerangka berpikir dalam penelitian ini menggunakan metode pengembangan (SDLC) dengan Model *Waterfall*. Model ini melakukan pendekatan secara sistematis dan berurutan. Metode Waterfall mempunyai tahapan-tahapan yang digambarkan pada Gambar 1.1



Gambar 1.1 Ilustrasi model waterfall (Rosa, 2016)

Berikut ini adalah penjelasan dari tahapan-tahapan yang dilakukan dalam Model *Waterfall:*

1. *Communication*

Layanan sistem, kendala, dan tujuan titetapkan oleh hasil konsultasi dengan pengguna dan wawancara dengan ahli pada bidang ini,agar bisa lebih mudah memahami dan mencapai tujuan yang ingin dicapai. Hasil dari komunikasi ini akan dijadikan bahan analisis permasalahan yang dihadapi lalu mengumpulkan

data data yang diperlukan sehingga fitur dan fungsi aplikasi bisa terdefinisikan dengan baik.

2. *Planning*

Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan. Perancangan perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

3. *Modelling*

Tahap ini merancang dan membuat model arsitektur sistem yang berfokus pada perancangan struktur data, asitektur software, desain *Unifies Modelling Language* (UML), tampilan interface, dan algoritma program. Tujuannya adalah agar bisa memahami gambaran besar dari program yang akan dibuat.

4. *Construction*

Tahapan ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadu untuk diperbaiki nantinya.

5. *Operation and maintenance*

Tahapan ini tidak dilakukan dalam penelitian ini.

1.6 Sistematika Penulisan

Sistematika penulisan skripsi ini disusun untuk memberikan gambaran umum tentang penelitian yang dijalankan. Sistematika penulisan skripsi ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menguraikan tentang latar belakang permasalahan, mencoba merumuskan inti permasalahan yang dihadapi, menentukan tujuan dan kegunaan penelitian, yang kemudian diikuti dengan pembatasan masalah, asumsi, serta sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini membahas berbagai konsep dasar dan teori-teori yang berkaitan dengan topik penelitian yang dilakukan dan hal-hal yang berguna dalam proses analisis permasalahan serta tinjauan terhadap penelitian-penelitian serupa yang telah pernah dilakukan sebelumnya termasuk sintesisnya. Terdapat landasan teori yang mendasari pembangunan aplikasi seperti pengertian android, informasi, sistem informasi pemodelan data, analisis sistem, kamus data, pengolahan data, dan tools – tools yang akan digunakan.

BAB III ANALISA MASALAH DAN PERANCANGAN PROGRAM

Bab ini membahas tentang analisis kebutuhan dalam membangun perangkat lunak, analisis sistem yang akan berjalan sesuai dengan metode pembangunan perangkat lunak yang digunakan. Terdapat pula perancangan antarmuka untuk aplikasi yang akan dibangun sesuai dengan hasil analisi yang tela dibuat. Model

dalam perancangan yang akan digunakan adalah pemodelan terstruktur. Bab ini juga membahas hasil implementasi dari hasil analisis dan perancangan yang telah dibuat disertai juga dengan hasi pengujian sehingga diketahui apakah sistem yang dibangun sudah memenuhi syarat sebagai aplikasi yang mudah digunakan.

BAB IV IMPLEMENTASI DAN UJI COBA

Bab ini berisi tentang penjelasan dalam pengoperasian aplikasi secara bertahap dan juga evaluasi dari implementasi aplikasi ini, termasuk kelebihan dan kekurangannya menggunakan metode kuantitatif.

BAB V PENUTUP

Bab ini berisi kesimpulan dan saran yang diperoleh dari hasil penelitian skripsi yang telah dibuat. Kesimpulan didapatkan dari penelitian yang telah dilakukan sampai pembuatan laporan serta saran agar aplikasi yang penulis buat menjadi lebih baik lagi untuk pengembangan selanjutnya.

BAB 2

LANDASAN TEORI

2.1 Machine Learning

Menurut Sergios Theodoridis (2015:17) *Machine Learning* merupakan nama yang memperoleh popularitas sebagai payung untuk metode yang sudah dipelajari dan dikembangkan selama beberapa dekade terakhir di berbagai bidang keilmuan dan juga dengan nama nama yang berbeda, seperti, *Statistical Learning*, *Statistical Signal Processing*, *Pattern Recognition*, *Adaptive Signal Processing*, *Image processing* & *Analysis*, *System Identification & Control*, *Data Mining & Information*, *Retrieval*, *Computer Vision*, dan *Computational Learning*. Nama "*Machine Learning"* mengindikasian apa saja kesamaan pada bidang kedisiplinan yang tadi disebutkan, yang "belajar dari data", dan membuat prediksi. Apa yang dicoba dipelajari dari suatu data dengan struktur yang mendasari dan ditetapkan, melalui pengembangan sebuah model, yang nantinya akan digunakan untuk memberikan sebuah prediksi.

2.1.1 Klasifikasi

Tujuan adanya klasifikasi adalah untuk menetapkan suatu *pola* yang tidak diketahui kedalam beberapa *class* yang sudah diketahui. Sebagai contoh, dalam *X-Ray* mammografi, kita diberikan sebuah gambar daerah yang mengindikasikan keberadaan tumor. Tujuan dari sistem diagnonsis *computer-aided* (CAD) adalah untuk memprediksi apakah tumor ini sama dengan klas *benign* atau klas

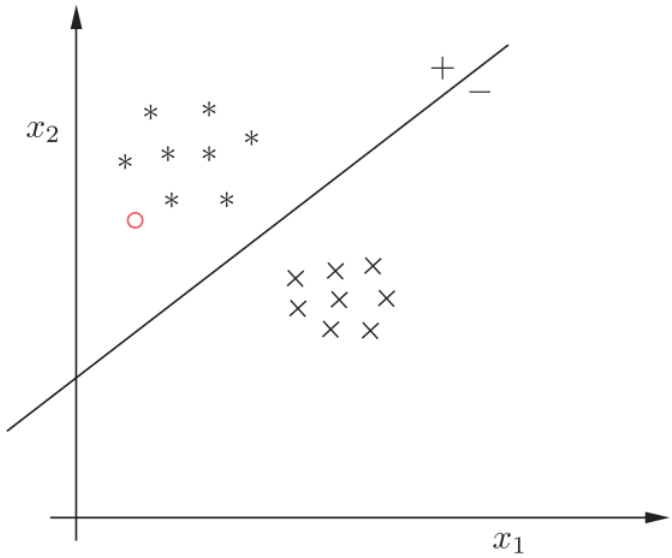
malignant. Sistem *Optical character recognition* (OCR) juga dibuat berdasarkan sistem klasifikasi, dimana kesamaan gambar terhadap lembar alfabet harus bisa dikenali dan ditetapkan pada satu dari 24 (untuk alfabet latin) klas.

Langkah pertama dalam mendesain setiap tugas *machine learning* adalah untuk menentukan bagaimana caranya merepresentasikan setiap pola dalam komputer. Hal ini didapatkan selama tahap sebelum proses, salah satu harus "mengkode" informasi bersangkutan yang terletak didalam data raw (*pixel* gambar atau karakter huruf dalam contoh sebelumnya) secara efisien dan cara yang informatif. Ini biasanya berlaku dengan mentransformasikan data raw didalam ruang baru dengan setiap pola yang direpresentasikan oleh vektor, $x \in \mathrm{R}^l$. Ini dikenal sebagai fitur vektor, dan elemen l ini diketahui sebagai fitur. Dengan ini, setiap pola menjadi satu poin dalam ruang l-dimensi, diketahui sebagai ruang fitur atau ruang input. Kita melihat ini sebagai tahap generasi fitur.

Setelah menentukan ruang input, dimana data direpresentasikan, salah satunya harus merangkai sebuah *classifier*. Berdasarkan *training data*, salah satunya dibuat sebagai *function*, *f,* yang memprediksi label *output* yang telah memperoleh *input*, hal ini memberikan pertimbangan yang baik sebagai nilai dari sebuah fitur. Fungsi ini diketahui sebagai *classifier*. Secara garis besar kita harus mendesain suatu rangkaian fungsi.

Setelah *classifier* di desain, sistem siap untuk memprediksi. Memperoleh suatu pola yang tidak diketahui, buat sebuah kesamaan fitur vektor, x, dari sebuah raw data, dan kita memasukan nilai ini terhadap *classifier*, tergantung dari sebuah nilai f(x) polanya di klasifikasi kedalam satu dari 2 klas. Pada mulanya kita

diberikan rangkaian poin, setiap poin merepresentasikan pola didalam ruang 2 dimensi (2 fitur digunakan, *xi*, *x2*). *Stars* termasuk dari 1 klas, *w1* misalkan dan yang bersilangan dengannya, *w2*, didalam tugas 2 klas klasifikasi. Ini semua adalah poin *training*. Berdasarkan semua poin ini, *classifier* telah dipelajari dan didapatkan, untuk kasus paling sederhana, ini ternyata sebuah fungsi linear.



Gambar 1.0 : Classifier, kasus simple fungsi linear (**Sergios Theodoridis**, 2015).

Tipe *machine learning* yang sudah dijelaskan sebelumnya ini diketahui sebagai *supervised learning*, dimana serangkaian *training data* dengan label yang diketahui sudah tersedia. Sebuah *training data* bisa dilihat sebagai data dari pengalaman sebelumnya, ini membuat model untuk prediksi kedepannya

Tahapan selanjutnya adalah regresi, dimana regresi membagikan besarnya tingkat fitur generasi/tahap seleksi, seperti dijelaskan sebelumnya, hanya saja, sekarang masuk bagian variabel *output*, y, bukan diskrit tetapi mengambil nilai

sebuah interval kedalam *real axis* atau dalam banyaknya daerah komples beberapa bidang. Ketika sebuah fungsi ditemukan, ketika poin yang tidak diketahui ada, sistem bisa memprediksi nilai *output*.

2.1.2 Jenis Jenis Machine Learning

Menimbang sebuah mesin (atau organisme hidup) yang menerima beberapa rangkaian dari input *x1,x2,x3,....*, dimana x_t adalah sensor input pada waktu *t*. Input ini, dimana hal yang biasa kita sebut *data*, bisa sesuai dengan gambar yang tergambarkan pada retina, *pixel* pada sebuah kamera, atau gelombang suara, semua itu bisa sama dengan kurang jelasnya data sensor. Sebagai contoh sebuah kata dalam sebuah berita, atau urutan benda didalam kantung pada sebuah supermarket.

Suatu jenis *machine learning* bisa dibedakan menjadi 4 jenis *machine learning* yang berbeda.

Didalam *supervised learning*, mesin diberikan rangkaian output yang diinginkan *y1,y2,....*, dan tujuan dari mesin adalah untuk memproduksi*output* yang sesuai setelah diberikan *input* baru. *Output* ini bisa juga disebut label klas (dalam klasifikasi) atau angka asli (dalam regresi).

Dalam *reinforcement learning* sebuah mesin berinteraksi pada keadaan sekitar dengan membuat aksi *a1,a2,....* Aksi aksi ini mempengaruhi kondisi keadaan sekitar, dimana hal ini membuat hasil dari sistem menerima sebuah ganjaran terukur/skalar (atau hukuman) *r1,r2,....* Tujuan dari mesin adalah untuk mempelajari sebuah aksi dengan cara memaksimalkan ganjaran yang akan datang yang diterima (atau memperkecil hukuman sekecil mungkin) dalam masa

pelaksanaanya. *Reinforcement learning* berhubungan dekat dengan bidang teori keputusan / *decision theory* (dalam statistik dan manajemen sains), dan *control theory* (dalam *engginering*). Masalah fundamental yang mempelajari bidang bidang berikut selalu setara secara formal, dan memiliki solusi yang sama, meskipun aspek masalahnya berbeda dan solusinya sering ditegaskan.

Jenis ketiga dari *machine learning* berhubungan dekat dengan *game theory* dan *renforcemenr learning* yang tergeneralisasi. Disini juga sistem memperoleh *input*, membuat aksi, dan memperoleh hasil. Hanya saja, keadaan sekitar yang berinteraksi dengan sistem bukan termasuk dunia statis, melainkan memiliki sistem lain yang merasakan, melakukan aksi, mendapatkan hasil, dan mempelajarinya. Pada akhirnya tujuan dari sistem adalah untuk melakukan aksi untuk memaksimalkan hasil dalam keterkaitannya dengan sistem lain, aksi masa sekarang dan yang akan datang. Semua ini mempunyai peran yang besar didalam *game theory* untuk sitem yang simpel, kasus dinamic dengan berbagai sistem yang beradaptasi tetap aktif dan menentang area yang dipelajari.

Yang terakhir, didalam *Unsupervised Learning* , sistem secara simpel menerima input *x1*,*x2*,.... tapi tidak mendapatkan target *supervised output*, tidak juga hasil dari lingkungannya. Mungkin *unsupervised learning* terlihat misterius untuk membayangkan bagaimana sistem bisa mempelajari sesuatu yang diberikan, tetapi sesuatu itu tidak memberikan suatu balasan apapun dari lingkungannya. Akan tetapi semua itu memiliki kemungkinan untuk mengembangkan *formal framework* untuk *unsupervised learning* berdasarkan gagasan dari sistem yang dibuat adalah untuk membuat representasi dari input yang bisa digunakan untuk

pembuat keputusan, memprediksi *input* yang akan datang, mengkomunikasikan *input* secara efisien terhadap sistem lain, dll. Dalam arti tertentu, *Unsupervised learning* bisa di kategorikan sebagai penemuan pola dalam data yang tersedia, dan melebihi apa yang akan dipertimbangkan sebagai *noise unstructured* yang asli. 2 contoh yang sangat simpel dari sebuah *unsupervised leraning* adalah *clusstering* dan *dimensionalityi reduction*.

2.2 Android

Android menurut Dian (2016:47) adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware, dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Android adalah sistem operasi yang menghidupkan lebih dari satu miliar smartphone dan tablet. Karena perangkat ini membuat hidup kita begitu manis, maka setiap versi Android dinamai dari makanan penutup (dessert).

Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya Open Handset Alliance, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel Android pertama mulai dijual pada bulan Oktober 2008.

Antarmuka pengguna Android umumnya berupa manipulasi langsung, menggunakan gerakan sentuh yang serupa dengan tindakan nyata, misalnya

menggeser, mengetuk, dan mencubit untuk memanipulasi objek di layar, serta papan ketik virtual untuk menulis teks. Selain perangkat layar sentuh, Google juga telah mengembangkan Android TV untuk televisi, Android Auto untuk mobil, dan Android Wear untuk jam tangan, masing-masingnya memiliki antarmuka pengguna yang berbeda. Varian Android juga digunakan pada Laptop, konsol permainan, kamera digital, dan peralatan elektronik lainnya.

Android adalah sistem operasi dengan sumber terbuka, dan Google merilis kodenya di bawah Lisensi Apache. Kode dengan sumber terbuka dan lisensi perizinan pada Android memungkinkan perangkat lunak untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel, dan pengembang aplikasi. Selain itu, Android memiliki sejumlah besar komunitas pengembang aplikasi (apps) yang memperluas fungsionalitas perangkat, umumnya ditulis dalam versi kustomisasi bahasa pemrograman Java.. Pada bulan

Oktober 2013, ada lebih dari satu juta aplikasi yang tersedia untuk Android, dan sekitar 50 miliar aplikasi telah diunduh dari Google Play, toko aplikasi utama Android. Sebuah survei pada bulan April-Mei 2013 menemukan bahwa Android adalah platform paling populer bagi para pengembang, digunakan oleh 71% pengembang aplikasi bergerak. Di Google I/O 2014, Google melaporkan terdapat lebih dari satu miliar pengguna aktif bulanan Android, meningkat dari 583 juta pada bulan Juni 2013.

Faktor-faktor di atas telah memberikan kontribusi terhadap perkembangan Android, menjadikannya sebagai sistem operasi telepon pintar yang paling banyak digunakan di dunia, mengalahkan Symbian pada tahun 2010. Android juga

menjadi pilihan bagi perusahaan teknologi yang menginginkan sistem operasi berbiaya rendah, bisa dikustomisasi, dan ringan untuk perangkat berteknologi tinggi tanpa harus mengembangkannya dari awal. Sifat Android yang terbuka juga telah mendorong munculnya sejumlah besar komunitas pengembang aplikasi untuk menggunakan kode sumber terbuka sebagai dasar proyek pembuatan aplikasi, dengan menambahkan fitur-fitur baru bagi pengguna tingkat lanjut atau mengoperasikan Android pada perangkat yang secara resmi dirilis dengan menggunakan sistem operasi lain.

Pada November 2013, Android menguasai pangsa pasar telepon pintar global, yang dipimpin oleh produk-produk Samsung, dengan persentase 64% pada bulan Maret 2013. Pada Juli 2013, terdapat 11.868 perangkat Android berbeda dengan beragam versi. Keberhasilan sistem operasi ini juga menjadikannya sebagai target ligitasi paten "perang telepon pintar" antar perusahaan-perusahaan teknologi. Hingga bulan Mei 2013, total 900 juta perangkat Android telah diaktifkan di seluruh dunia, dan 48 miliar aplikasi telah dipasang dari Google Play.

2.3 Hubungan Android dengan Aplikasi

Android memungkinkan penggunanya untuk memasang aplikasi pihak ketiga, baik yang diperoleh dari toko aplikasi seperti Google Play, Amazon Appstore, ataupun dengan mengunduh dan memasang berkas APK dari situs pihak ketiga. Di Google Play, pengguna bisa menjelajah, mengunduh, dan memperbarui aplikasi yang diterbitkan oleh Google dan pengembang pihak ketiga, sesuai dengan persyaratan kompatibilitas Google. Google Play akan

menyaring daftar aplikasi yang tersedia berdasarkan kompatibilitasnya dengan perangkat pengguna, dan pengembang dapat membatasi aplikasi ciptaan mereka bagi operator atau negara tertentu untuk alasan bisnis. Pembelian aplikasi yang tidak sesuai dengan keinginan pengguna dapat dikembalikan dalam waktu 15 menit setelah pengunduhan. Beberapa operator seluler juga menawarkan tagihan langsung untuk pembelian aplikasi di Google Play dengan cara menambahkan harga pembelian aplikasi pada tagihan bulanan pengguna. Pada bulan September 2012, ada lebih dari 675.000 aplikasi yang tersedia untuk Android, dan perkiraan jumlah aplikasi yang diunduh dari Play Store adalah 25 miliar.

Aplikasi Android dikembangkan dalam bahasa pemrograman Java dengan menggunakan kit pengembangan perangkat lunak Android (SDK). SDK ini terdiri dari seperangkat perkakas pengembangan, termasuk debugger, perpustakaan perangkat lunak, emulator handset yang berbasis QEMU, dokumentasi, kode sampel, dan tutorial. Didukung secara resmi oleh lingkungan pengembangan terpadu (IDE) Eclipse, yang menggunakan plugin Android Development Tools (ADT). Perkakas pengembangan lain yang tersedia di antaranya adalah Native Development Kit untuk aplikasi atau ekstensi dalam C atau C++, Google App Inventor, lingkungan visual untuk pemrogram pemula, dan berbagai kerangka kerja aplikasi web seluler lintas platform.

Dalam rangka menghadapi penyensoran Internet di Republik Rakyat Tiongkok, perangkat Android yang dijual di RRT umumnya disesuaikan dengan layanan yang disetujui oleh negara.

2.4 Aplikasi

Menurut Andi Juansyah (2015:2) Secara pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju. Menurut kamus computer eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang di harapkan.

Pengertian aplikasi menurut Kamus Besar Bahasa Indonesia, "Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu".

Aplikasi adalah suatu subkelas dari suatu perangkat lunak komputer yang memanfaatkan kemampuan komputer secara langsung untuk melakukan suatu tugas yang diinginkan pengguna (Wikipedia, 2012). Aplikasi dapat juga dikatakan sebagai penerjemah perintah-perintah yang dijalankan pengguna komputer untuk diteruskan ke atau diproses oleh perangkat keras. Aplikasi merupakan program yang secara langsung dapat melakukan proses-proses yang digunakan dalam komputer oleh pengguna. Aplikasi merupakan kumpulan dari file-file tertentu yang berisi kode program yang menghubungkan antara pengguna dan perangkat keras Komputer.

Aplikasi sering juga disebut sebagai perangkat lunak, merupakan program komputer yang isi instruksinya dapat diubah dengan mudah. Aplikasi pada

umumnya digunakan untuk mengontrol perangkat keras (yang sering disebut sebagai device driver), melakukan proses perhitungan, dan berinteraksi dengan aplikasi yang lebih mendasar lainnya (seperti sistem operasi, dan bahasa pemrograman). Secara umum aplikasi dapat dibagi menjadi 3 tingkatan yaitu tingkatan program aplikasi (application program misalnya Microsoft Office), tingkatan sistem operasi (operating system misalnya Microsoft Windows), dan tingkatan bahasa pemrograman (misalnya PHP).

Beberapa aplikasi telah digabung menjadi suatu paket aplikasi dan sering disebut sebagai suite aplikasi (application suite). Contohnya adalah Microsoft Office dan OpenOffice.org, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

Aplikasi atau perangkat lunak dapat digolongkan menjadi beberapa kelas, antara lain:

1. Perangkat lunak perusahaan (enterprise)
2. Perangkat lunak infrastruktur perusahaan
3. Perangkat lunak informasi kerja

4. Perangkat lunak media dan hiburan
5. Perangkat lunak pendidikan
6. Perangkat lunak pengembangan media
7. Perangkat lunak rekayasa produk

2.5 Sistem Intelijensi Visual

Menurut Bloei Zhou (2018:24) Sistem intelijensi visual merupakan model kecerdasan buatan yang berusaha menyamai tingkat pengenalan visual manusia, bisa mempelajari, mengidentifikasi, dan menguraikan struktur yang mendasari faktor penjelasan pada *input* data yang diamati.

Memahami dunia visual dalam satu tatapan adalah salah satu prestasi kecerdasan manusia yang paling menakjubkan. Dengan mengambil sampel adegan puluhan kali per detik, kami terpapar jutaan gambar alami dalam setahun. Pengalaman visual yang kaya dan beragam yang terkandung dalam konteks adegan ini memandu penafsiran kita tentang dunia dan membentuk pengetahuan kita tentang realitas. Terinspirasi oleh ini, model pembelajaran mendalam baru-baru ini membangun representasi internal mereka dalam pendekatan pembelajaran berbasis data dari jutaan sampel pelatihan dari dataset adegan skala besar. Digeneralisasi dari jutaan sampel pelatihan, banyak model pengenalan visual yang didukung oleh jaringan saraf yang dalam telah mencapai kinerja tingkat manusia.

Model AI mencapai pengakuan visual tingkat manusia, sehingga terdapat kemungkinan besar sistem dapat belajar mengidentifikasi dan menguraikan faktor-faktor penjelas yang mendasari data input yang diamati. Pertanyaan utama

tentang pembelajaran representasi yang dapat ditafsirkan adalah apa saja faktor penjelas yang mendasari ketika sistem dilatih untuk mengenali objek atau adegan, dan lebih jauh lagi, apakah faktor-faktor ini bermakna dan dapat ditafsirkan bagi manusia atau tidak.

Dalam pelaksanaanya, sistem intelijensi visual sangatlah erat dengan visi komputer.

2.6 Visi Komputer (*Computer Vision*)

Visi Komputer merupakan sub disiplin ilmu dari kecerdasan buatan yang mempelajari bagaimana mesin dapat mengenali objek yang diamati atau diobservas. Dapat juga disebut sebagai machine vision, ilmu ini mengembangkan teori-teori dan algoritma dimana informasi yang berguna diekstraksi dan dianalisis dari sebuah citra penelitian, sekumpulan citra, atau citra yang berurutan dari sebuah komputasi yang dibuat oleh computer.

Visi komputer dapat juga diartikan sebagai berikut :

1. Otomatis dan integrasi sebuah range yang luas yang terdiri dari proses-proses dan representasi-representasi terhadap persepsi visual.
2. visi komputer berhubungan dengan perolehan gambar, pemrosesan, klasifikasi, pengenalan, dan menjadi penggabungan, pengurutan pembuatan keputusan menuju pengenalan.
3. visi komputer sesuai dengan sifatnya, merupakan suatu subyek yang merangkul berbagai disiplin tradisional secara luas guna mendasari

prinsip-prinsip formalnya, dan dalam mengembangkan suatu metodologi yang berlainan dari apa yang dimilikinya, pertama-tama harus mengembangkan dan secara berurutan membangun materi yang mendasari ini.

4. Visi komputer adalah suatu bidang yang bertujuan untuk membuat suatu keputusan yang berguna mengenai objek fisik nyata dan keadaan berdasarkan atas sebuah citra. Visi komputer merupakan kombinasi antara pengolahan citra dan pengenalan pola. Hasil keluaran dari proses visi komputer adalah pengertian tentang citra.Boyle dan Thomas (C & Boyle, 1988), mengatakan bahwa computer vision lebih daripada pengenalan, computer vision melakukan operasi “low level processing” sebagai algoritma image processing yang murni. Mereka juga yang menggolongkan image processing ke dalam computer vision.

2.7 Pengolahan Citra

Pengolahan citra merupakan bidang studi yang mempelajari proses pengolahan gambar dimana baik masukan maupun keluarannya berbentuk berkas citra digital. Juga dijelaskan bahwa pengolahan citra merupakan pengolahan dan analisis citra yang banyak melibatkan persepsi visual.

Kebutuhan untuk memproses sebuah gambar dengan cep at dalam satu ap likasi merupakan salah satu masalah utama dalam melakukan pengolahan citra. Sedangkan untuk ap likasi yang berjalan secara real time lebih bergantung pada pemrosesan piksel atau signal yang cepat daripada metode optimasi lain yang rumit dan memakan waktu.

Niblack (Niblack, 1986) menjelaskan image processing sebagai pemrosesan dari citra-citra, kemudian menambahkan bahwa output dari image processing akan juga menjadi sebuah image.

2.8 Citra Digital

Ada beberapa definisi citra digital, yaitu :

1. Suatu representasi, kemiripan, atau imitasi dari suatu objek atau benda (menurut kamus webster).
2. Gambar dua dimensi yang dihasilkan dari gambar analog dua dimensi yang kontinu menjadi gambar diskrit melalui proses sampling. Gambar analog adalah representasi visual dari suatu objek menggunakan kamera an alog yang secara matematika dapat direpresentasikan sebagai rentang nilai yang menunjukkan letak (posisi) dan intensitas. Gambar analog dibagi menjadi N baris dan M kolom.
3. sehingga menjadi gambar diskrit. Persilangan antara baris dan kolom tertentu disebut dengan piksel. Sampling pada penjelasan di atas ada proses untuk menentukan warna pada piksel tertentu pada citra dari sebuah gambar yang kontinu. Pada proses sampling biasanya dicari warna rata-rata dari gambar analog yang kemudian dibulatkan. Proses sampling sering disebut juga dengan proses digitisasi.
4. Representasi dari suatu objek nyata baik dalam bentuk dua dimensi maupun tiga dimensi menjadi bentuk gambar digital yang dikenali oleh komputer.

5. Fungsi intensitas warna dua dimensi f(x,y) dimana x dan y mewakili koordinat lokasi suatu titik dan nilai dari fungsi yang merupakan tingkat intensitas warna atau tingkat keabu-abuan dari titik tersebut.

2.9 Pengenalan Wajah (*Face Recognition*)

Sistem pengenalan wajah merupakan subjek yang menarik untuk diteliti sejak lama. Hal ini dikarenakan pengenalan wajah merupakan cara yang utama bagi manusia untuk mengenali dan mengidentifikasi seseorang. Setiap orang memiliki ciri wajah yang unik dan berbeda. Dengan melihat dari wajah, seseorang dapat dibedakan dengan orang yang lainnya.

Tehnik pengenalan wajah telah berkembang jauh sebelum komputer ada. Francis Galton, pada tahun 1888, mengemukakan metode dengan menggunakan pendekatan geometris. Dalam proposalnya, Galton menitikberatkan perhatian pada fitur-fitur penting dari wajah, yang disebut keys points. Keys points ini meliputi alis, mata, hidung, bibir dan sebagainya. Dengan menghitung jarak antara keys points dapat diperoleh ciri wajah seseorang. Akan tetapi tehnik yang dikemukakan oleh Galton sangat sensitif terhadap perubahan wajah, sehingga akan mengalami kegagalan ketika ada penambahan atribut pada wajah.

Dalam usaha ingin meniru kemampuan manusia dalam mengenali dan mengidentifikasikan seseorang, para peneliti dalam bidang Computer Vision pun melakukan penelitian. Hasilnya metode-metode dalam sistem pengenalan wajah ini dikelompokkan menjadi dua kategori pendekatan (Chelappa et al 1955) yaitu : feature- based dan holistic. Sedangkan menurut Quintiliano dan Rosa (Quintiliano

& Rosa, 2006) dibagi menjadi tiga, yaitu : template-based, feature-based dan appearance-based.

Pendekatan template-based sendiri mempunyai dua versi, yang pertama (yang paling sederhan a) menggamb arkan citra wajah sebagai matriks bidimensional yang nilainya mewakili edges dari tingkat kelonjongan wajah dan factor dari wajah. Yang kedua, lebih lengkap, menggambarkan citra wajah sebagai multip le templates yang dimabil dari berbagai sudut dan arah pandang. Kelebihan dari teknik ini adalah kesederhanaannya meskipun menggunakan memori yang cukup besar dan algoritma yang tidak efisien. (Quintiliano & Rosa, 2006).

Pendekatan dengan feature-based pada dasarnya tergantung pendeteksian dan pengkarakterisasian dari feature wajah seseorang serta hubungan geometriknya. Pada umumnya feature yang dimaksud adalah mata, mulut dan hidung. Sedangkan pendekatan holistic lebih melibatkan keseluruhan citra wajah dengan melakukan proses encoding dan memberlakukan "kode" wajah dari hasil encoding tersebut sebagai sebuah titik dalam ruang berdimensi tinggi.

Pada pendekatan feature-based, proses awal pada automatisasi proses pengenalan wajah lebih penting. Proses awal ini melibatkan penggunaan teknik pengolahan citra yang sederhana, seperti pendeteksian sisi dan lain-lain dengan tujuan mendeteksi wajah dan feature-featurenya. Sebagai contoh: tehnik dari Sakai pada tahun 1969 dimana pertama-tama peta sisi diekstrak dari sebuah citra masukan dan kemudian dicocokkan pada template oval yang besar, dengan variasi-variasi posisi dan ukuran yang mungkin. Adanya wajah atau tidak

dikonfirmasi dengan mencari sisi-sisi pada lokasi yang diperkirakan dari feature-feature seperti mata dan mulut.

Pada tahun 1970, Kelly menggunakan sebuah pendeteksi sisi untuk mengekstrak outline wajah manusia dari berbagai macam latar belakang. Govindaraju, Srihari dan Sher pada tahun 1990 memperkenalkan sebuah teknik dimana meletakkan wajah pada citra kacau yang menggunakan template yang dapat diuraikan. Teknik ini juga mirip dengan penelitian yang dilakukan oleh Yuille, Cohen dan Hallinan pada tahun 1989.

Pendekatan ini menunjukkan kinerja yang baik ketika diuji pada sebuah set data yang kecil, tetapi terkadang terdapat peningkatan pada pengenalan yang salah. Sedangkan teknik yang diperkenalkan oleh Kanade pada tahun 1973 serta Harmon dan Hunt pada tahun 1977 merupakan teknik dimana ketika citra wajah dimasukkan akan langsung melakukan perhitungan. Teknik feature-based menggunakan lebih sedikit penghitungan daripada templa te-based.

Sedangkan pendekatan holistic yang paling sukses adalah menggunakan Karhunen Loeve Transform (KLT). Teknik ini pertama kali diperkenalkan oleh Kirby dan Sirovich pada tahun 1990 yang dinamakan sebagai Principal Component Analysis (PCA). Transformasi ini menghasilkan perluasan dari sebuah citra masukan yang berhubungan dengan suatu set dari citra dasar atau dinamakan eigen images. Kemudian pada tahun 1991, Turk dan Pentland memperkenalkan teknik dimana hanya menggunakan beberapa koefisien KLT yang digunakan untuk mempresentasikan wajah yang dinamakan sebagai face space. Sistem bekerja sangat baik bagi citra yang diambil frontal. Tetapi sistem ini

tidak dapat mengatasi variasi pada orientasi wajah, posisi dan ilumin asi. Sehingga terdap at beberapa penelitian untuk mengatasi masalah ini, seperti pada tahun 1991 yang dilakukan Akamatsu, Sasaki, Fukamachi dan Suenega dimana beberapa proses ditambahkan untuk menstandarisasi citra wajah dalam posisi dan ukuran.

Pendekatan holistic lainnya adalah menggunakan Discrete Cosine Transform (DCT) sebagai proses feature extraction untuk klasifikasi selanjutnya DCT dihitung pada citra wajah kemudian hanya dipilih beberapa koefisien yang menjadi feature vector. Feature vector ini dapat dianggap sebagai representasi sebuah titik dalam ruang "wajah" berdimensi tinggi.

Secara umum proses pengenalan wajah dibagi menjadi 3, yaitu :

a. Pemisahan gambar dari gambar-gambar yang tidak teratur.

Pemisahan biasanya dengan menggunakan algoritma berikut ini. Sebuah sisi di construct, lalu sisi-sisi ini dihubungkan dengan beberapa heuristic, dan sisi-sisi disesuaikan ke dalam bentuk elips dengan menggunakan transformasi Hough. Hal tersebut dilakukan bila input diubah dari video images. Pemisahan dilakukan menggunakan gerakan (motion) sebagai isyarat.

b. Mengekstrak objek dari area wajah

Ada 2 tipe objek yang penting: objek holistic (dimana masing-masing objek adalah sebuah karakteristik dari seluruh wajah) dan objek partial membuat beberapa ukuran ke dalam beberapa point penting dari wajah, sedangkan tehnik

object holistic selalu menggunakan wajah secara keseluruhan. PCA adalah sebuah tehnik objek holistic.

c. Keputusan

Sebuah keputusan diambil dari data yang terkumpul pada tahap sebelumnya. Tiga tipe keputusan yang dapat diambil adalah:

1. Identifikasi, nama atau label individu yang didapat.

2. Pengen alan seseorang, keputusan dimana individu siap dilihat.

3. Pengkategorian, yang mana wajah harus diberikan dalam kategori.

Tehnik pengenalan wajah yang umum berdasarkan Jain et.al. (Jain, Halici, Hayashi, Lee, & Tsutsui, 1999) :

a. Hidden Markov Models (HMMs)

Samario & Harter pada tahun 1994 menggunakan konvensional Hidden Markov Models sebagai sebuah pendekatan grafis untuk mengubah informasi fitur dalam bentuk kode. Namun karena kompleksitas perhitungan yang amat tinggi, maka tehnik ini tidak cocok untuk digunakan dalam teknik real time walalupun tingkat akurasi yang tinggi dapat dicapai.

b. Eigenface

Turk & Pentland pada tahun 1991 menerapkan tehnik ini sebagai pengembangan dari PCA yang telah ada dengan menghitung eigenfaces untuk tiap citra dari data yang didapat baik dalam pelatihan dan pengujian.

c. Conventional Networks

Laurance, Giles, Tsai dan Back pada tahun 1997 menggunakan sebuah Self Organising Map (SOM) untuk mengurangi dimensi dari representasi input dan lima layer conventional networks untuk mengatasi masalah translasi dan perubahan citra wajah. Tehnik ini lebih cepat dari pendekatan Hidden Markov Models sebelumnya, namun masih memerlukan waktu pelatihan yang cukup lama.

d. Probabilistic Decision-Based Neural Networks

Tehnik ini diteliti oleh Lim Kung dan Lim pada tahun 1997, dimana teknik ini bertujuan untuk mengatasi masalah waktu pelatihan dan klasifikasi yang terdapat pada teknik conventional network.

e. Radial Basis Function Networks (RBF)

Hawell pada tahun 1997 menggunakan pendekatan dengan jaringan Radial Basis Function yang sangat cepat dalam pelatihan dan merupakan yang tercepat untuk klasifikasi dari semua teknik yang ada.

f. Continous µ-tuple Classifier

Tehnik ini dikembangkan oleh Lucas pada tahun 1997 yang mana merupakan pengembangan dari µ-tuple Classifier sebelumnya. Pengembangan ini dilakukan hanya untuk mengatasi masalah kecepatan dan efisiensi penyimpanan, namun belum mengatasi masalah yang terdapat dalam pengujian di dunia nyata pada pose dan pencahay aan citra.

g. Nearest Neighbor Classifier

Lucas pada tahun 1997 juga berhasil mencapai performa yang tinggi dengan I- Nearest Neighbor Classifier sederhana menggunakan pengukuran jarak City- Block.

2.10 Black Box Testing

Pengujian kotak hitam, juga disebut pengujian prilaku, berfokus pada persyaratan fugsional perangakat lunak. Artinya, teknik pengujian kotak hitam memungkinkan anda untuk membuat beberapa kumpulan kondisi masukan yang sepenuhnya akan melakukan semua kebutuhan fungsional untuk program. Pengujian kotak hitam bukan alternatif untuk pengujian kotak putih. Sebaliknya ini merupakan pelengkap yang mungkin dilakukan untuk mengungkap kelas kesalahan yang berbeda dari yang diungkap oleh metode whitebox testing.

Pengujian kotak hitam berupaya untuk menemukan kesalahan dalam kategori berikut:

1. fungsi yang salah atau hilang
2. kesalahan antarmuka
3. kesalahan dalam struktur data atau akses basis data eksternal
4. kesalahan perilaku atau kinerja
5. kesalahan inisialisasi dan penghentian.

Tidak seperti whitebox testing, yang dilakukan pada awal proses pengujian, pengujian kotak hitam cenderung diterapkan selama tahap-tahap pengujian selanjutnya. Karena pengujian kotak hitam sengaja mengabaikan struktur kendali, perhatian di fokuskan pada ranah informasi. Pengujian dirancang untuk menjawab pertanyan-pertanyaan berikut:

- Bagaimana validasi fungsional diuji?
- Bagaimana periraku dan kinerja sistem diuji?
- Kelas-kelas masukan apakah yang akan membentuk test case yang baik?
- Apakah sistem sangat sensitif pada nilai masukan tertentu?
- Bagaimana batas-batas kelas data diisolasi?
- Berapa kecepatan dan volume data yang dapat ditolerir oleh sistem?
- Apa pengaruh kombinasi spesifik data pada operasi sistem?

Dengan menerapkan teknik kotak hitam, ada mendapatkan serangkaian test case yang memenuhi kriteria berikut:

- test case yang mengurangi-dengan jumlah yang lebih besar dari satu-jumlah test case tambahan yang harus dirancang untuk mencapai pengujian yang wajar.
- test case yang mengatakan sesuatu tentang ada atau tidak adanya kelas kesalahan, dari pada kesalahan yang terkait hanya dengan pengujian khusus yang telah dibuat (Rosa, 2016).

2.11 *Teachable Machine Tensorflow*

Teachable Machine adalah alat GUI *(Graphic User Interface)* yang digunakan untuk membuat kecerdasan buatan model klasifikasi yang memudahkan masyarakat umum membuat aplikasi kecerdasan buatan sendiri tanpa perlu seseorang yang ahli di bidang *machine learning* / kecerdasan buatan. Kecerdasan buatan yang disediakan google ini menggunakan sistem *learning* untuk menganalisa data tanpa terprogram secara eksplisit. Tujuan dibuatnya adalah untuk memudahkan pelajar, guru, desainer dan bidang lainnya tentang kecerdasan buatan dengan membuat klasifikasi modelnya sendiri.

Beberapa kontribusi penting *teachable machine by google* di bidang pendidikan antara lain:

1. Bersifat fleksibel, interface yang mudah dipahami dalamTensorFlow juga ternyata dapat membantu penyelamatan hutan dengan memberi peringatan dini terhadap potensi aktivtas deforestasi ilegal. setiap pembelajaran algoritma rumit.

2. Kumpulan teknis dan desain keputusan yang bisa mengenalkan masa depan tentang alat kecerdasan buatan yang interaktif.

3. Merupakan pionir contoh tentang bagaimana terstrukturisasinya pembelajaran kecerdasan buatan dalam pengenalannya terhadap masyarakat luas yang mengakses dan mempelajari konsep komputerisasi kecerdasan buatan.

2.12 Tensorflow *Distributions*

Tensorflow adalah kerangka *machine learning* yang memudahkan user memperoleh data dan mempelajari kecerdasan buatan secara mendalam.

Tensorflow mengimplementasikan gambaran dari probabilitas teori yang di adaptasikan ke paradigma modern *deep-learning* dari *end-to-end* diferensiasi komputasi. Dibangun dari dua abtraksi, tensorflow menawarkan pembangunan fleksibel untuk probabilitas komputasi.

Distribusinya tersedia secara luas, cepat, dan megandung metode yang stabil untuk menciptakan *sample* dan statistika komputasi.

Kerangka kerja TensorFlow pun dikembangkan untuk memungkinkan para peneliti dan developer untuk mengerjakan model AI secara bersama sehingga memungkinkan banyak orang untuk menggunakannya.

TensorFlow sendiri pertama kali diperkenalkan pada akhir 2015. Pada tahun 2017 baru versi pertama dari TensorFlow muncul. TensorFlow bahkan telah digunakan untuk mencegah kebutaan. Caranya, dengan membantu dokter melakukan filter terhadap retinopati diabetes.

Manfaat dari tensorflow itu sendiri diantaranya :

1. Memberikan kemudahan berinteraksi seorang *user* seperti *developer* pada umumnya. Penulisan dan *debugging* dilakukan baris demi baris secara efisien sehingga *programmer* bisa membuat coding lebih mudah.

2. Tensorflow adalah tools andalan google setelah generasi pertama yaitu Distbelieve, untuk sejumlah aplikasi seperti pengenalan suara di aplikasi Google dan foto di Photos. Ia juga berada di balik fitur penjawab email Smart Reply yang baru-baru ini diluncurkan. TensorFlow sendiri dipastikan kini berlisensi Apache 2.0, bekerja dengan baik di perangkat komputer maupun smartphone. Ia dirancang untuk berjalan dengan baik pada prosesor, grafis, desktop, server dan platform komputasi mobile yang ada.

3. Tensoflow bisa diaplikasian ke bahasa lain selain Python, diantaranya R, Swift, hingga Javascript.

4. Dapat diaplikasikan melalui browser sehingga *programmer* dapat melatih dan menjalankan model langsung melalui browser berkat adanya TensorFlow.js.. Dengan begitu pengembang dapat mengenali prosesnya dengan mudah serta menemukan demo unik ketika menghadapi masalah.

5. Bisa memulai dan mempelajari coding *machine learning* dari project *machine learning* yang lain sehingga tidak perlu memulai dari nol.

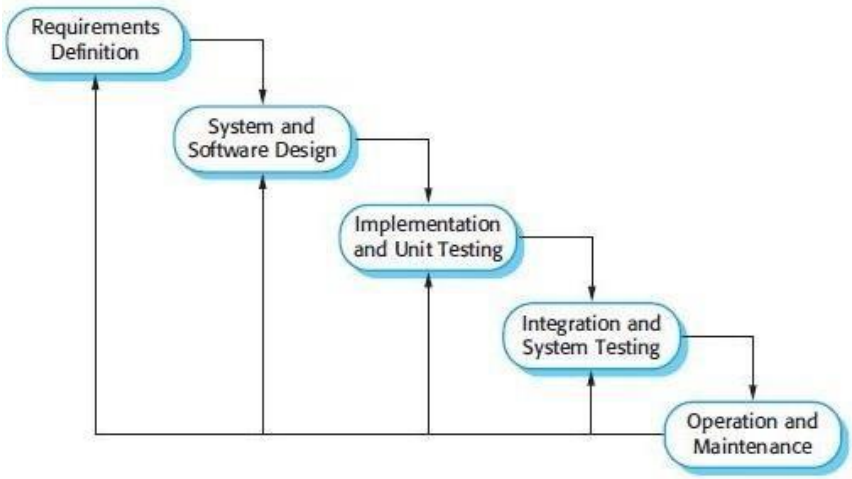
2.13 *Deep-Learning*

Bidang kecerdasan buatan sedang menghadapi era keemasan dikarenakan *deep-learning* perlahan menjadi pionir di domain ini. *Deep learning* menggunakan banyak lapisan yang merepresentasikan abstraksi data untuk

dijadikan model komputasi. Beberapa kunci penyedia algoritma *deep learning* seperti *generative adversarial networks, convolutional neural networks,* dan model transfer telah sepenuhnya merubah persepsi proses sebuah informasi.

2.14 Metode Waterfall

Metode yang digunakan dalam pengembangan sistem pada penelitian menggunakan siklus hidup pengembangan sistem SDLC (System Development Life Cycle) dengan Model Waterfall. Model ini melakukan pendekatan secara sistematis dan berurutan. Metode Waterfall mempunyai tahapan-tahapan yang digambarkan pada Gambar 2.2.



Gambar 2.1 Metode Waterfall (Rosa, 2016)

Menurut pernyataan gambar sebelumnya, model waterfall di uraikan dengan tahap-tahap sebagai berikut:

1. **Requirement Analysis and definition**

Merupakan tahapan penetapan fitur, analisa kendala dan tujuan sistem melalui konsultasi dengan pengguna sistem. Semua tahapan tersebut akan ditetapkan secara rinci dan berfungsi sebagai spesifikasi sistem.

2. System and Software Design

Merupakan tahapan pembentukan arsitektur sistem berdasarkan persyaratan yang telah ditetapkan pada tahap sebelumnya. Pada tahap ini juga mengidentifikasi dan menggambarkan abstraksi dasar sistem perangkat lunak yang akan dibuat serta hubungan- hubungannya.

3. Implementation and unit testing

Merupakan tahapan hasil dari desain perangkat lunak untuk direalisasikan sebagai satu set program atau unit program. Setiap unit akan diuji apakah sudah memenuhi spesifikasinya.

4. Integration and system testing

Merupakan tahapan pengintegrasian setiap unit program satu sama lain dan diuji sebagai satu sistem yang utuh untuk memastikan sistem sudah memenuhi persyaratan yang ada. Setelah itu sistem akan dikirim ke pengguna sistem.

5. Operation and Maintenance

Merupakan tahapan penginstalasian dan penerapan sistem. Pada tahap ini juga dilakukan pengujian pada saat sistem dijalankan untuk menemukan dan memperbaiki error yang tidak ditemukan pada tahap pembuatan. Dalam tahap ini juga dilakukan pengembangan sistem seperti penambahan fitur dan fungsi baru.

2.15 Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. (Rosa A.S, M.Shalahuddin,2016) Saat ini sudah banyak Bahasa pemrograman berorientasi objek. Banyak orang yang berfikir bahwa pemrograman berorientasi objek identic dengan basaha Java. Memang Bahasa java merupakan Bahasa yang paling konsisten dalam menimplementasikan paradigma pemrograman berorientasi objek. Namun sebenarnya Bahasa pemrograman yang mendukung pemrograman berorientasi objek tidak hanya Bahasa java. Contohnya: C++, PHP.

2.16 Skenario Testing

Pengujian berbasis skenario berkonsentrasi pada apa yang dikerjakan pengguna, bukan apa yang dikerjakan produk. Ini berarti menangkap tugas-tugas (melalui use case) yang harus dilakukan oleh pengguna dan kemudian menerapkan tugas-tugas beserta varian mereka sebagai pengujian-pengujian.

Skenario menemukan kesalah-kesalahan interaksi. Tetapi untuk mencapai hal ini, skenario pengujian harus lebih kompleks dan lebih realistis dari pada pengujian berbasis kesalahan. Pengujian berbasi skenario cenderung menjalankan banyak subsistem dalam satu pengujian tunggal (pengguna tidak membatasi diri

mereka terhadap terhadap penggunaan sebuah subsistem pada suatu waktu) (Rosa, 2016).

2.17 *Unified Modeling Language*

Rosa dan Shalahudin (2016:133), menjelaskan bahwa UML (Unified Modeling Language) adalah salah suatu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan design, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. UML menyediakan serangkaian gambar dan diagram yang sangat baik. Beberapa diagram memfokuskan diri pada ketangguhan teori object oriented dan sebagian lagi memfokuskan pada detail rancangan dan konstruksi. Semua dimaksudkan sebagai sarana komunikasi antar team programmer maupun dengan pengguna.

UML mempunyai sejumlah elemen grafis yang bias dikombinasikan menjadi diagram. UML memliki sejumah aturan untuk menggabungkan / mengkombinasikan elemen-elemen tersebut.

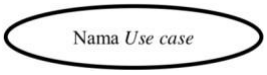
Berikut adalah diagram yang terdapat di dalam UML:

1. Use Case Diagram

Use case atau diagram use case merupakan pemodelan untuk kelakukan (behavior) sistem informasi yang akan dibuat. Use case mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem informasi yang akan dibuat. Secara kasar, use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem infomasi dan siapa saja yang berhak menggunakan funsi-fungsi itu.

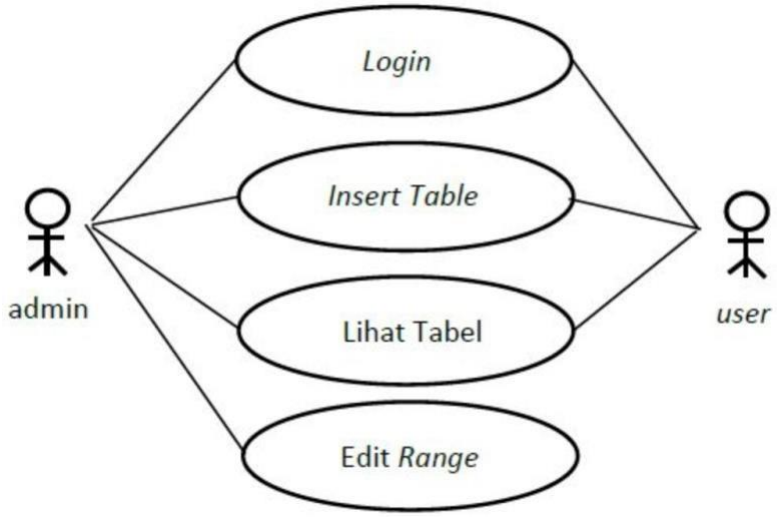
Simbol-simbol yang digunakan dalam use case Diagram yaitu:

Table 2.1 Simbol-simbol Use Case Diagram (Rosa dan Shalahuddin, 2016:155)

Use Case  	Fungsionalitas yang disediakan *system* sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Biasanya dinyatakan dengan menggunakan kata kerja du awal di awal frase nama *use case*.
Aktor / Actor  	Orang, proses, atau *system* lain yang berinteraksi dengan system informasi yang akan dibuat di luar *system* informasi yang akan dibuat itu sendiri jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi / Association 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memilki interaksi dengan aktor.

Ekstensi / Extend  	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu. Mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek.
Generalisasi / Generalization 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan / include  	Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini.

Contoh *use case diagram* sederhana:



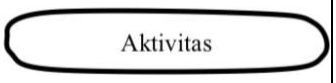
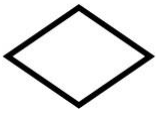
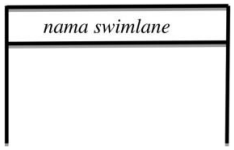
Gambar 2.2 Contoh Use Case Diagram

2. *Activity Diagram*

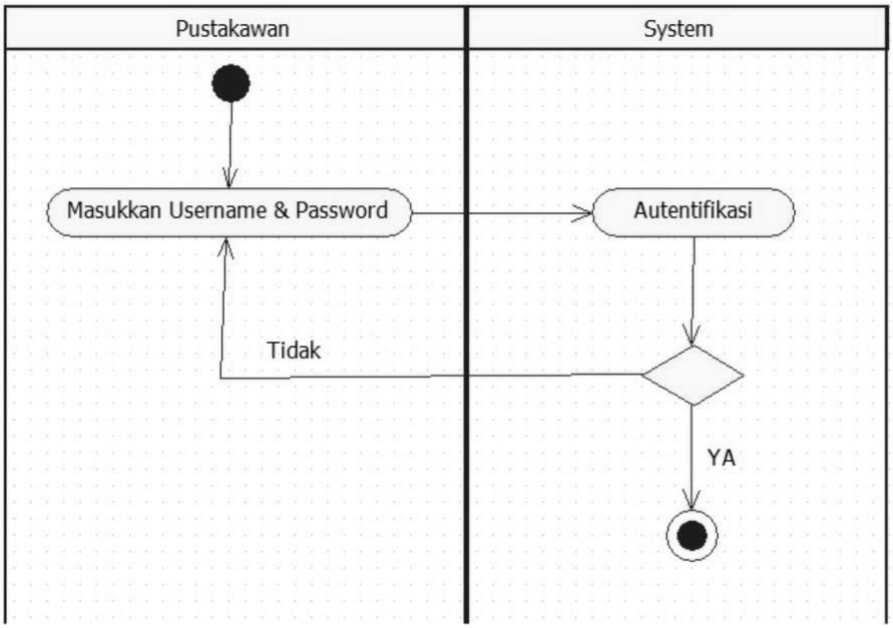
Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambaran aktivitas sistem bukan apa yang dilakukan aktor.

Simbol-simbol yang digunakan dalam Activity Diagram yaitu:

Table 2.2 Simbol-simbol Activity Diagram (Rosa dan Shalahuddin, 2016:162)

Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan *system*, aktivitas biasanya diawali dengan kata kerja.
Percabangan / decision 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan / join	Asosiasi penggabungan dimana lebih dari satu aktvitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan system, sebuah diagram aktivitas memliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Contoh *Activity Diagram* sederhana:



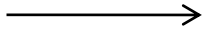
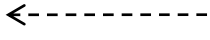
Gambar 2.3 Contoh Activity Diagram

3. *Sequence Diagram*

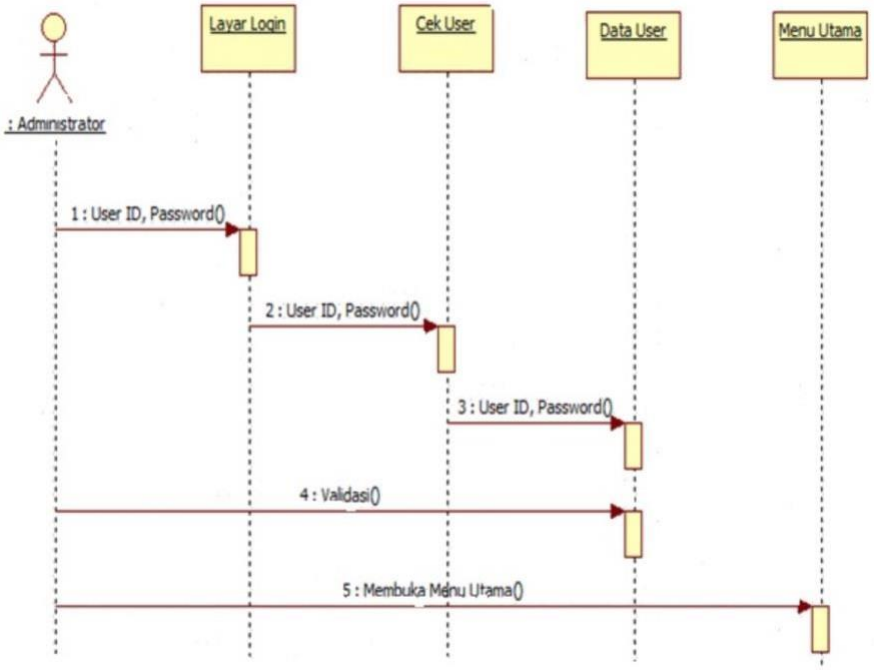
Diagram Sequence menggambarkan kelakuan objek pada use case dengan medeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek.

Simbol-simbol yang digunakan dalam Sequence Diagram yaitu:

Table 2.3 Simbol-simbol Sequence Diagram (Rosa dan Shalahuddin, 2018:165)

<<*Entity Class*>>	*Entity* class, merupakan bagian dari sistem *yang* berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
<<*Boundary Class*>>	*Boundary class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan form entry dan form cetak.
<<*Control Class*>>	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
Message 	*Message*, symbol mengirim pesan antar *class*.
Recursive 	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
Activation 	*Activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbandingan lurus dengan durasi aktivitas sebuah operasi.
Lifeline	*Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

Contoh *sequence diagram* sederhana:

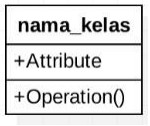


Gambar 2.4 Contoh Sequence Diagram

4. *Class Diagram*

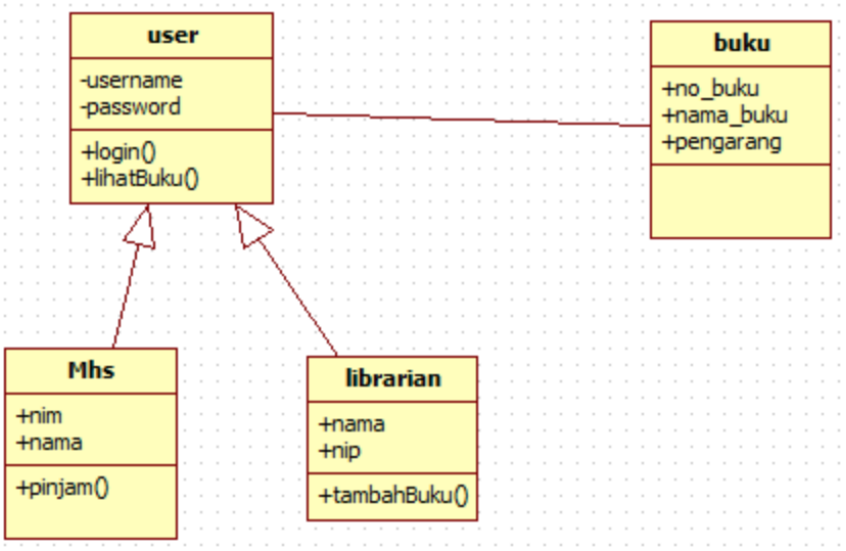
Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Simbol-simbol yang digunakan dalam Class Diagram yaitu:

Table 2.4 Simbol-simbol Class Diagram (Rosa dan Shalahuddin, 2016:146)

Kelas 	Kelas pada struktur sistem
Antarmuka / *interface* 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
Asosiasi berarah / *directed association* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
Generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi(umum khusus)

Kebergantungan / *dependency* 	Relasi antarkelas dengan makna kebergantugan antarkelas
Agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-port*)

Contoh *class diagram* sederhana:



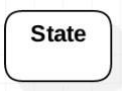
Gambar 2.5 Contoh Class Diagram

5. *Statechart Diagram*

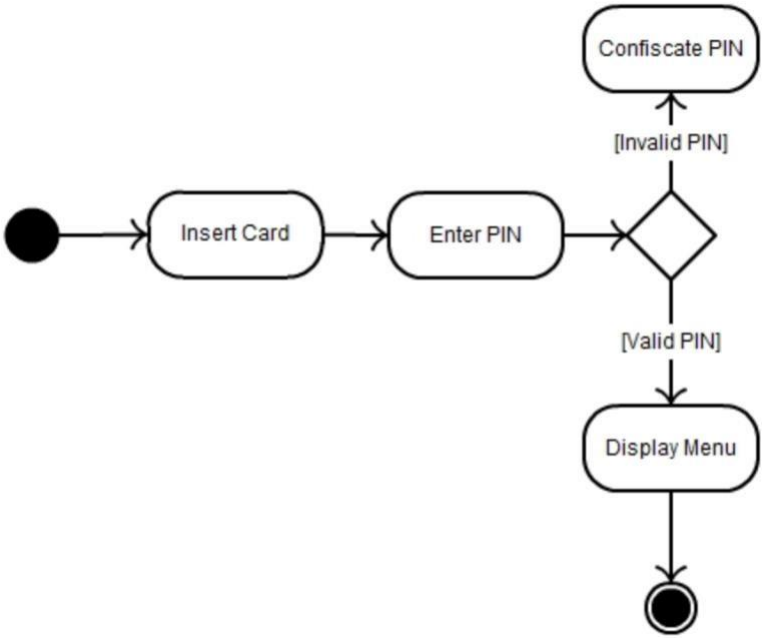
Statechart diagram digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek. Jika diagram sekuen digunakan untuk interaksi antar objek. Perubahan tersebut digambarkan dalam suatu graf berarah.

Simbol-simbol yang digunakan dalam Statechart Diagram yaitu:

Table 2.5 Simbol-simbol Statechart Diagram (Rosa dan Shalahuddin, 2016:163)

Start / Status Awal (*Initial State*) 	*Start* atau *initial state* adalah *state* atau keadaan awal pada saat sistem mulai hidup.
End / Status Akhir (*Final State*) 	*End* atau *final state* adalah *state* keadaan akhir dari daur hidup suatu sistem.
Event 	*Event* adalah kegiatan yang menyebabkan berubahnya status mesin.
State 	Sistem pada waktu tertentu. *State* dapat beubah jika ada event tertentu yang memicu purbahan tersebut.

Contoh *statechart diagram* sederhana:



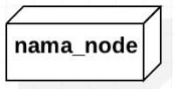
Gambar 2.6 Contoh Statechart Diagram

6. *Deployment Diagram*

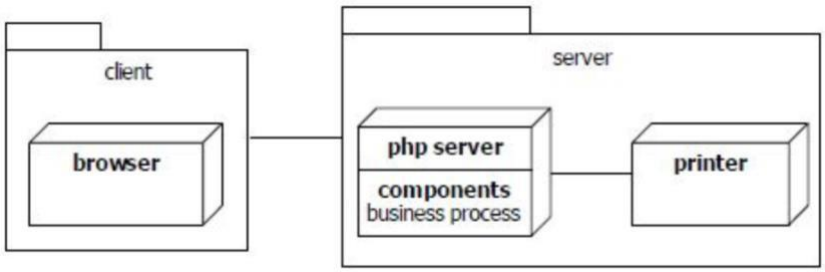
Diagram deployment atau deployment diagram menunjukan konfigurasi komponen dalam prose eksekusi aplikasi.

Simbol-simbol yang digunakan dalam Deployment Diagram yaitu:

Table 2.6 Simbol-simbol Deployment Diagram (Rosa dan Shalahuddin, 2016:154)

Package 	*Package* merupakan sebuah bungkusan dari satu atau lebih node.
Node 	Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak tidak dibuat sendiri (*software*).
Kebergantungan / *dependency* 	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.
Link 	Relasi antar *node*

Contoh *Deployment Diagram* sederhana:



Gambar 2.7 Contoh Deployment Diagram

2.18 *Basis Data (Database)*

Secara garis besar, kita bisa mengkategorisasi sistem teknologi informasi sebagai *Online trasactional processing* (OLTP) *system* dan *Online Analytical Processing* (OLAP) *system*. OLTP *system* dikategorisasikan dari banyaknya jumlah transaksi pendek online (*Read, Insert, Update, Delete)*. Di sekitar tahun 2008, terjadi luapan basis data baru yang terciptakan, dan tidak satupun dari sistem tersebut yang mengikuti implementasi relasi tradisional. Basis data yang baru ini, disebut juga sebagai *NoSQL databases*, yang didesain untuk menyimpan dan memproses pertumbuhan secara eksponensial dari kuantitas data ("*Big data*"). Basisdata *NoSQL* ini melakukan oportuniti dan tantangan sistem OLTP dan OLAP.

Basis data terdiri dari 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan Data adalah represtatsi fakta nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan yang diwujudkan dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

Sebagai satu kesatuan istilah, Basis Data (Database) sendiri dapat didefinisikan, himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.

Berikut merupakan nama beberapa aplikasi database yang dikategorikan menurut komersi dan nonkomersil:

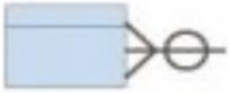
1. Aplikasi database komersil seperti: DB2, Microsoft SQL Server, Oracle, Sybase dan Teradata.
2. Aplikasi database nonkomersi seperti: MySQL, Firebird, PostgreSQL.

Ada juga database gratis yang biasa digunakan developer ketika sedang mengembangkan suatu aplikasi. Beberapa contohnya antara lain : PHPMyAdmin, MySQL Workbench, DBVisualizer, SQL Power Architect, Toad for MySQL, HeidiSQL, SQuirrel SQL, pgAdmin, phpPgadmin.

2.19 *Entity Relationship Diagram (ERD)*

Menurut Aprilia Cerry Natalin Rovita (2017:3) ERD adalah diagram yang menggambarkan keterhubungan antar data secara konseptual. Penggambaran keterhubungan antar data ini didasarkan pada anggapan bahwa dunia nyata dari kumpulan objek yang disebut entitas (entity), dan hubungan yang terjadi diantaranya yang disebut relasi (relationship).

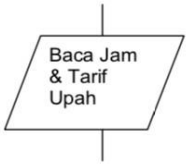
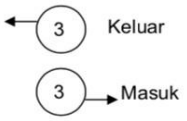
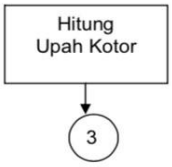
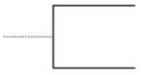
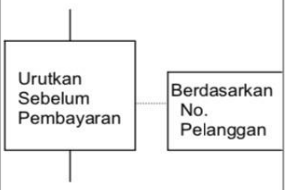
Table 2.7 Kardinalitas pada ERD versi James Martin

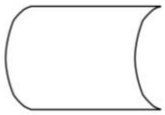
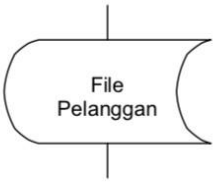
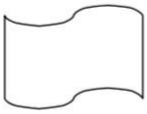
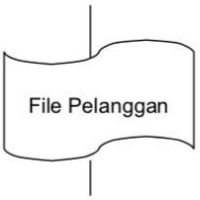
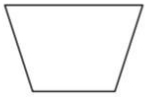
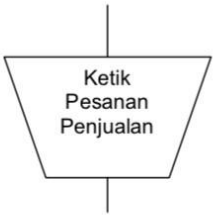
Simbol	Arti
	Satu dan hanya satu (One and only one)
	Satu atau lebih (One or more)
	Tidak ada atau lebih(Zero or more)
	Tidak ada atau satu (Zero or one)

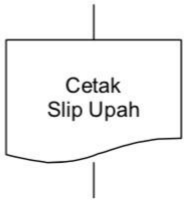
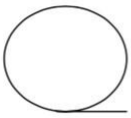
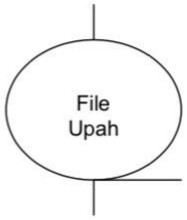
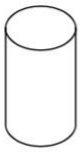
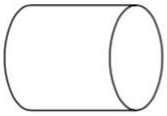
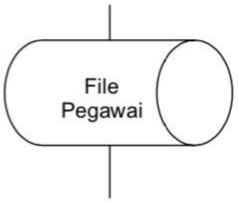
2.20 *Flowchart*

Menurut Eka Iswandy (2015:4) bahwa flowchart merupakan urutan-urutan langkah kerja suatu proses yang digambarkan dengan menggunakan simbol-simbol yang disusun secara sistematis. Simbol-simbol yang digunakan untuk menggambarkan flowchart dapat di lihat seperti pada gambar 2.8.

Table 2.8 Simbol - Simbol Flowchart

SIMBOL	ARTI	CONTOH
Input / Output 	Merepresentasikan Input data atau Output data yang diproses atau Informasi.	
Proses 	Mempresentasikan operasi	
Penghubung 	Keluar ke atau masuk dari bagian lain flowchart khususnya halaman yang sama	
Anak Panah 	Merepresentasikan alur kerja	
Penjelasan 	Digunakan untuk komentar tambahan	

SIMBOL	ARTI	CONTOH
On-line Storage 	I/O yang menggunakan penyimpanan akses langsung	
Punched Tape 	I/O yang menggunakan pita kertas berlubang	
Manual Input 	Input yang dimasukkan secara manual dari keyboard	
Display 	Output yang ditampilkan pada terminal	
Manual Operation 	Operasi Manual	

SIMBOL	ARTI	CONTOH
Dokumen 	I/O dalam format yang dicetak	
Magnetic Tape 	I/O yang menggunakan pita magnetik	
Magnetic Disk 	I/O yang menggunakan disk magnetik	
Magnetic Drum 	I/O yang menggunakan drum magnetik	

2.21 Fase Pertumbuhan Tumbuhan

Pertumbuhan tanaman merupakan hasil dari berbagai proses fisiologi, melibatkan faktor genotipe yang berinteraksi dalam tubuh tanaman dengan faktor lingkungan. Proses tersebut yaitu pertambahan ukuran, bentuk, dan jumlah.

Ciri-ciri pertumbuhan pada tanaman yang tampak sebagai *fenotipe*

utamanya dipengaruhi oleh faktor genotipe, sedangkan ciri-ciri lainnya ditentukan oleh pengaruh lingkungan sehingga pertumbuhan merupakan fungsi dari genotipe x lingkungan.Dalam usaha pertanian, aspek pertumbuhan tanaman mengacu pada tujuan utamanya yaitu memaksimalkan laju pertumbuhan dan hasil panen melalui manipulasi genetik dan lingkungan.

2.22 Fase Vegetatif Tumbuhan

Secara umum, pertumbuhan didefinisikan sebagai proses pembelahan dan pemanjangan sel. Pertumbuhan tanaman dalam arti terbatas menunjuk pada pertambahan ukuran yang tidak dapat balik, mencerminkan pertambahan protoplasma dan bobot kering pada tanaman. Pertambahan bobot kering umumnya digunakan sebagai penunjuk ciri pertumbuhan karena pada umumnya hal tersebut mempunyai kepentingan ekonomi yang paling besar. Adapun parameter lain di antaranya adalah tinggi, volume, dan luas daun juga dapat digunakan untuk mendeteksi adanya pertumbuhan pada tanaman. Adapun parameter lain yaitu bobot basah tidak banyak digunakan karena angkanya berfluktuasi walaupun pada kepentingan tertentu, parameter ini menjadi penting daripada bobot kering (digabung dengan faktor kualitas) terutama pada studi dan produksi hortikultura.

2.23 Faktor Pertumbuhan Tanaman

Secara umum, faktor pertumbuhan tanaman meliputi faktor internal (genetik) dan faktor eksternal (lingkungan).

1. Faktor Internal

Faktor internal yang mempengaruhi pertumbuhan tanaman adalah sebagai berikut:

1. Ketahanan terhadap tekanan iklim, tanah, dan biologis.
2. Laju fotosintesis.
3. Respirasi.
4. Pembagian hasil asimilasi dan nitrogen.
5. Klorofil, karoten, dan kandungan pigmen lainnya.
6. Tipe dan letak meristem.
7. Kapasitas untuk menyimpan cadangan makanan.
8. Aktivitas enzim.
9. Pengaruh langsung oleh gen, misalnya heterosis, epistatis.
10. Diferensiasi.
11. Faktor eksternal

2. Faktor eksternal yang mempengaruhi pertumbuhan tanaman adalah sebagai berikut:

1. Faktor iklim, meliputi cahaya, temperatur, air, panjang hari, angin, dan gas
2. Faktor edafik, meliputi tekstur, struktur, bahan organik, kapasitas pertukaran kation, pH, kejenuhan basa, dan ketersediaan nutrisi
3. Faktor biologis, meliputi gulma, serangga, organisme penyebab penyakit, nematoda, herbivora, dan mikroorganisme tanah

4. Adanya faktor pembatas dalam pertumbuhan tanaman berakibat pada terjadinya pengurangan pertumbuhan dan perkembangan tanaman.

2.24 Flowchart

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. *Flowchart* menolong analis dalam untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.

Flowchart biasanya mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut. Proses di lingkungan organisasi pada umumnya merupakan suatu rangkaian kegiatan yang berulang. Setiap siklus kegiatan tersebut biasanya dapat dipecahkan ke dalam beberapa langkah kecil. Dari uraian langkah-langkah tersebut, kita dapat mencari langkah mana saja yang bisa kita perbaiki(*improve*).

Langkah-langkah tersebut akan lebih mudah dimengerti jika kita menggambarkannya dalam suatu bagan yang dikenal dengan istilah: flowchart atau bagan alir.

Seperti yang dikemukakan Dr. W. Edwards Deming *"Draw a flowchart for whatever you do. Until you do, you do not know what you are doing, you just have a job*" Pentingnya *flowchart* juga menjadi perhatian Dr. Kaoru Ishikawa, tokoh kualitas Jepang, dengan menjadikan alat ini sebagai salah satu dari tujuh

alat kualitas dasar (*7 basic quality tools*) yang harus dikuasai oleh para anggota gugus kendali kualitas (*quality control circle*).

Dalam dokumen standar internasional keluaran ISO, *flowchart* didefinisikan sebagai:

> *1. A graphical representation of a process or the step-by-step solution of a problem, using suitably annotated geometric figures connected by flowlines for the purpose of designing or documenting a process or program (ISO/IEC 2382-1:1993 Information technology–Vocabulary–Part 1: Fundamental terms,01.05.06).*
>
> *2. Graphical representation of the definition, analysis, or method of solution of a problem in which symbols are used to represent operations, data, flow, equipment, etc. (ISO 5807:1985 Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.3).*
>
> *3. A control flow diagram in which suitably annotated geometrical figures are used to represent operations, data, or equipment, and arrows are used to indicate the sequential flow from one to another (ISO/IEC/IEEE 24765:2010 Systems and software engineering–Vocabulary).*

Jadi, *flowchart* adalah diagram yang menyatakan aliran proses dengan menggunakan anotasi bidang-bidang geometri, seperti lingkaran, persegi empat, wajik, oval, dan sebagainya untuk merepresentasikan langkah-langkah kegiatan beserta urutannya dengan menghubungkan masing masing langkah tersebut menggunakan tanda panah.

2.25 *Urban Farming*

Pertanian perkotaan (*Urban Farming*) menawarkan kesempatan untuk menyediakan makanan lokal yang segar bagi masyarakat perkotaan. Namun, pertanian perkotaan hanya dapat berhasil ditanamkan di daerah perkotaan jika konsumen memandang pertanian perkotaan secara positif dan menerima pertanian perkotaan dalam komunitas mereka. Keberhasilan pertanian perkotaan berakar pada persepsi positif dari mereka yang tinggal di dekatnya, dan persepsi tersebut sangat mempengaruhi penerimaan pertanian dalam kedekatan langsung individu.

Ketahanan dan keberlanjutan pangan merupakan perhatian utama karena urbanisasi yang cepat, kelangkaan lahan, dan rendahnya produksi pangan lokal berupa ikan dan sayuran berdaun.

Penurunan kualitas hidup yang dialami oleh masyarakat kota juga dapat kembali ditingkatkan lewat aktivitas berkebun di rumah yang menyegarkan pikiran.

Namun apabila dilihat dalam jangkauan yang lebih luas, *urban farming* memiliki dampak yang lebih besar bagi kelangsungan hidup masyarakat perkotaan. Sejumlah penelitian pun menyebutkan bahwa urban farming dapat menjadi konsep pertanian ideal di masa depan.

Masifnya pembangunan di perkotaan menyebabkan tergusurnya ruang-ruang terbuka hijau. Hilangnya ruang terbuka hijau sangat memengaruhi kestabilan ekosistem lingkungan, sekaligus meningkatkan polusi yang mana berdampak buruk bagi kesehatan masyarakat kota.

Konsep urban farming lantas menawarkan solusi dengan menciptakan lahan terbuka hijau ditengah padatnya bangunan perkotaan. *Urban farming* dapat mengelola wilayah perkotaan yang tercemar menjadi lingkungan yang nyaman dan sehat untuk ditinggali.

2.26 Pengkolaborasian Metode Supervised Learning

seseorang dapat melatih mesin dengan menggunakan data yang "diberi label". Artinya beberapa data sudah diberi label dengan jawaban yang benar. Ini dapat dibandingkan dengan pembelajaran yang berlangsung di hadapan pengawas atau guru. Algoritma pembelajaran yang terarah dapat mempelajari pola tersembunyi dari data pelatihan yang telah berlabel, hal ini akan membantu kita memprediksi hasil untuk data yang belum pernah dipelajari sebelumnya. Untuk dapat berhasil membangun, mengatur, dan menerapkan model mesin pembelajar yang terarah dengan akurasi tinggi, dibutuhkan waktu dan keahlian teknis dari tim peneliti-data (data scientist) yang sangat terampil. Selain itu, para

peneliti data sebaiknya harus mampu membangun kembali model untuk memastikan prediksi yang dihasilkan tetap benar walaupun datanya berganti.

Pembelajaran terarah memungkinkan kita untuk mengumpulkan data atau menghasilkan keluaran data berdasarkan dari pengalaman sebelumnya. Hal ini dapat membantu mengoptimalkan kriteria kinerja berdasarkan pengalaman mesin. Pembelajaran terarah juga dapat membantu memecahkan berbagai jenis masalah komputasi dunia nyata.

1. Penggunaan metode *Similarity Learning*

Metode ini diambil dari turunan *supervised learning* yang masuk kedalam kecerdasan buatan. *Similarity learning* mempunyai kaitan erat dengan *regression* dan *classification*, tetapi tujuannya adalah untuk mempelajari *similarity function* yang memastikan seberapa mirip atau seberapa kuat hubungan antara kedua objek yang disandingkan.

Dalam ilmu statiskika dan bidang yang berkaitan, *similarity function* adalah fungsi asli bernilai yang mengkuantifikasi kesamaan antara 2 objek. Meskipun tidak ada definisi tunggal dari ukuran kesamaan, biasanya ukuran seperti itu dalam beberapa hal merupakan kebalikan dari *distance metrics*. Teknik ini mengambil nilai besar untuk objek serupa dan nilai nol atau negatif untuk objek yang sangat berbeda.

2. *Regression* pada *Similarity Learning*

Pada situasi ini, pasangan objek di labelkan (x_i^1, x_i^2), bersama dengan mengukur persamaan mereka Y_1 € *R*. Tujuannya untuk mempelajari

fungsi yang memperkirakan f(x_i^1, x_i^2) ~ Y_i untuk setiap triplet label baru. Cara ini bias dilakukan dengan meminimalisir kekurangan/ketidak miripan yang masih bisa ditolerir . $\min_W \sum_i loss(w; x_i^1, x_i^2, y_i) + reg(w)$.

3. *Classification* pada *Similarity Learning*

Didapatkan sepasang objek yang memiliki kemiripan (X_1,X_2^+) dan benda yang tidak mirip (X_i, X_i^-). Diketahui sebuah formulasi persamaan dimana (x_i^1, x_i^2) diberikan bersama dengan *binary label* € {0,1} yang menentukan apakah objek memiliki kemiripan atau tidak. Tujuan nya tidak lain adalah untuk mengetahui *classifier* yang bisa menentukan apakah suatu pasangan benda memiliki kemiripan atau tidak.

3. Pengaplikasian Coding dengan konsep *Similarity Learning*

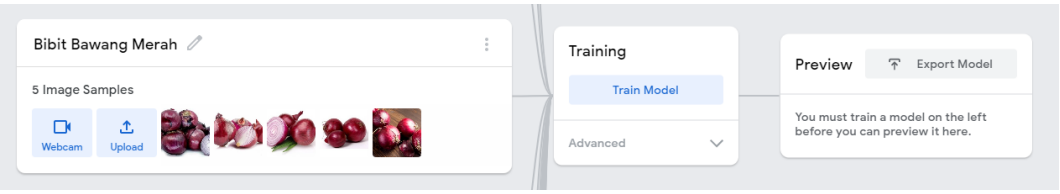
```
// Get a prediction for the current video frame
function classifyVideo() {
  flippedVideo = ml5.flipImage(video)
  classifier.classify(flippedVideo, gotResult);
  flippedVideo.remove();

}

// When we get a result
function gotResult(error, results) {
  // If there is an error
  if (error) {
    console.error(error);
    return;
  }
  // The results are in an array ordered by confidence.
  // console.log(results[0]);
  label = results[0].label;
  // Classifiy again!
  classifyVideo();
}
```

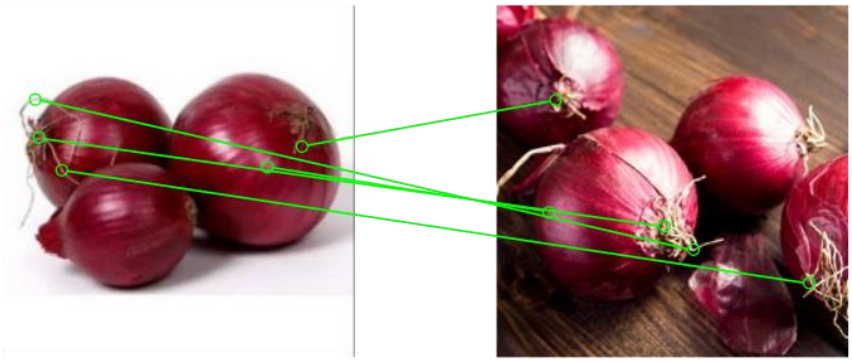
Gambar 2.8 Lampiran *Coding* metode *Classification* dalam *Similarity Learning*

Fungsi classifyVideo() diatas menerapkan cara *Classification* pada metode *Similarity Learning* kecerdasan buatan penelitian ini. Pengkodingan dapat berjalan ketika aplikasi sudah memiliki data model yang siap diproses.



Gambar 2.9 Penguploadan Model Data

Setelah *model data* di*upload* maka program dijalankan dengan analogi gambaran dari yang aplikasi lakukan seperti gambar dibawah ini



Gambar 3.0 Program Mencari Persamaan dan Pertidaksamaan Kedua Objek

BAB III

PEMBAHASAN

Dalam skripsi ini digunakan metode pengembangan SDLC (*System Development Life Cycle*) dengan model *Waterfall* (Pressman, 2015). Ada lima tahap dalam pengembangannya namun dalam skripsi ini dibatasi sampai dengan tahap empat. Keempat tahap tersebut yaitu, *Communication, planning*, *modeling*, dan *construction*.

3.1 Communication

Pada tahap ini dilakukan proses pengumpulan informasi berupa data yang berkaitan dengan penelitian.

1. Metode pengumpulan data

Metode pengumpulan data merupakan teknik atau cara yang dilakukan untuk mengumpulkan data. Proses pengumpulan data diantaranya di lakukan dengan metode wawancara dan studi *literatur*.

2. Metode Wawancara

Penulis melakukan wawancara terhadap salah seorang ahli kebun dan juga lulusan pertanian di kecamatan sukajadi dan parongpong bandung barat, dimana daerah ini khususnya daerah utara sebagian besar dari tanah nya merupakan lahan pertanian yang di garap oleh warga sekitar. Didapatkan kesimpulan bahwa masa generatif tumbuhan mempunyai ciri yang cukup bisa membedakan tanaman satu dengan yang lainnya, meskipun ada juga yang bisa dibedakan sejak berupa biji,

tapi sebagian besar tanaman khususnya pada sayur sayuran seperti bayam dan selada memiliki bentuk yang hampir sama bijinya yaitu berupa bulat kecil dan hanya memiliki sedikit perbedaan warna pada keduanya. Begitu biji tanaman disemai dan bertumbuh tunasnya, ciri-ciri tanaman yang cukup signifikan bisa terlihat dari batang pertama , bentuk daun, dan warnanya. Dengan ciri tersebut pula bisa diuraikan apa saja yang harus dilakukan agar tanaman bisa tumbuh dengan optimal.

3. Metode Studi Literatur

Studi literatur merupakan teknik yang digunakan untuk memperoleh informasi atau data dengan cara mempelajari buku-buku, jurnal, dan juga internet. Dalam skripsi ini sumber informasi juga didapat dari buku-buku serta jurnal yang berkaitan dengan *Artificial Intelligence*, pertanian *urban*, visi komputer, dan informasi lainnya yang berkaitan dengan skripsi, berikut adalah tabel literatur yang menjadi referensi:

Tabel 3.1 Referensi Penelitian

No	*Literature*	Pembahasan
1	Zhoe, Bolei, "*Interpretable Representation Learning for Visual Intelligence",* Jurnal Visi Komputer. Vol.1, No.1, Mei. 2018.	Penelitian ini membahas bagaimana Sistem Intelijensi Visual menggunakan 10 juta lebih gambar didalam *database* untuk bisa diproses dengan cara tertentu agar bisa dimanfaatkan menjadi informasi pengenalan objek dan skema yang bersangkutan dengan cara pengambilan *interest point* yang dijadikan sebagai parameter agar sistem bisa memproses gambar tersebut.
2	Theodoridis, Sergios, "*Machine Learning, A Bayesian And Optimization Perspective*", Jurnal Kecerdasan Buatan. Vol.1, No.2, 2015.	Penelitian ini berisi tentang bagaimana kecerdasan buatan bisa digunakan secara efektif dengan menggunakan cara mendasar yang berasal dari rumus aljabar yaitu *Classification* dan *regression*. Tahap selanjutnya setelah itu adalah melakukan probabilitas, *Stochastic Processes* untuk menentukan variabel yang didapat dari gambar. Konsep ini merujuk kepada salah

		satu cabang dari *artficial intelligence* yaitu *supervised learning*.
3	Rifqi Fauzi, Ahmad, "Pertanian Perkotaan : Urgensi, Peranan, Dan Praktik Terbaik", Jurnal Agroteknologi, Vol. 10 No.01, 2016.	Penelitian ini melakukan riset mengenai meningkatnya kebutuhan pangan masyarakat khususnya perkotaan sehingga pelaksaan *urban farming* adalah solusi terbaik agar masalah kebutuhan pangan ini bisa teratasi. Peranan pertanian perkotaan saat ini sudah memberikan dampak positif bukan hanya dalam pemenuhan kebutuhan pangan tetapi juga nilai-nilai praktis yang dapat berdampak bagi keberlanjutan ekologi maupun ekonomi wilayah perkotaan.

3.1.2 Analisa Permasalahan

Teknik *Edge Detector* pada visi komputer dan kecerdasan buatan ini memproses suatu gambar yang diberikan kepada sistem, mengolahnya dan menghasilkan *interest point* yang nantinya akan menjadi parameter untuk memperoleh informasi yang terkait pada gambar dengan terhubung pada database yang tersedia, lalu hasil akhirnya akan dimunculkan keputusan yang bisa

membantu pengguna dalam melakukan suatu pekerjaan yang berkaitan. *Hardware* yang dibutuhkan tentu saja kamera, komputer, & *Handphone* agar sistem dapat memperoleh gambar yang akan diproses. *User* membutuhkan keputusan yang tepat tetapi terkadang kesulitan dalam mengidentifikasi masalah, maka dengan teknik pengambilan gambar, *user* akan terbantu dalam memperoleh informasi yang tepat saat menghadapi permasalahan dalam bercocok tanam

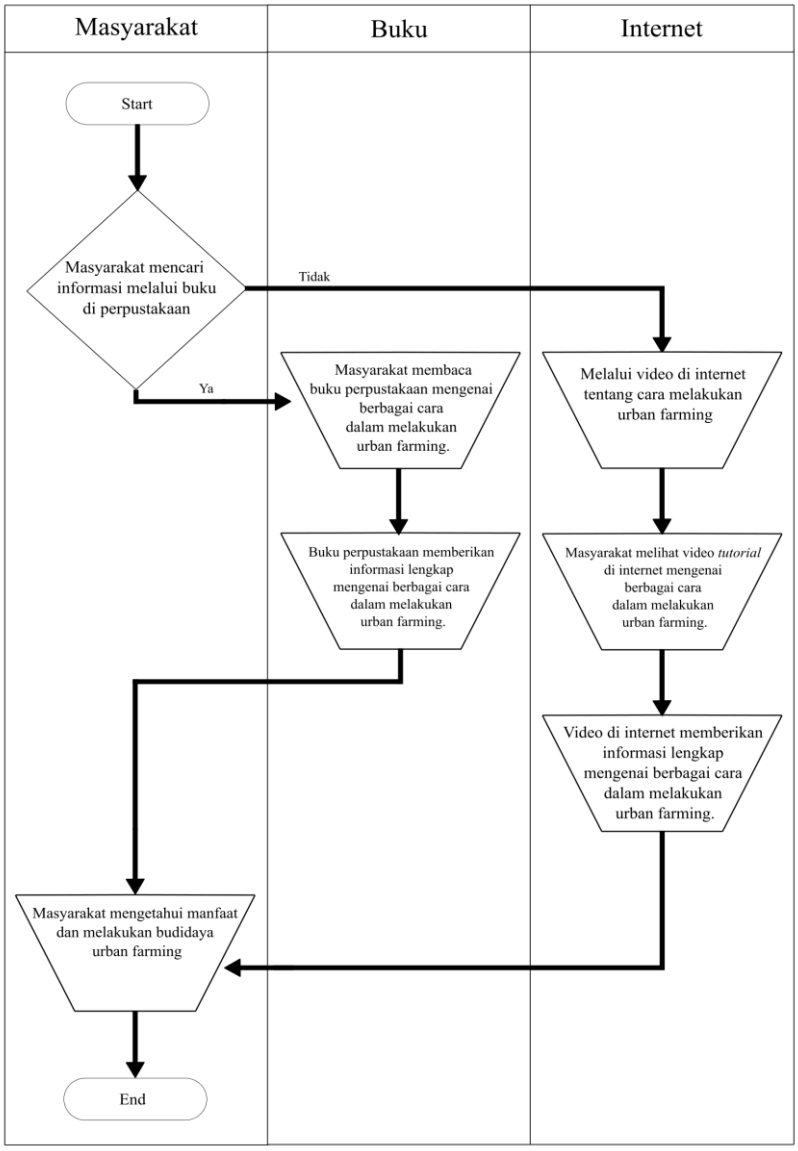
3.1.2.1 Analisis Proses/Prosedur yang Sedang Berjalan

Analisis terhadap sistem yang sedang berjalan bertujuan untuk mengetahui bagaimana cara kerja suatu sistem dalam aplikasi dan mengetahui masalah yang dihadapi sistem untuk dapat dijadikan sebagai landasan usulan perancangan. Tahap analisis diperlukan untuk mengetahui bagaimana proses pengenalan masalah yang terjadi saat melakukan *urban farming*. Pada tahap perancangan ini sistem yang berjalan digambarkan dalam bentuk prosedur *flowchart*. Berdasarkan hasil analisis proses *urban farming* yang dilakukan oleh masyarakat adalah sebagai berikut:

1. Masyarakat mencari informasi melalui buku di perpustakaan dan melihat video di internet tentang cara melakukan *urban farming*.
2. Masyarakat membaca buku dan melihat video di internet tentang bagaimana cara melakukan *urban farming*.
3. Buku di perpustakaan dan video *tutorial* di internet memberikan informasi lengkap mengenai berbagai cara dalam melakukan *urban farming*.

4. Masyarakat mengetahui manfaat dan melakukan budidaya *urban farming*.

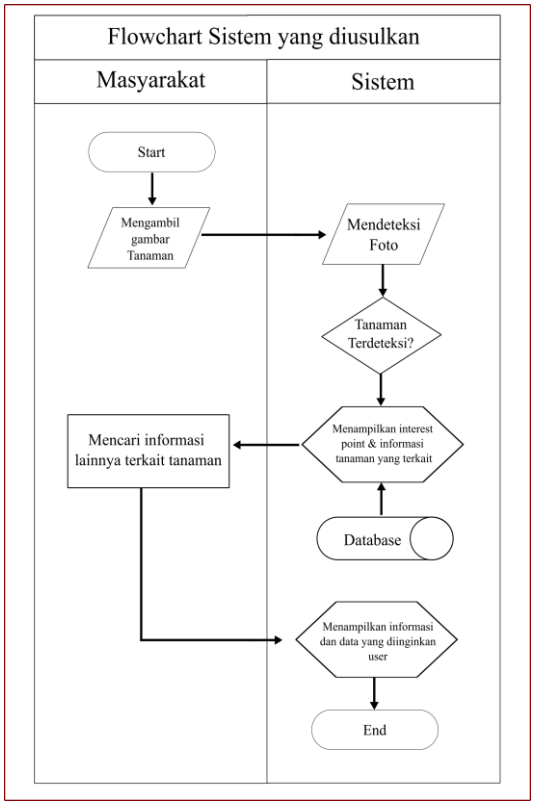
Adapun penggambaran *flowchart* berdasarkan prosedur diatas digambarkan dalam gambar 3.1.



Gambar 3.1 Flowchart sistem yang akan dibangun

3.1.3 Gambaran Umum Sistem Yang Diusulkan

Dalam skripsi ini aplikasi yang diusulkan merupakan sebuah aplikasi yang akan memberikan informasi secara terperinci melalui *artificial intelligence* mengenai tanaman yang *user* ambil gambar nya dengan metode *edge detector*. Setelah gambar diambil maka sistem akan memproses gambar yang nantinya akan memunculkan *interest point*, yaitu ciri dari objek yang nantinya akan di diketahui oleh sistem, tanaman apa dan apa masalah yang kira-kira ditemui oleh tanaman yang berada di gambar tersebut. Informasi yang terkait didapat dari *database* yang sudah tersedia. Gambaran sistem pada aplikasi yang diusulkan akan digambarkan pada gambar 3.2.



Gambar 3.2 Flowchart sistem yang akan diusulkan

3.1.4 Analisis Pengumpulan Data

Berdasarkan analisis permasalahan yang telah diuraikan diatas maka dalam skripsi ini akan dibuat sebuah aplikasi *android* kecerdasan buatan analisis tanaman generatif tumbuhan dengan teknik *edge detector* yang digunakan untuk membantu masyarakat agar mudah mengimplementasikan *urban farming* di perkotaan.

Aplikasi ini menggunakan database firebase dimana didalam firebase ini disimpan data tentang tanaman yang nantinya akan disamakan dengan objek dari kamera user untuk selanjutnya ditampilkan informasi dengan tanaman yang terkait.

Data training / past experience juga menjadi data yang berperan penting dalam aplikasi ini bila user memasukan gambar objek yang belum dikenal oleh sistem. Sistem akan memperbaharui data yang ada didalam database melalui *data training* yang diperoleh, sehingga semakin banyak *past experience* yang masuk, maka kemungkinan aplikasi dalam tidak mengenali objek akan semakin sedikit.

3.1.4 Analisis Kebutuhan Perangkat Keras dan Perangkat Lunak

Untuk membuat dan menjalankan aplikasi ini dibutuhkan persyaratan minimum perangkat keras agar aplikasi dapat berjalan. Berikut penjelasan syarat minimum perangkat keras yang digunakan untuk membangun dan menjalankan aplikasi.

A. Analisis Perangkat Keras *Personal Computer Developer*

1. Processor : Dual Core 3.0 Ghz / AMD Athlon(tm) II X2 240
2. Memory : RAM 3 GB
3. Harddisk : Free Storage Harddisk 5GB
4. Monitor : Resolusi Layar Minimal 1280 x 720
5. VGA : Minimal Intel(R) HD Graphics 512 mb

B. Analisis Perangkat Keras *Handphone Developer*

1. Processor : Quad-core 1.2 Ghz Co
2. Layar : Layar Sentuh
3. Resolusi Layar : 480 x 800 px
4. RAM : 2GB
5. ROM *free space* : 100MB
6. Kamera : 12MP (belakang)

Selain perangkat keras untuk membangun dan menjalankan aplikasi ini, dibutuhkan persyaratan minimum perangkat lunak agar aplikasi dapat berjalan. Dibawah ini menjelaskan persyaratan minimum perangkat lunak yang digunakan.

A. Analisis Perangkat Lunak *Personal Computer Developer*

1. Sistem Operasi : Minimal versi Windows 7 64 bit

2. Perangkat Lunak Pendukung :

    a. Android Studio 4.0

    b. Inkspace 1.0

B. Analisis Perangkat Lunak *Handphone Developer*

Sistem Operasi : Android versi 9.0 Pie

3.2 *Planning*

Dalam tahap ini penelitian memfokuskan pada penjadwalan pengerjaan penelitian. Pada penelitian ini terdapat beberapa proses yang harus dilakukan dari tahap *communication* hingga *implementation* dan *testing* maka dari itu diperlukan penjadwalan yang tepat agar penelitian ini dapat selesai tepat pada waktunya, berikut penjadwalan pengerjaan skripsi ini berdasarkan aktifitas yang dilakukan dengan skala waktu satu minggu. Tabel 3.2 dibawah ini akan menjelaskan penjadwalan penelitian.

Tahapan	No	Aktifitas	Mei				Juni				Juli				Agustus				September	
			1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Communication	1	Pengumpulan data																		
	2	Analisis Permasalahan dan Kebutuhan																		
	3	Pendefinisian Fungsi																		
Scheduling,Estimation	1	Penjadwalan																		
	2	Analisis Kebutuhan Perangkat Keras dan Perangkat Lunak																		
Analysis and Design	1	Perancangan Sistem menggunakan (UML dan ERD)																		
	2	Perancangan *User Interface*																		
Construction	1	Pengkodean																		
	2	Testing (*Blackbox*)																		

Tabel 3.2 Penjadwalan

3.3 Modelling

Pada tahap ini dilakukan perancangan aplikasi yang akan dibangun, meliputi perancangan diagram dan perancangan antarmuka dari aplikasi yang akan dibangun.

3.3.1 Definisi Aktor

Masyarakat disini bisa juga disebut *user*, dimana perannya adalah orang yang akan menggunakan aplikasi ini.

3.3.1.1 Definisi Use Case

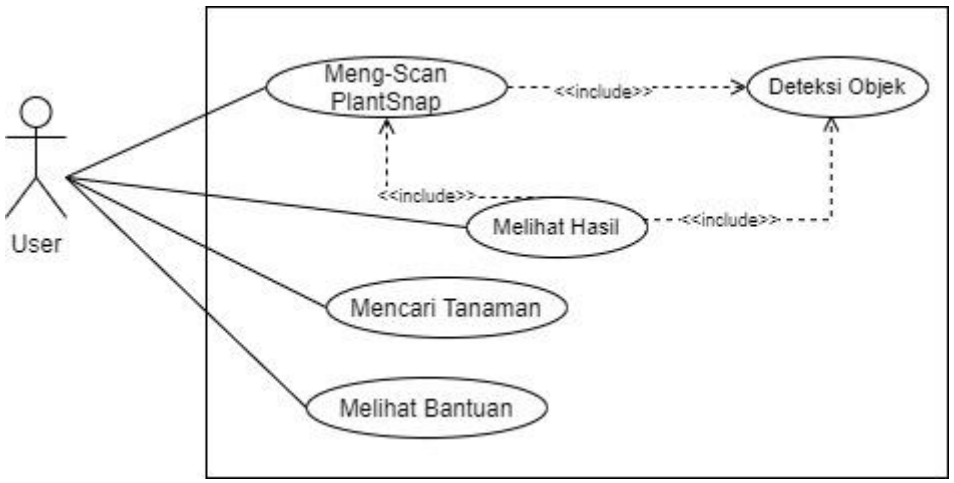
Use Case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

Tabel 3.3 Tabel Deskripsi Perancangan Use Case

No	Deskripsi
1	*User* adalah pihak yang akan menggunakan aplikasi tanamurban, dimana *user* ini dapat menggunakan semua fitur didalam aplikasi.
2	Merupakan fitur dimana *user* mengambil foto menggunakan kamera yang nantinya akan di analisis oleh aplikasi, setelah objek foto mempunyai kesamaan dengan salah satu data yang ada di database, aplikasi akan menampilkan informasi tentang objek tanaman yang terkait.
3	Merupakan fitur dimana *user* tidak mencari informasi tanaman melalui foto melainkan melalui pencarian berdasarkan kata yang ditulis *user* didalam menu pencarian.
4	Merupakan fitur untuk memberitahu *user* cara menggunakan aplikasi Tanam Urban dengan benar.

3.3.1 Perancangan *Use Case*

Use Case diagram digunakan untuk menggambarkan sistem dari sudut pandang pengguna sistem tersebut. *Use Case* untuk aplikasi android Tanam Urban akan dijelaskan pada gambar 3.3.



Gambar 3.3 Use Case Diagram Aplikasi Tanam Urban

a. Use Case Skenario

Berdasarkan *use case* diagram aplikasi diatas maka dibuatlah skenario dari tiap proses yang ada pada *use case* diagram tersebut untuk memudahkan dalam menganalisa skenario yang akan digunakan dalam fase selanjutnya. Untuk memudahkan proses analisa, maka skenario dari setiap *use case* akan dibuat menjadi beberapa tabel skenario sebagai berikut:

Tabel 3.4 Skenario Use Case Plantsnap

Identifikasi	
Use Case	Plantsnap
Aktor	*User*
Deskripsi	Merupakan fitur dimana *user* mengambil foto menggunakan kamera yang nantinya akan di analisis oleh aplikasi, setelah objek foto mempunyai kesamaan dengan salah satu data yang ada di database, aplikasi akan menampilkan informasi tentang objek tanaman yang terkait.
Kondisi Awal	Membuka menu Plantsnap
Aktivitas Aktor	Aktor mengarahkan kamera ke tanaman
Aktivitas Sistem	Menampilkan kamera, melakukan identifikasi terhadap tanaman dan menampilkan informasi tanaman.
Kondisi Akhir	Tampil informasi mengenai tanaman yang difoto sebelumnya.

Tabel 3.5 Skenario Use Case Cari Tanaman

Identifikasi	
Use Case	Cari Tanaman
Aktor	*User*
Deskripsi	Merupakan fitur dimana *user* tidak mencari informasi tanaman melalui foto melainkan melalui pencarian berdasarkan kata yang ditulis *user* didalam menu pencarian.
Kondisi Awal	Menampilkan menu pencarian Tanaman

Aktivitas Aktor	Aktor memasukan kata di menu pencarian
Aktivitas Sistem	Menampilkan informasi tanaman sesuai dengan kata yang dimasukan *user*.
Kondisi Akhir	Tampil informasi mengenai tanaman.

Tabel 3.6 Skenario Use Case Bantuan

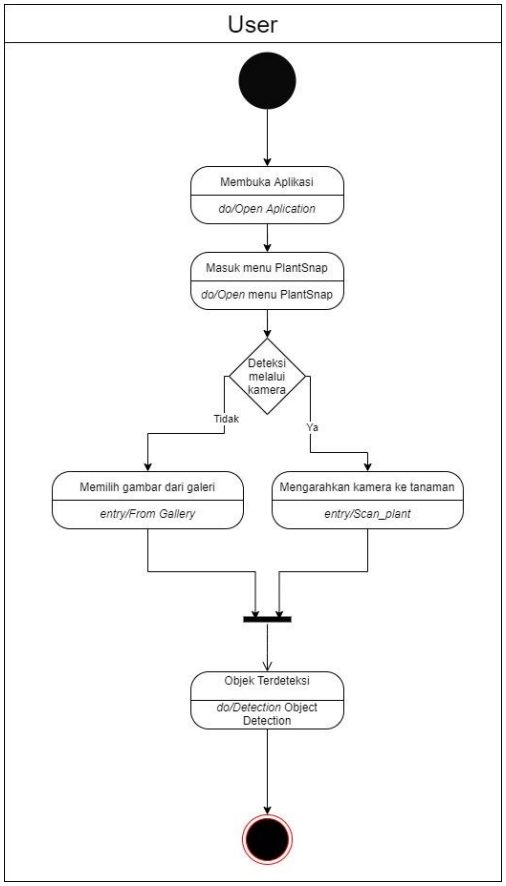
Identifikasi	
Use Case	Bantuan
Aktor	*User*
Deskripsi	Merupakan fitur untuk memberitahu *user* cara menggunakan aplikasi Tanam Urban dengan benar.
Kondisi Awal	Membuka menu Bantuan
Aktivitas Aktor	Aktor memilih bantuan mengenai salah satu fitur
Aktivitas Sistem	Menampilkan informasi cara menggunakan fitur.
Kondisi Akhir	Tampil informasi mengenai cara penggunaan, dan tombol kembali ke menu utama.

3.3.2 Activity Diagram

Activity diagram digunakan untuk menggambarkan *event – event* yang terjadi dalam *use case diagram*. Berdasarkan *use case diagram* aplikasi Tanam urban pada gambar 3.3, maka dibuat *activity diagram* dibawah ini.

1. *Activity Diagram* Plantsnap

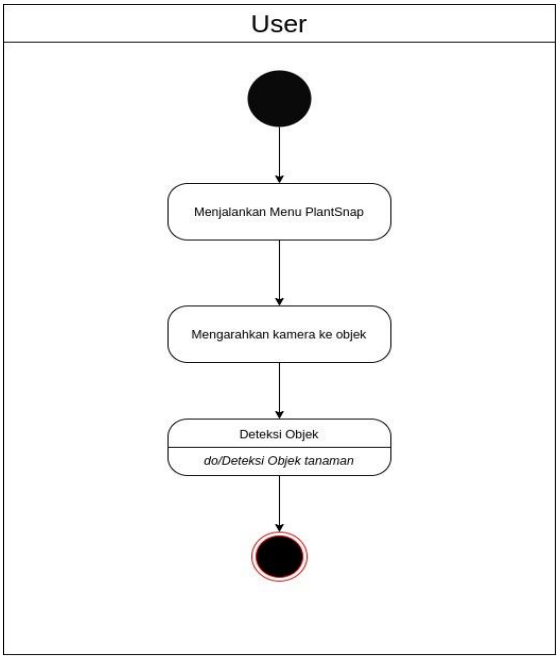
Activity Diagram Plantsnap pada gambar 3.4 menggambarkan bagaimana proses identifikasi tanaman. Pengguna harus melakukan foto terlebih dahulu untuk bisa melihat informasi mengenai tanaman yang terkait.



Gambar 3.4 Activiy Diagram Plantsnap

2. *Activity Diagram* Deteksi Objek

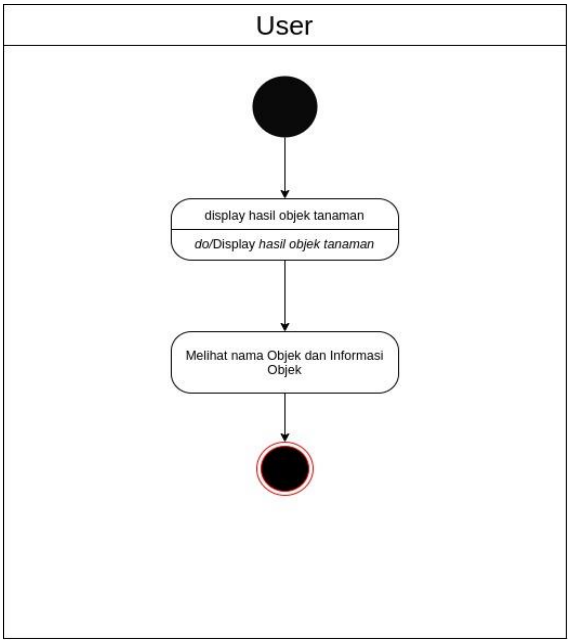
Activity Diagram deteksi objek pada gambar 3.5 menggambarkan bagaimana proses aplikasi melakukan deteksi objek.



Gambar 3.5 Activiy Diagram Deteksi Objek

3. *Activity Diagram* Hasil

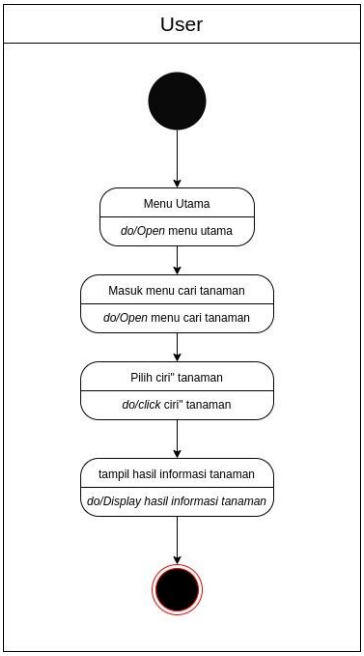
Activity Diagram hasil pada gambar 3.6 menggambarkan bagaimana proses penampilan hasil informasi mengenai setelah deteksi objek berhasil.



Gambar 3.6 Activiy Diagram Hasil

4. *Activity Diagram* Cari Tanaman

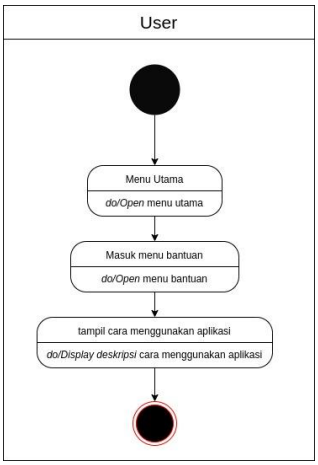
Activity Diagram Cari Tanaman pada gambar 3.7 menggambarkan bagaimana proses penampilan informasi tanaman sesuai masukan *user*. Pengguna harus memasukan nama tanaman terlebih dahulu untuk bisa melihat informasi mengenai tanaman yang terkait.



Gambar 3.7 Activity Diagram Cari Tanaman

5. *Activity Diagram* Bantuan

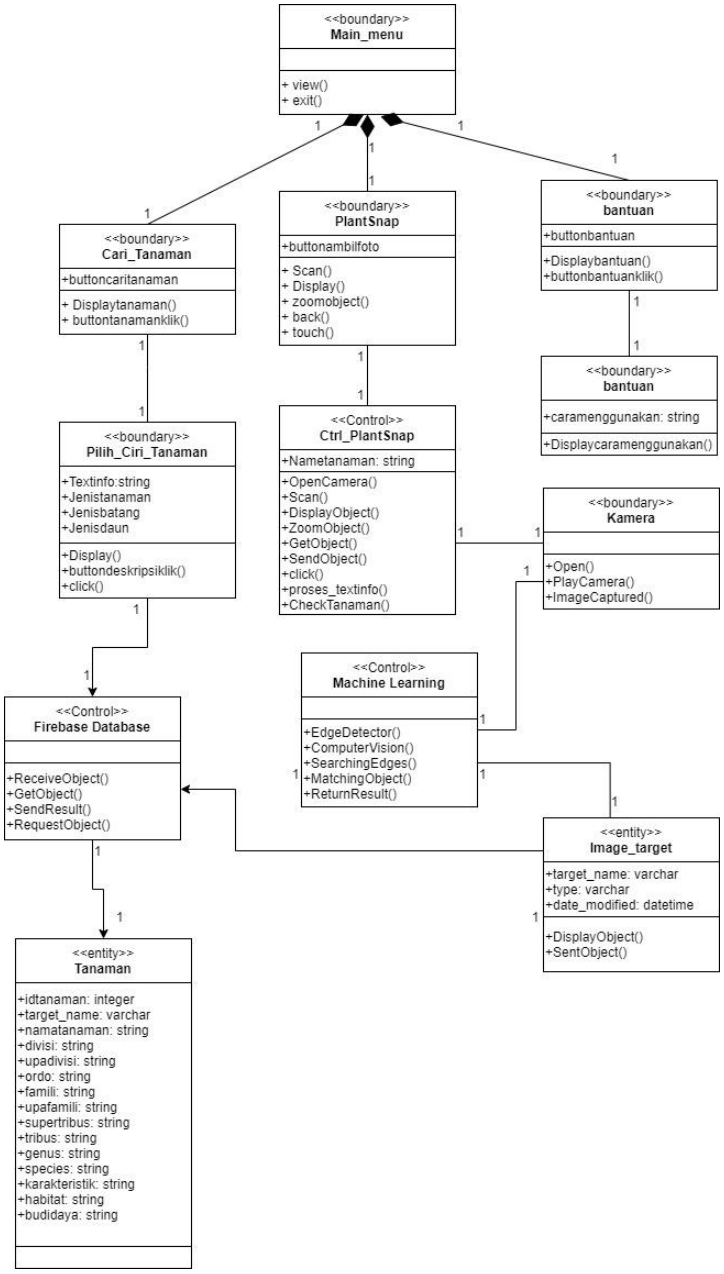
Activity Diagram Plantsnap pada gambar 3.8 menggambarkan bagaimana proses penampilan informasi mengenai cara *user* menggunakan aplikasi.



Gambar 3.8 Activity Diagram Bantuan

3.3.3 Class Diagram

Class diagram menggambarkan striktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Berikut ini gambar 3.9 yang merupakan *class diagram* dari aplikasi android Tanam Urban.



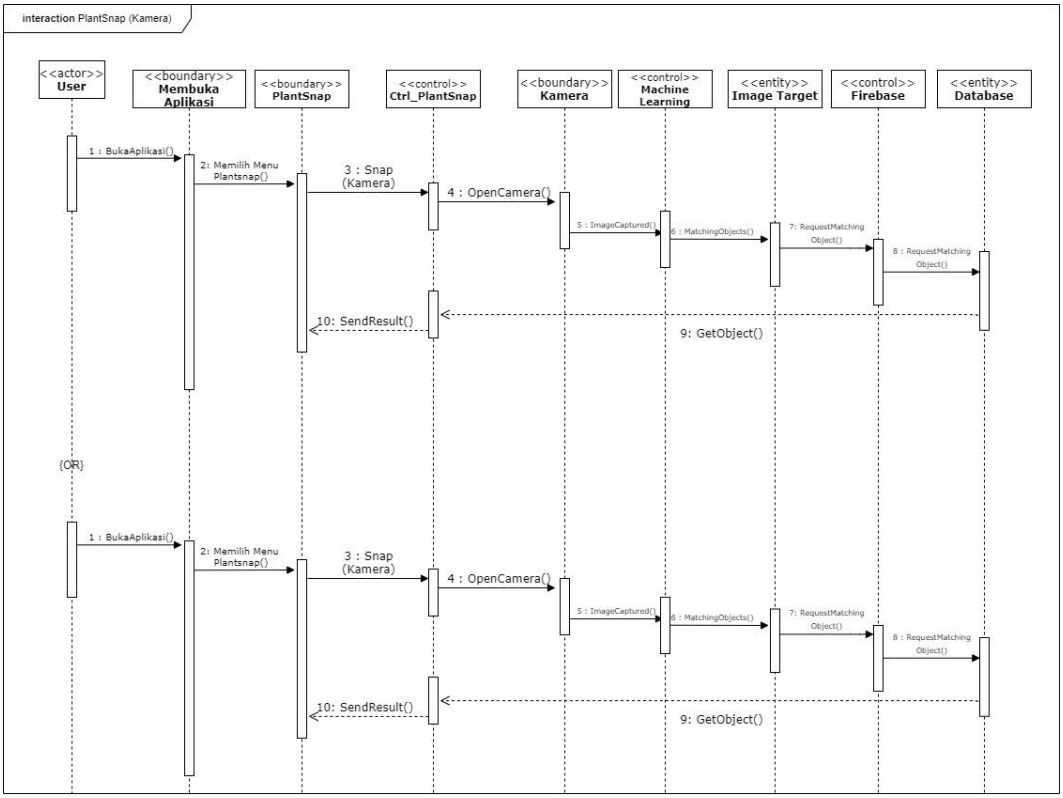
Gambar 3.9 *Class* Diagram Bantuan

3.3.4 Sequence Diagram

Sequence diagram merupakan suatu diagram yang menggambarkan interaksi antara sebuah objek dalam urutan waktu. Kegunaannya yaitu untuk menunjukan rangkaian pesan yang dikirim antar objek juga interaksi antara objek yang terjadi pada titik tertentu dalam eksekusi sistem. *Sequence Diagram* aplikasi Tanam urban dijelaskan pada gambar 3.10 sampai dengan 3.12

1. *Sequence Diagram* Plantsnap

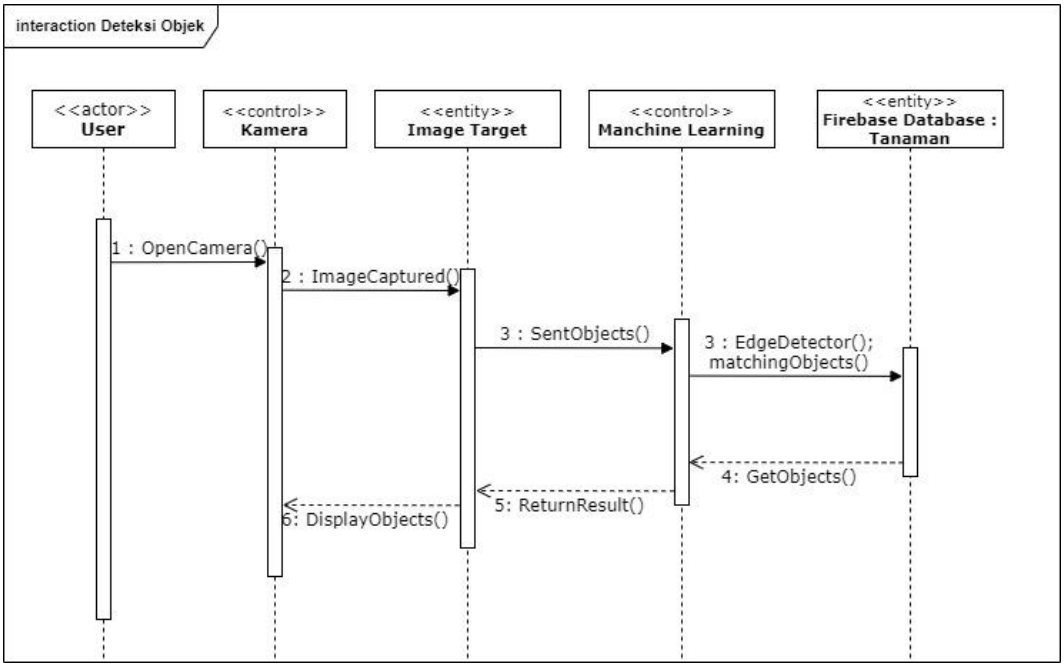
Pada diagram Plantsnap menjelaskan alur interaksi saat proses melakukan foto terhadap tanaman yang ada untuk menampilkan informasi.



Gambar 3.10 Sequence Diagram Plantsnap

2. *Sequence Diagram* Deteksi Objek

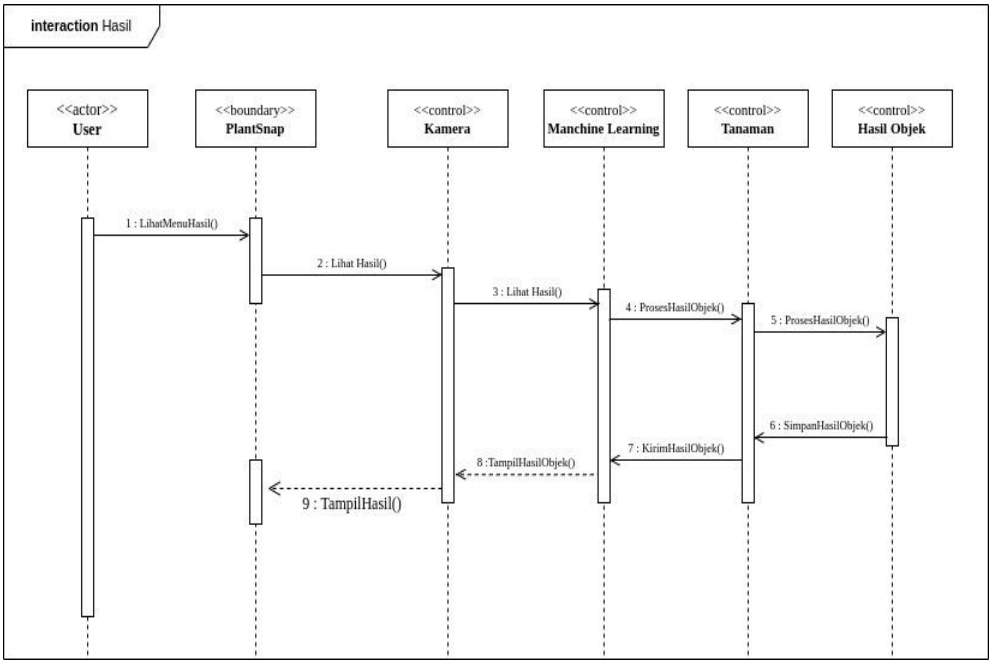
Pada diagram Cari Tanaman menjelaskan alur interaksi saat proses melakukan pencarian objek dengan metode *edge detector* pada *machine learning* pada data yang terkait untuk menampilkan informasi.



Gambar 3.11 Sequence Diagram Deteksi Objek

3. *Sequence Diagram* Hasil

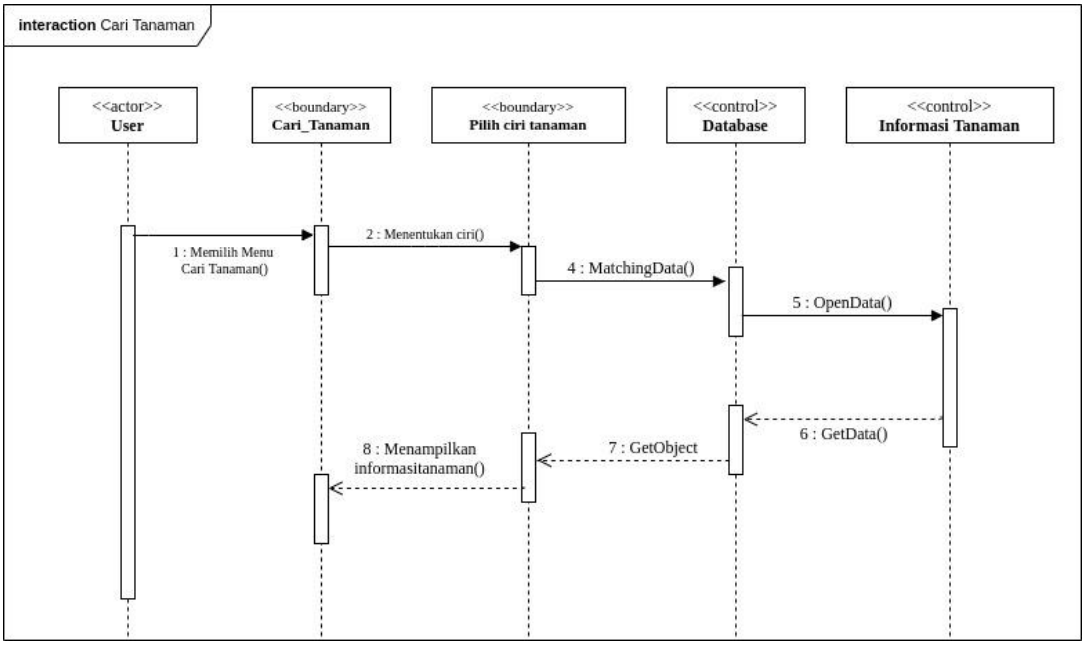
Pada diagram Cari Tanaman menjelaskan alur interaksi saat proses program menampilkan informasi.



Gambar 3.12 Sequence Diagram Hasil

4. *Sequence Diagram* Cari Tanaman

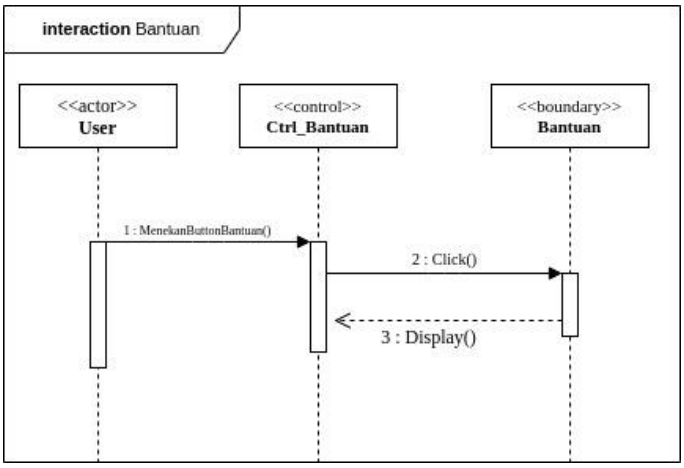
Pada diagram Cari Tanaman menjelaskan alur interaksi saat proses melakukan pencarian kata terhadap tanaman yang terkait untuk menampilkan informasi.



Gambar 3.13 Sequence Diagram Cari Tanaman

5. *Sequence Diagram* Bantuan

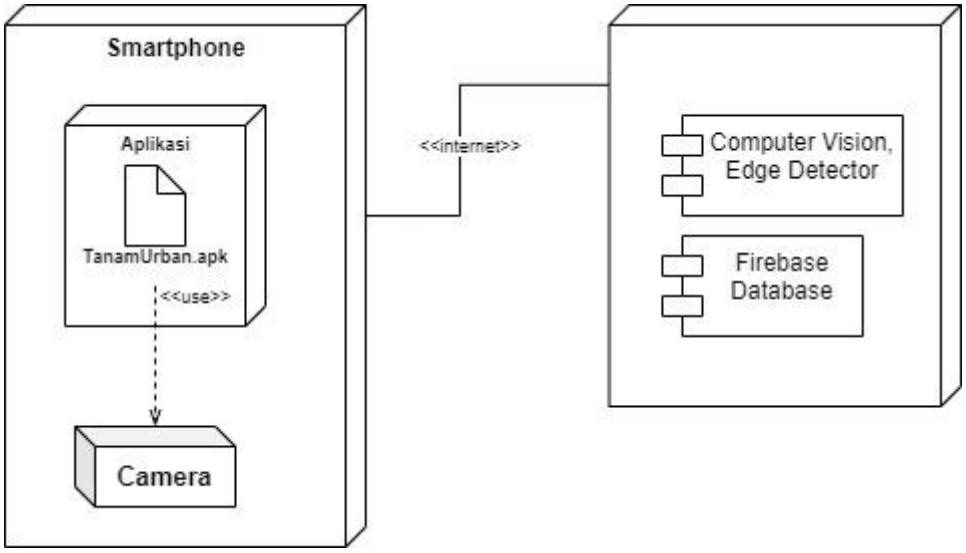
Pada diagram Bantuan menjelaskan alur interaksi saat *user* kurang paham dengan cara penggunaan aplikasi yang ada,sehingga ditampilkan informasi mengenai cara menggunakan aplikasi.



Gambar 3.14 Sequence Diagram Bantuan

3.3.5 Deployment Diagram

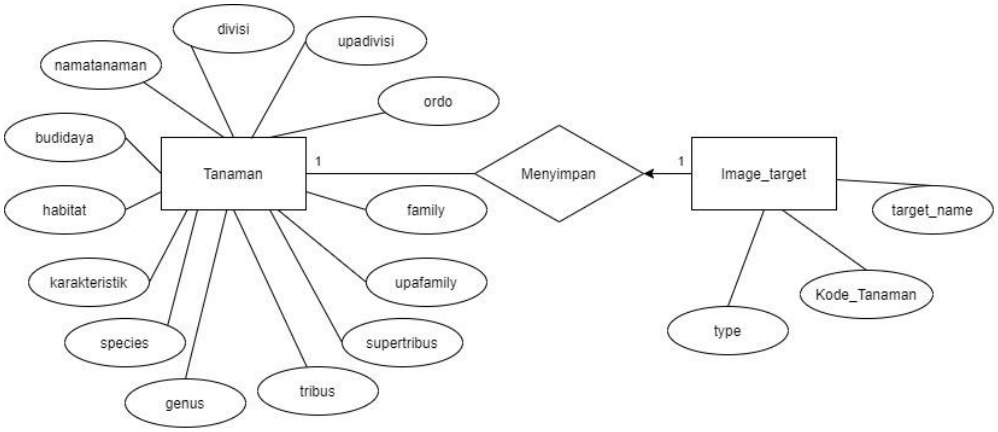
Deployment diagram Tanam Urban menjelaskan hubungan antar user, aplikasi dan *database*. Model *deployment diagram* aplikasi Tanam Urban dapat dilihat pada gambar 3.15.



Gambar 3.15 Deployment Diagram Aplikasi Tanam Urban

3.3.6 Entity Relationship Diagran (ERD)

Entity relationship diagram meupakan suatu model yang menjelaskan hubungan antar data dalam database berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. *Entity relationship diagram* memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol. ERD Tanam Urban dapat dilihat pada gambar 3.16



Gambar 3.16 Entity Relationship Diagram Aplikasi Tanam Urban

Terdapat dua tabel yang ada pada aplikasi Tanam Urban diantaranya tabel *image target* dan Tanaman. Pada Tabel 3.7 menunjukan penjelasan dari isi tabel *image target*.

Tabel 3.7 Tabel Image Target

Nama	**Tipe Data**	**Ukuran**	**Key**
Target_name	Varchar	70	*Primary-Key*
Type	Varchar	15	
Unique_code	Varchar	25	*Foreign-Key*

Tabel Tanaman berisikan data dan informasi mengenai tanaman yang akan ditampilkan.

Tabel 3.7 Tabel Tanaman

Nama	**Tipe Data**	**Ukuran**	**Key**
idtanaman	Varchar	10	*Primary-Key*
Target_name	Varchar	70	*Foreign-Key*
namatanaman	Varchar	30	

divisi	Varchar	20	
upadivisi	Varchar	20	
ordo	Varchar	20	
family	Varchar	20	
upafamily	Varchar	20	
supertibus	Varchar	20	
tribus	Varchar	20	
genus	Varchar	20	
species	Varchar	20	
karakteristik	Varchar	20	
habitat	Varchar	20	
budidaya	Varchar	20	

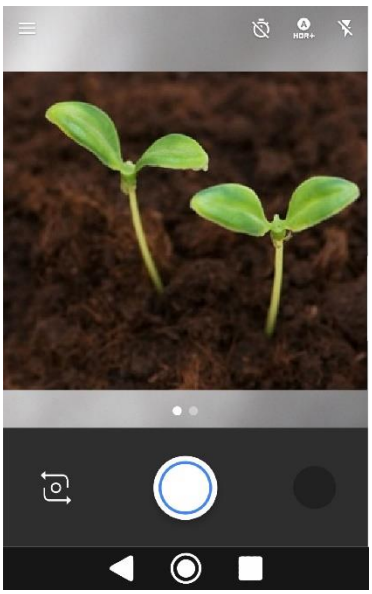
3.3.9 *Design Interface*

Setelah melakukan analisis diagram pada tahap kali ini akan dilakukan analisis desain *interface* yang akan dibangun, *interface* yang akan dibangun terdiri dari menu utama, menu Plantsnap, menu cari tanaman, dan menu bantuan.

1. Perancangan *Interface* Menu Utama

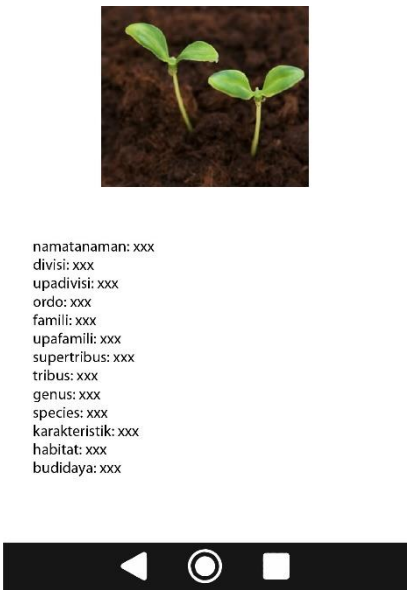
Gambar 3.17 Perancangan Menu Utama

2. Perancangan *Interface* Menu Plantsnap (Mengambil Gambar)



Gambar 3.18 Perancangan Menu Plantsnap (Mengambil Gambar)

3. Perancangan *Interface* Menu Plantsnap (Informasi Gambar)



Gambar 3.19 Perancangan Menu Plantsnap (Informasi Gambar)

4. Perancangan *Interface* Menu Cari Tanaman

Gambar 3.20 Perancangan Menu Cari Tanaman

5. Perancangan *Interface* Menu Bantuan

Gambar 3.21 Perancangan Menu Bantuan

3.4. *Construction*

Dalam tahap ini penelitian berfokus pada pengkodean menggunakan bahasa pemrograman Java dan menggunakan Android Studio, memproses data model untuk dijadikan data *tarining*, setelah itu dilakukan pengujian hasil menggunakan metode *blackbox testing*.

BAB IV

IMPLEMENTASI DAN UJI COBA

4.1 Construction (Code & Test)

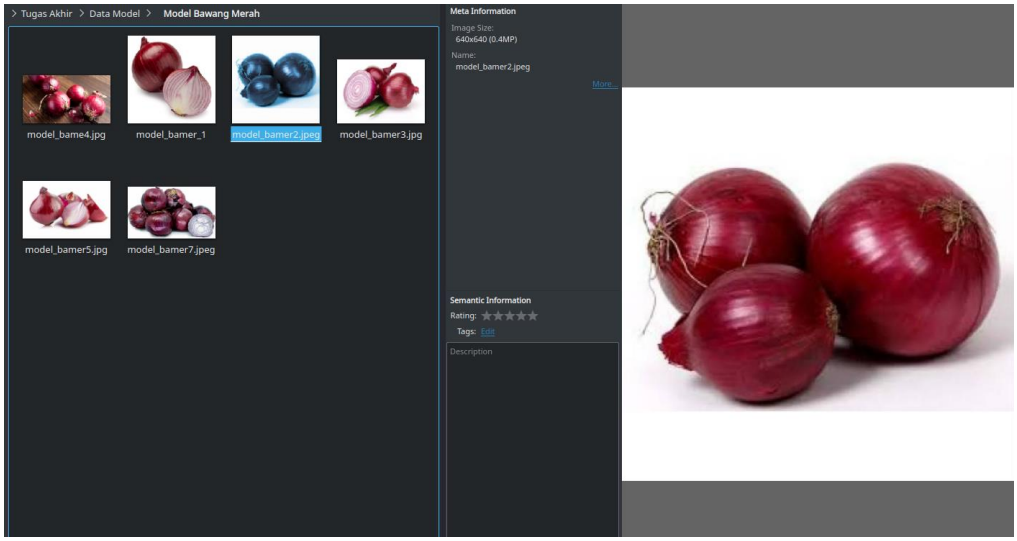
Dalam tahapan ini penelitian berfokus pada pengkodean meanggunakan bahasa Python dan Java. Setelah itu dilakukan pengujian data model yang sudah disusun sebelumnya didalam pengklasifikasian objek dan pengujian fungsi menggunakan metode blackbox testing. *Blackbox testing* ini berfungsi untuk menguji spesifikasi suatu fungsi atau modul, apakah berjalan sesuai yang diharapkan atau tidak.

4.1.1 Implementasi

Dalam tahapan ini metode *edge detector* dan *machine learning* akan diimplementasikan ke dalam apliasi menggunakan bahasa pemrograman Java sesuai dengan perancangan yang dilakukan.

4.1.2 Data Model

Dalam tahapan ini, foto tanaman yang sudah di ambil di *upload* dan di *training* menjadi data model yang nantinya akan diteruskan menjadi objek yang bisa dibaca ketika *machine learning edge detection* dijalankan.

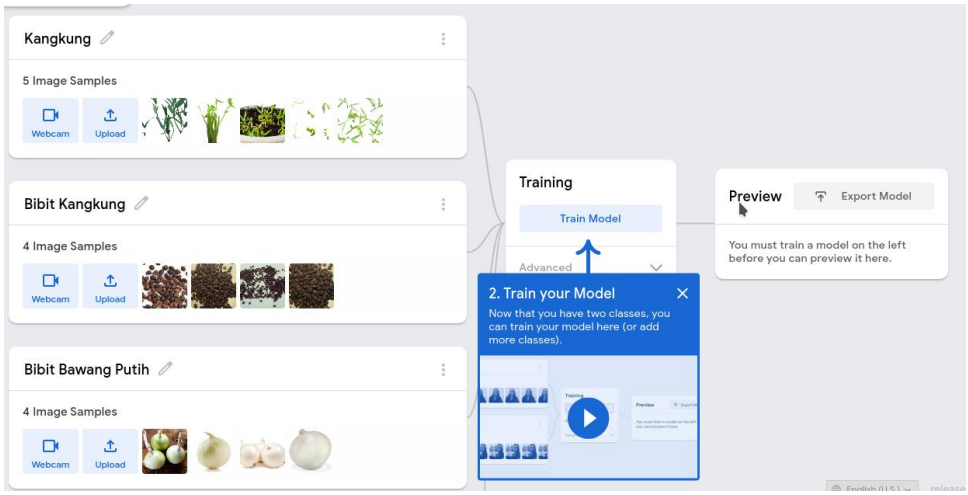


GAMBAR: 4.1 Tampilan Data Model

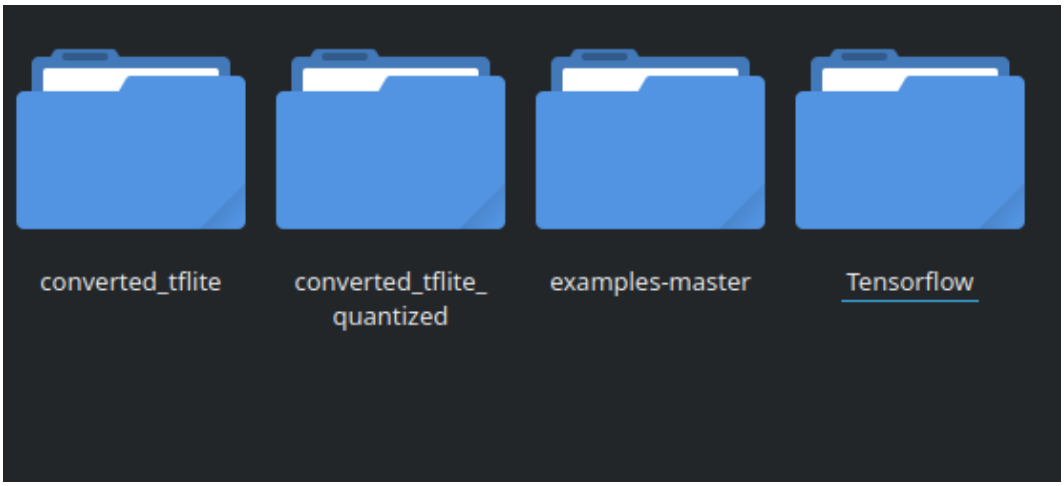
GAMBAR: 4.2 Tampilan Data Model Yang Dikumpulkan

4.1.3 Data Training

Dalam tahapan ini, data model yang di *upload* lalu diubah menjadi file xml dan *data quantization* dimana data ini merupakan perubahan dari file gambar menjadi kumpulan kode biner yang nantinya bisa diubah lagi menjadi *format file* berbentuk JPEG.



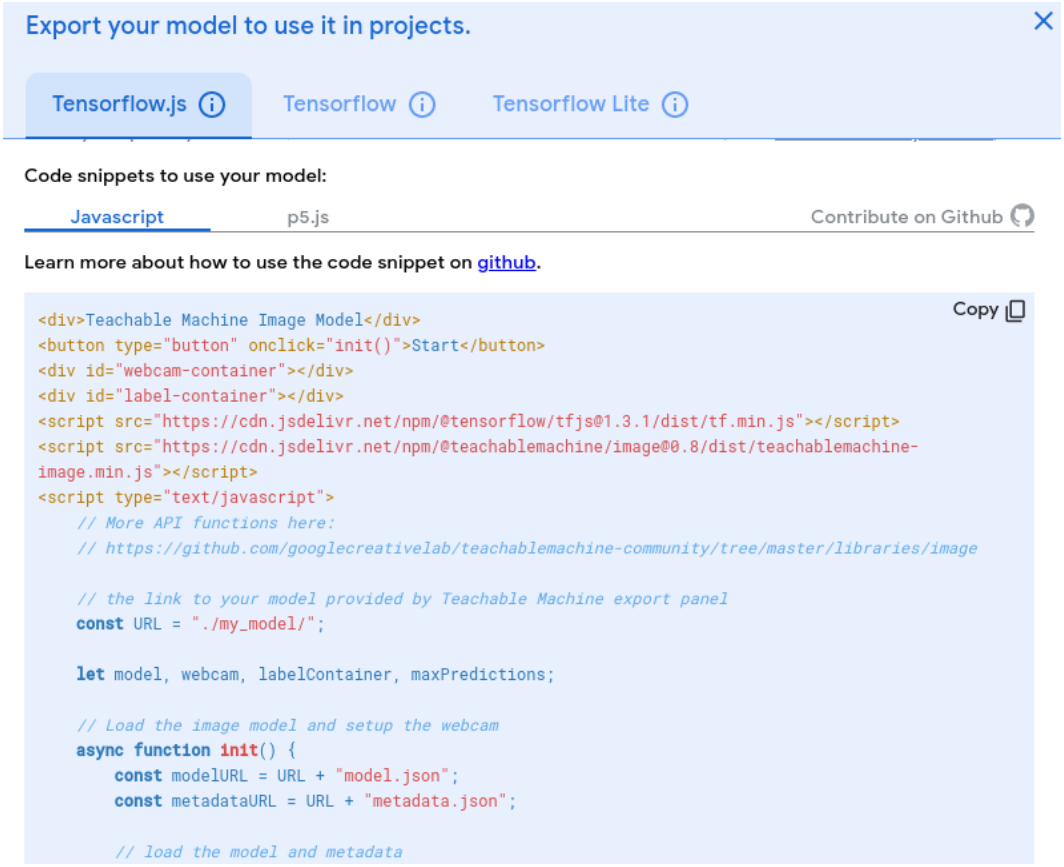
GAMBAR: 4.3 Proses Data Model yang di *Training*



GAMBAR: 4.4 File Data Model Yang Sudah Ter-*Convert*

4.1.4 *Upload Data*

Dalam tahapan ini, data model sudah di upload dan diubah menjadi bentuk *web* yang nantinya diakses oleh *user* melalui aplikasi



GAMBAR: 4.5 Data Training Yang Sudah di Convert Siap di Upload Ke Web

a. Spesifikasi Perangkat Keras dan Perangkat Lunak

Spesifikasi perabgkat keras yang digunakan dalam pembuatan aplikasi ini adalah:

TABEL:4.1 Spesifikasi Kebutuhan Perangkat Keras

Developer	
Perangkat Keras	*Processor* AMD Athlon(tm) II X2 240
	Ram 8 Gb
	Free Storage Harddisk
	Monitor

TABEL:4.2 Spesifikasi Kebutuhan Perangkat Luna**k**

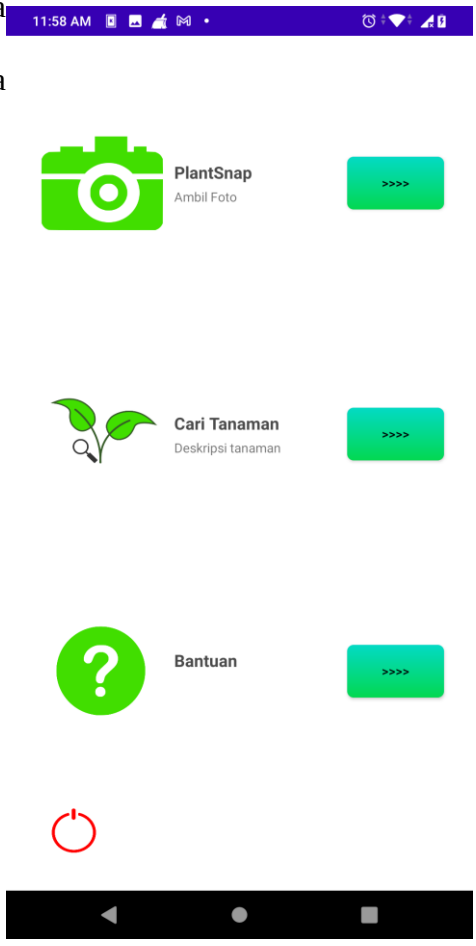
Developer	
Perangkat Lunak	Sistem Operasi Linus Kubuntu 20.0
	Android Studio 4.1.1
	Pycharm Community Edition
	Inkscape 2.0
	Browser

a. Implementasi Perancangan Antarmuka

Implementasi perancangan antarmuka yang telah dilakukan pada bab 3 adalah sebagai berikut:

1. Tampilan Halaman Utama

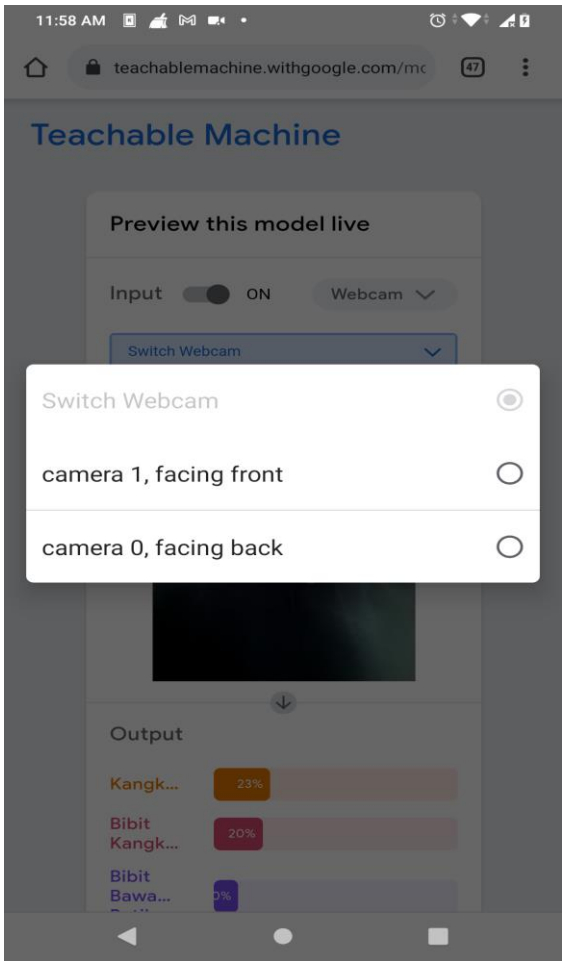
Halaman utama / *main menu* merupakan halaman pilihan fitur apa yang akan digunakan oleh *user*. Halaman utama dapat dilihat pada Gambar 4.1.



GAMBAR: 4.6 Tampilan Halaman Utama

3. Tampilan Halaman Pilihan Kamera Plantsnap

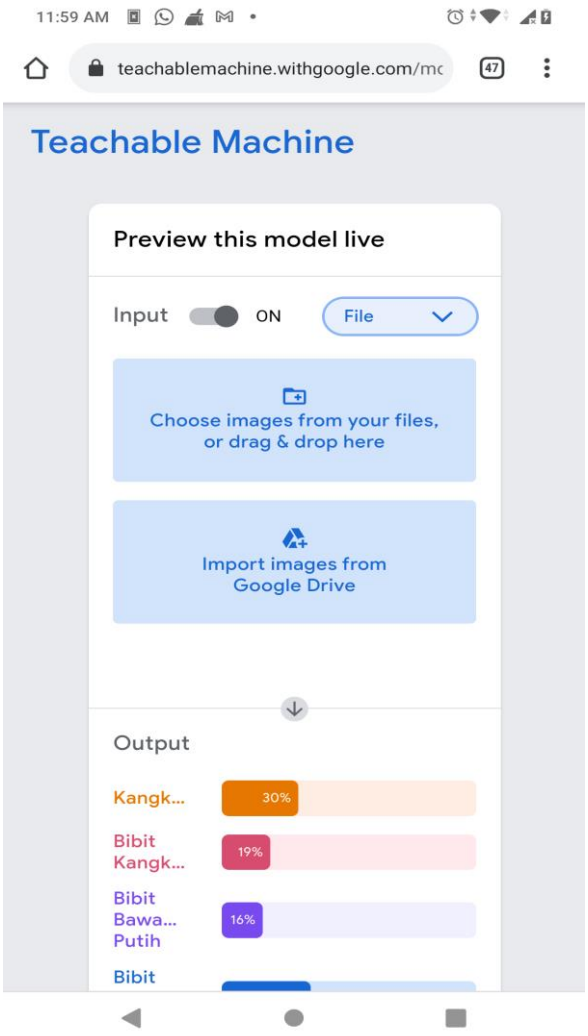
Halaman pilihan kamera *Plantsnap* adalah halaman yang membuka kamera *user* yang nantinya digunakan untuk *scanning* tanaman. Halaman pilihan kamera *Plantsnap* dapat dilihat pada Gambar 4.3.



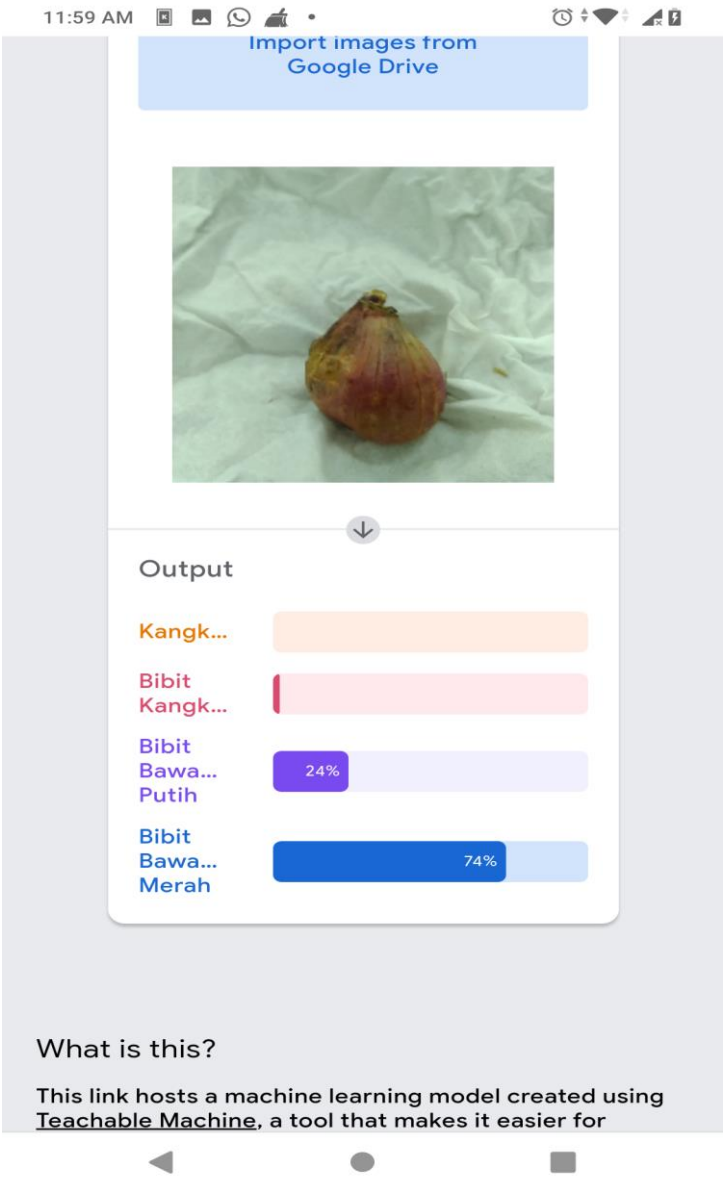
GAMBAR: 4.7. Tampilan Halaman pilihan kamera Plantsnap

4. Tampilan Halaman Pilihan Galeri Plantsnap

Halaman pilihan galeri *Plantsnap* adalah halaman yang membuka galeri *user* yang nantinya foto yang dipilih dipindai oleh aplikasi. Halaman pilihan kamera *Plantsnap* dapat dilihat pada Gambar 4.4 dan Gambar 4.5



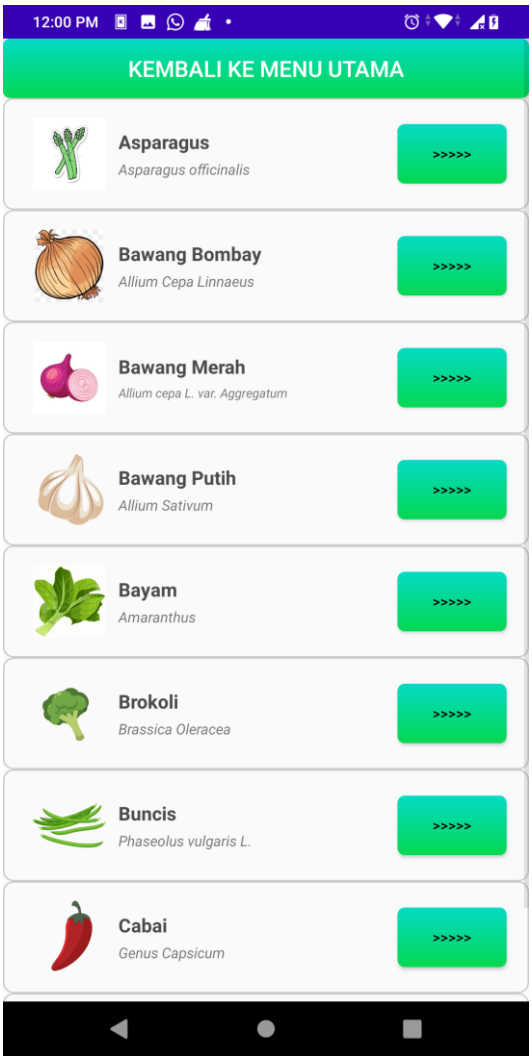
GAMBAR: 4.8. Tampilan Halaman pilihan galeri Plantsnap



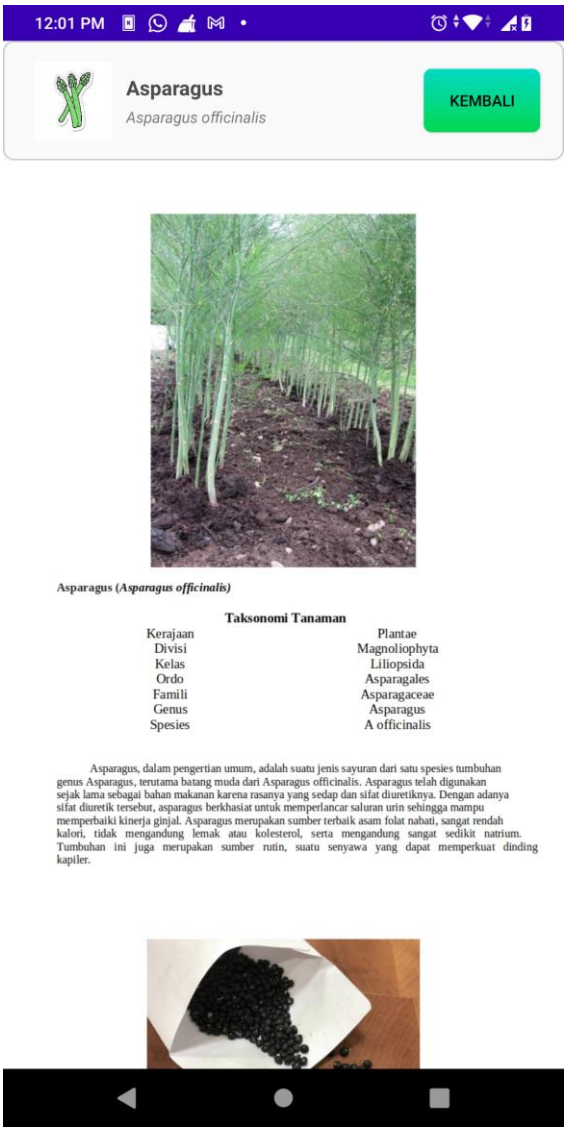
GAMBAR: 4.9. Tampilan Halaman pilihan galeri Plantsnap

5. Tampilan Halaman Cari Tanaman

Halaman cari tanaman adalah halaman yang digunakan *user* untuk mencari tanaman dimana halaman ini berisi *list* dari semua tanaman yang berisikan info lengkap tentang tanaman. Halaman cari tanaman dan halaman isi submenu dapat dilihat pada Gambar 4.6. dan Gambar 4.7.



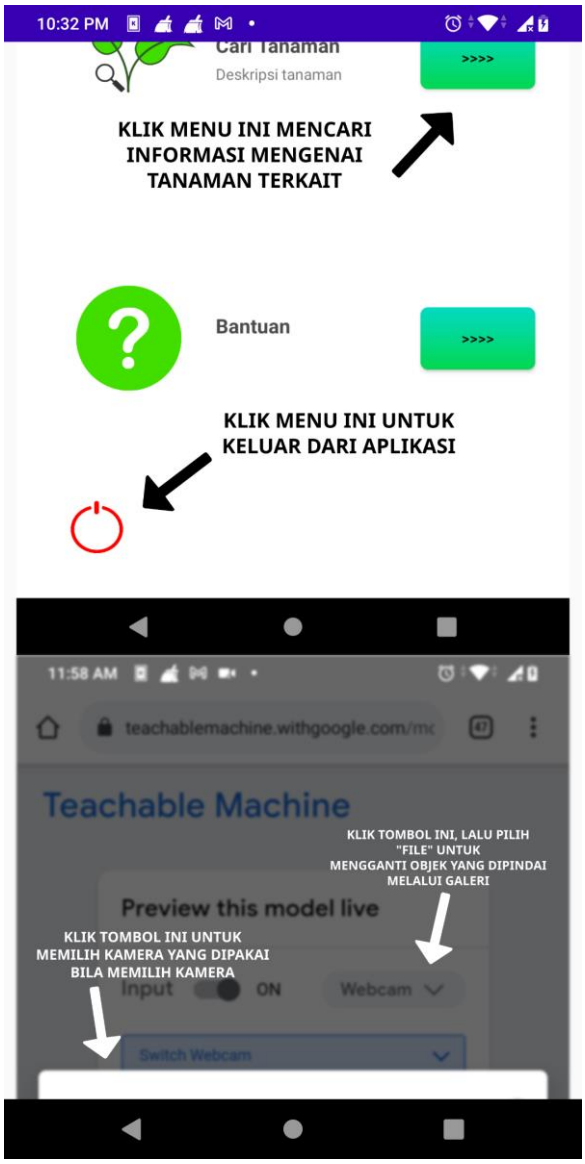
GAMBAR: 4.10. Tampilan Halaman Cari Tanaman



GAMBAR: 4.11. Tampilan Halaman Info Tanaman

6. Tampilan Halaman Bantuan

Halaman bantuan adalah informasi bagaimana *user* menggunakan aplikasi Tanam Urban. Halaman bantuan dapat dilihat pada Gambar 4.7. dan Gambar 4.8.



GAMBAR: 4.12. Tampilan Halaman Bantuan

GAMBAR: 4.13. Tampilan Halaman Bantuan

4.2. *Testing*

TABEL 4.3. menunjukan rencana pengujian

TABEL: 4.2. Tabel Rencana Pengujian

No	**Kelas Uji**	**Butir Uji**	**Jenis Pengujian**
1	Pengujian pindai objek dengan kamera depan	Validasi Layar Kamera dan *Object*	*Black Box*
2	Pengujian pindai objek melalui galeri	Validasi Data *Object*	*Black Box*
3	Pengujian pindai objek dengan kamera belakang	Validasi Layar Kamera dan *Object*	*Black Box*
4	Pengujian pindai objek melalui galeri google *drive*	Validasi Data *Object*	*Black Box*

5	Pengujian tombol kembali dari halaman Plantsnap	Validasi Tombol Aplikasi & Data Object	*Black Box*
6	Pengujian tombol cari tanaman di halaman utama	Validasi Tombol Aplikasi & Data Object	*Black Box*
7	Pengujian tombol plantsnap di halaman utama	Validasi Tombol Aplikasi & Data Object	*Black Box*
8	Pengujian tombol bantuan di halaman utama	Validasi Tombol Aplikasi & Data Object	*Black Box*
9	Pengujian tombol keluar di halaman utama	Validasi Tombol Aplikasi & Data Object	*Black Box*
10	Pengujian tombol pilih tanaman di menu cari tanaman	Validasi Tombol Aplikasi & Data Object	*Black Box*

11	Pengujian kembali di menu cari tanaman	Validasi Tombol Aplikasi & Data Object	*Black Box*
12	Pengujian kembali di menu informasi tanaman	Validasi Tombol Aplikasi & Data Object	*Black Box*

Setelah dilakukan pengujian dengan menggunakan metode *black box* maka didapatkan hasil pengujian pada Tabel berikut.

TABEL: 4.3. Tabel Hasil Pengujian Black Box

No	**Kelas Uji**	**Hasil yang diharapkan**	**Respon Sistem**	**Kesimpulan**
1	Pengujian pindai objek dengan kamera depan	Objek muncul di layar kamera depan	Aplikasi menampilkan objek di layar kamera depan, dan sistem langsung memindai objek	Valid

2	Pengujian pindai objek melalui galeri	Muncul galeri setelah button di klik, dan setelah memilih gambar, meminta konfirmasi, lalu gambar di pindai	Aplikasi memunculkan galery, dan gambar yang dipilih langsung di pindai	Valid
3	Pengujian pindai objek dengan kamera belakang	Objek muncul di layar kamera depan	Aplikasi menampilkan objek di layar kamera depan, dan sistem langsung memindai objek	Valid
4	Pengujian pindai objek melalui galeri google *drive*	Muncul Halaman Google Drive setelah button di klik, dan setelah memilih gambar, meminta konfirmasi, lalu gambar di pindai	Aplikasi memunculkan halaman Google Drive, dan gambar yang dipilih langsung di pindai	Valid
5	Pengujian tombol kembali dari halaman	Muncul tampilan halaman menu utama setelah mengklik tombol kembali	Muncul tampilan halaman menu utama setelah tombol kembali di	Valid

	Plantsnap		klik	
6	Pengujian tombol cari tanaman di halaman utama	Muncul tampilan halaman cari tanaman setelah mengklik tombol “cari tanaman”	Muncul tampilan halaman menu cari tanaman setelah tombol di klik	Valid
7	Pengujian tombol plantsnap di halaman utama	Muncul tampilan halaman plantsnap setelah mengklik tombol “plantsnap”	Muncul tampilan halaman menu plantsnap setelah tombol di klik	Valid
8	Pengujian tombol bantuan di halaman utama	Muncul tampilan halaman bantuan setelah mengklik tombol “bantuan”	Muncul tampilan halaman menu bantuan setelah tombol di klik	Valid
9	Pengujian tombol keluar di halaman utama	Keluar dari aplikasi setelah tombol keluar diklik	Aplikasi berhenti setelah tombol keluar diklik	Valid
10	Pengujian tombol pilih	Muncul tampilan halaman deskripsi	Muncul tampilan halaman informasi	Valid

	tanaman di menu cari tanaman	tanaman setelah mengklik salah satu tombol nama tanaman	tanaman setelah tombol tanaman yang dipilih di klik	
11	Pengujian kembali di menu cari tanaman	Muncul tampilan halaman menu utama setelah mengklik tombol “kembali”	Muncul tampilan halaman menu utama setelah tombol kembali di klik	Valid
12	Pengujian kembali di menu informasi tanaman	Muncul tampilan halaman menu cari tanaman setelah mengklik tombol “kembali”	Muncul tampilan halaman cari tanaman setelah tombol kembali di klik	Valid

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil pembahasan beserta penelitian yang telah dilakukan, maka dapat diambil beberapa kesimpulan, diantaranya:

1. *Machine Learning* yang dibuat dapat mengenali suatu objek tanaman berupa data *model* dengan memproses data biasa yang di *training* menjadi data *model* dan hasil pindai kamera terhadap objek semakin tinggi akurasinya bila data *model*-nya banyak.
2. Metode *edge detection* dapat mengatasi perbedaan dengan mengklasifikasi objek yang dibedakan dari bentuk, warna, pencahayaan dan posisinya. Dikarenakan data *modelnya* sudah ada, maka proses pemindaian dan pengenalan objek dapat berjalan dalam waktu yang singkat dengan akurasi yang tinggi.
3. *Machine Learning* pada penelitian ini bersifat *Unsupervised Learning* sehingga data model yang terpindai tidak dijadikan parameter data model tambahan karena sistem hanya mengandalkan data *model* yang sudah disiapkan saja sebelumnya.

5.2 Saran

Kesimpulan yang ada sebelumnya menciptakan beberapa saran yang bertujuan untuk meningkatkan performa, fungsi, dan efisiensi dibanding aplikasi yang sudah dibuat sebelumnya.

1. Aplikasi ini hanya bisa mendeteksi objek tanaman yang berada di masa generatif tanaman sehingga tanaman yang berada dimasa vegetatif tidak bisa terdeteksi, kecuali untuk tanaman tertentu yang tidak terlalu mempunyai perbedaan diantara kedua fase tersebut.
2. *Data model* yang sudah diterapkan di aplikasi tidak mencangkup seluruh tanaman, dikarenakan kendala media *developer* yang membutuhkan spesifikasi komputer yang lebih besar sehingga hanya mengambil beberapa *data model* saja untuk efisiensi pengerjaan.
3. Aplikasi akan lebih baik lagi bila algoritma *machine learning unsupervised learning* ini dipadukan dengan *supervised learning* sehingga tidak perlu menambahkan *data model* secara manual melainkan sudah otomatis ditambahkan sendiri oleh sistem.

DAFTAR PUSAKA

Dian. (2016): *Game Edukasi Berbasis Android Sebagai Media Pembelajaran Untuk Anak Usia Dini* (Studi Kasus: Fakultas Teknologi Informasi Universitas Merdeka Pasuruan), 47.

Fathul Wahid. (2004): *Metodologi Penelitian Sistem Informasi: Sebuah Gambaran Umum* (Studi Kasus: Fakultas Teknologi Industri,Universitas Islam Indonesia), 75.

KBBI. (1998): *Aplikasi* (Ebta Setiawan).

Lexy J. Moleong. (2002): *Metodologi Penelitian Kualitatif* (Penelitian : Remaja Rosdakarya Perpustakaan BPPSDMK), 15.

Jogiyanto, H.M., 1999. *Pengenalan komputer*. Andi Offset, *Yogyakarta*.

Narimawati U. (2008): *Metodologi Penelitian Kualitatif dan Kuantitatif* (Universitas Komputer Indonesia).

Gresse von Wangenheim, C., Marques, L. S., & Hauck, J. C. R. (2020, August 28). *Machine Learning for All – Introducing Machine Learning in K-12*

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, Rif A. Saurous. (2017). *Tensorflow Distributions* (Google, Columbia University),2.

Pressman, Roger S.(2017). *Rekayasa Perangkat Lunak– Buku Satu, Pendekatan Praktisi (Edisi 9)*. Yogyakarta: Andi

A.S, Rosa dan Shalahuddin, M. *Rekayasa Perangkat Lunak Terstruktur dan Beroientasi Objek.* Bandung: Imformatika, 2016.

Samira Pouyanfar. (2018). *A Survey on Deep Learning: Algorithms, Techniques, and Applications*. Florida International University.

Andi Juansyah. (2016). *Pembangunan Aplikasi Child Tracker Berbasis Assisted – Global Positioning System (A-GPS) dengan Platform Android*. Universitas Komputer Indonesia. 2.

Bolei Zhou. (2018). *Interpreting Deep Visual Representations via Network Dissection*. Massachusetts Institute Of Technology. 1.

Rinaldi Munir. (2019). *Pengantar Pengolahan Citra*. Program Studi Teknik Informatika Institut Teknologi Bandung.

Pulung Nurtantio A., Sutojo T., Muljono. (2017). *Pengolahan Citra Digital.* Universitas Dian Nuswantoro, Semarang.

Pulung Nurtantio A., Sutojo T., Muljono. (2017). *Pengolahan Citra Digital*. Universitas Dian Nuswantoro, Semarang.

Mei Wang, Weihong Deng. (2020). *Deep Face Recognition: A Survey*. Beijing University of Posts and Telecomunication.

James Hays, Unaiza Ahsan, Chen Sun, Irfan Essa. (2017). *Complex Event Recognition from Images with Few Training Examples*. Georgia Institute of Technology & University of Southern California.

David A. Forsyth, Jean Ponce. (2002). *Computer Vision: a modern Approach.* University of California, Berkeley. Vol. 1 No. 1 dan Vol. 2 No. 1.

Sitompul, S.M dan B. Guritno (1995). *Analisis Pertumbuhan Tanaman.* Yogyakarta: Gadjah Mada University Press.

Carleo, G., Cirac, I., Cranmer, K., Daudet, L., Schuld, M., Tishby, N., ... & Zdeborová, L. (2019). *Machine learning and the physical sciences. Reviews of Modern Physics*, 91(4), 045002.

Ridlo, I. A. (2017). Panduan Pembuatan *Flowchart*. Fakultas Kesehatan Masyarakat, Departemen Administrasi Dan Kebijakan Kesehatan.