SKRIPSI

Sistem Pengendalian Pemberian Jumlah Pakan Berdasarkan Jumlah Populasi dan Rata-rata Sampel Berat Ikan

(Studi Kasus : BPBAP Takalar)

Disusun dan diajukan oleh:

AYU LESTARI RAMADANI

D42116005

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

LEMBAR PENGESAHAN SKRIPSI

SISTEM PENGENDALIAN PEMBERIAN JUMLAH PAKAN BERDASARKAN JUMLAH POPULASI DAN RATA-RATA SAMPEL BERAT IKAN (STUDI KASUS : BPBAP TAKALAR)

Disusun dan diajukan oleh

AYU LESTARI RAMADANI
D42116005

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin pada tanggal 9 Desember 2022 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Eng. Muhammad Niswar, S.T., M.IT.	Dr. Amil Ahmad Ilham, S.T., M.IT.
Nip. 197309221999031001	Nip. 19731010199821001

Ketua Program Studi,

Dr. Indrabayu, S.T., M.T., M.Bus.Sys
Nip. 19750716 200212 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Ayu Lestari Ramadani

NIM : D42116005

Departemen : Teknik Informatika

Jenjang : S1

Menyatakan dengan ini karya tulisan saya berjudul:

SISTEM PENGENDALIAN PEMBERIAN JUMLAH PAKAN BERDASARKAN JUMLAH POPULASI DAN RATA-RATA SAMPEL BERAT IKAN (STUDI KASUS : BPBAP TAKALAR)

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilalihan tulisan orang lain bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 09 Desember 2022

Yang menyatakan,

Ayu Lestari Ramadani

KATA PENGANTAR

Segala puji dan syukur kita panjatkan kepada Allah SWT yang telah memberikan rahmat dan karunia-Nya kepada penulis, sehingga penulis dapat menyelesaikan penyusunan tugas akhir yang berjudul **"Sistem Pengendalian Pemberian Jumlah Pakan Berdasarkan Jumlah Populasi dan Rata-rata Sampel Berat Ikan (Studi Kasus : BPBAP Takalar)"** dengan baik. Selawat dan salam tidak lupa kita panjatkan ke hadirat Nabi Besar Muhammad SAW. Adapun tujuan dari penyusunan tugas akhir ini sebagai salah satu syarat untuk dapat menyelesaikan Program Sarjana (S1) di Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Selama proses penyusunan tugas akhir ini, penulis melalui berbagai hambatan tetapi pada akhirnya penulis dapat menyelesaikan penyusunan tugas akhir ini. Maka dari itu, penulis ingin menyampaikan ucapan terima kasih kepada semua pihak yang telah membantu dalam penyusunan skripsi ini terutama kepada:

1. Kedua orang tua dan keluarga, Bapak Dahlan, Ibu Suarfine dan Adik Rafli, yang senantiasa memberikan dukungan dan doa serta tiada hentinya mengingatkan penulis untuk segera menyelesaikan tugas akhir ini.
2. Bapak Dr. Eng. Muhammad Niswar, S.T., M.IT. selaku dosen pembimbing I dan Bapak Dr. Amil Ahmad Ilham, S.T., M.IT. selaku dosen pembimbing II, yang selalu menyediakan waktu, tenaga dan pikirannya sehingga penulis mendapatkan ilmu berlimpah pada perkuliahan dan proses bimbingan yang dapat penulis terapkan dalam penyusunan Tugas Akhir hingga pada dunia kerja.

3. Ibu Anugrayani Bustamin, S.T., M.T. selaku dosen yang senantiasa memberikan ilmu serta mengarahkan penulis baik dilingkungan kampus maupun di luar kampus.
4. Bapak Dr. Ir. Zahir Zainuddin, M.Sc. selaku penanggung jawab penulis selama menjadi asisten Dasar Pemrograman Komputer, Bapak Dr. Adnan, S.T., M.T. selaku Kepala Lab *Internet of Things*, Bapak Dr. Eng. Ady Wahyudi Paundu, S.T., M.T. selaku dosen pembimbing Kerja Praktik penulis, Bapak Prof. Dr. Ir. Andani, M.T. Serta Segenap Dosen dan Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin, yang telah banyak membantu penulis selama masa perkuliahan.
5. Amila Saliha Mustarin selaku sahabat penulis yang dengan sabar menemani, mengingatkan dan menerima segala keluh-kesah penulis sehingga penulis dapat menyelesaikan penyusunan Tugas Akhir ini.
6. Muh. Fathin Abdillah Halik, Muh. Raedi Radifan, Tedi Setiady Pakiding Ba'Ka', Asri Oktianawati, Cici Purnamasari dan Irham Sahbana selaku jajaran pengurus OKIF FT-UH Masa Bakti 2019 dari semasa kepengurusan hingga saat ini saling bahu-membahu dalam menyelesaikan segala urusan Bersama.
7. Muh. Yusril Adri, Muh. Fachrul Alam, Lutfi Qadri, Putri Angriani, Malyana Ariani, Safina, Fadhilah Istiqomah, Tuti Amalia, Nurul Musfirah, Dhinda Fitri Wiludjeng, Diki Siswanto, Muh. Khaeril Syam, Nishrina Nurul Amirah, Patricia Palada, Afifah Ilham, Ismayanti, Rizky Alfiansyah, Muhammad Musyawir, Andi Muh. Agung, Andi Wijaya, Kevin Jordi, Ibnu Gaury serta

Saudara-saudari IGNITER 16 yang selalu menyemangati dan membantu penyelesaian skripsi ini serta mengisi hari-hari semasa kuliah menjadi sangat menyenangkan.

8. KalukuPintaOfficial, Team Gammara09, EASTCO serta semua pihak atas dukungan dan bantuannya yang tidak dapat penulis tuliskan satu persatu.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna dikarenakan terbatasnya pengalaman dan pengetahuan yang dimiliki penulis. Oleh karena itu, penulis mengharapkan segala bentuk saran serta masukan bahkan kritik yang membangun dari berbagai pihak. Semoga skripsi ini dapat bermanfaat bagi para pembaca dan semua pihak.

Makassar, November 2022

Penulis

ABSTRACT

Tilapia (*Oreochromis niloticus*) is known as chicken fish, widely cultivated in almost all countries of the world. Its global production reached 6.3 million tons in 2018 and has experienced a significant increase in production in recent years. The increasing demand for tilapia commodities correlates with the intensification of aquaculture which will affect the need for feed as one of the limiting factors in growth. The cost of feed needs for farmed fish is about 60-70% of the total cost of production. However, currently feeding at BPBAP Takalar is still carried out conventionally with the feed given still based on the estimated weight and number of fish populations, of course this is less efficient regarding controlling the amount of feed. Therefore, in this study, we designed a system for calculating the number of tilapia populations using object detection method and assembling a feeder system so that it can calculate the need for the amount of feed and set a feeding schedule, especially for fingerlings. In this study process using 98 pictures with fish sizes in figures 3-7 cm. The algorithm for object detection uses 4th version of You Only Look Once (YOLO) and Keras model. For the feeder assembly process, it uses the ESP32-Cam as a microcontroller and to take pictures while to control feed production using the MG966R Servo Motor. The results of this study are an fingerlings detection system using YOLOv4 with *a* precision 0.99, recall 0.89 and an accuracy of 88.3% while with Keras model precision 0.98, recall 0.84 and accuracy of 82.3%

Keywords: Tilapia Fingerlings, Feeding, *You Only Look Once* (YOLO), ESP32-Cam, *Flask*

ABSTRAK

Ikan nila (*Oreochromis niloticus*) dikenal dengan sebutan *chicken fish*, banyak dibudidayakan di hampir seluruh Negara di dunia. Produksinya secara global mencapai 6.3 juta ton pada tahun 2018 serta mengalami peningkatan produksi yang signifikan dalam beberapa tahun terakhir. Permintaan komoditas ikan nila yang semakin meningkat berkorelasi terhadap intensifikasi budidaya yang akan mempengaruhi kebutuhan pakan sebagai salah satu faktor pembatas dalam pertumbuhan. Biaya kebutuhan pakan untuk ikan budidaya sekitar 60-70% dari total biaya produksi. Namun saat ini pemberian pakan di BPBAP Takalar masih dilakukan secara konvensional dengan pakan yang diberikan masih berdasarkan dari perkiraan berat serta perkiraan jumlah populasi ikan, tentu saja hal ini kurang efisien terkait pengendalian jumlah pakan. Maka dari itu pada penelitian ini merancang sistem penghitungan jumlah populasi ikan nila menggunakan metode *object detection* dan merakit sistem *feeder* sehingga dapat menghitung keperluan jumlah pakan serta mengatur jadwal pemberian pakan terkhusus pada benih nila. Pada proses penelitian ini menggunakan 98 gambar dengan ukuran ikan pada gambar 3-7 cm. Algoritma untuk melakukan *object detection* menggunakan *You Only Look Once* (YOLO) versi ke-4 dan juga *saved model* Keras. Untuk proses perakitan *feeder* menggunakan ESP32-Cam sebagai mikrokontroler dan alat untuk mengambil gambar sedangkan untuk mengontrol pengeluaran pakan menggunakan Servo Motor MG966R. Hasil dari penelitian ini yaitu sistem deteksi benih nila menggunakan YOLOv4 dengan nilai *precision* 0.99, *recall* 0.89 dan akurasi 88.3% sedangkan dengan model Keras *precision* 0.98, *recall* 0.84 dan akurasi 82.3%

Kata Kunci: Benih Nila, Pemberian Pakan, *You Only Look Once* (YOLO), ESP32-*Cam*, *Flask*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada proses pembudidayaan ikan nila terdapat beberapa faktor yang mempengaruhi keberlangsungan budidaya tersebut salah satunya yaitu terkait pakan yang diberikan (Yanuar, 2017). Pakan merupakan unsur terpenting dalam menunjang pertumbuhan dan keberlangsungan hidup ikan. Pemilihan pakan yang efisien harus sesuai dengan jenis, jumlah serta jadwal pemberian pakan.

Hal-hal yang perlu diperhatikan untuk pemberian pakan yang sesuai yaitu jenis pakan, jumlah populasi ikan dan jadwal pemberian pakan(Anggriani, 2020). Saat ini pakan yang diberikan hanya berdasarkan dari perkiraan berat serta perkiraan jumlah populasi ikan. Tentu saja hal ini tidaklah efisien, terlebih lagi karena ikan nila termasuk ke dalam golongan omnivora (pemakan segalanya). Kekurangan atau kelebihan jumlah pakan yang diberi dapat berdampak pada keberlangsungan hidup ikan nila ke depannya.

Salah satu penyakit yang dapat menyerang ikan nila yaitu *Motile Aeromonad Septicaemia* yang disebabkan oleh infeksi *Aeromonas hydrophila* (AlYahya, 2018). Infeksi ini biasanya berkaitan dengan kondisi, antara lain stres karena kepadatan atau malnutrisi. Jumlah pakan yang berlebih juga dapat menjadi racun amonia pada ikan nila. Maka dari itu, perlu adanya sistem yang dapat mengendalikan jumlah pemberian pakan, berdasarkan

dengan perhitungan jumlah ikan serta berat ikan dengan akurat.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan di atas, maka rumusan masalah pada tugas akhir ini adalah:

1. Bagaimana ketepatan mendeteksi ikan dengan menggunakan *object detection*?
2. Bagaimana merancang *feeder* sebagai *prototype* untuk sistem pengendalian jumlah pakan?

1.3 Tujuan Penelitian

1. Membuat sistem yang dapat melakukan deteksi objek benih nila pada akuarium.
2. Membuat rancangan *feeder* sebagai *prototype* yang dapat mengendalikan jumlah pakan berdasarkan jumlah populasi dan rata-rata sampel berat ikan.

1.4 Manfaat Penelitian

1. Untuk mengetahui jumlah ikan dalam satu akuarium dengan perhitungan yang lebih akurat.
2. Untuk memberikan jumlah pakan yang sesuai kepada benih nila yang sedang dibudidayakan.
3. Untuk mengendalikan jumlah pakan yang akan diberikan ke depannya.

1.5 Batasan Masalah Penelitian

1. Objek penelitian adalah benih ikan nila salin di BPBAP Takalar.
2. Menghitung jumlah benih nila dengan menggunakan metode *object detection.*
3. Menghitung berat benih nila setiap seminggu sekali.
4. Menghitung jumlah pakan dan pemberian pakan secara otomatis

1.6 Sistematika Penelitian

Untuk memberikan gambaran singkat mengenai isi tulisan secara keseluruhan, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu:

BAB I PENDAHULUAN

Bab ini menguraikan secara umum, mengenai hal yang menyangkut latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini membahas terkait teori-teori yang berhubungan dengan penelitian dan menjadi dasar acuan pada penelitian ini. Secara garis besar yaitu tata cara menghitung jumlah pakan untuk benih ikan, metode dalam pembuatan sistem *object detection* serta metode-metode yang digunakan selama penelitian.

BAB III METODOLOGI PENELITIAN

Bab ini mencakup uraian tentang metode penelitian mulai dari

tahapan penelitian, waktu dan lokasi penelitian, instrumen penelitian, proses pengambilan data, perancangan sistem, penerapan algoritma hingga tahapan analisis kerja sistem.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menguraikan hasil penelitian dan pembahasan terkait pengolahan data yang telah dilakukan disertai rangkuman hasil penelitian yang telah dilakukan.

BAB V PENUTUP

Bab ini mencantumkan kesimpulan dari hasil penelitian serta saran-saran yang dapat membantu dalam proses pengembangan sistem selanjutnya.

BAB II

TINJAUAN PUSTAKA

2.1 Ikan nila Salin

Ikan nila (*Oreochromis niloticus*) dikenal dengan sebutan *chicken fish*, banyak dibudidayakan di hampir seluruh Negara di dunia, dan produksinya secara global mencapai 6.3 juta ton pada tahun 2018 dan Indonesia merupakan produsen terbesar kedua ikan Nila dunia (Suhermanto et al. 2019; Fitzsimmons 2018). Ikan nila dapat beradaptasi dengan lingkungan budidaya, sistem budidaya skala tradisional, semi maupun intensif, mempunyai nilai ekonomis tinggi, dan tidak terpengaruh fluktuasi harga pasar, penyumbang protein hewani, serta mengalami peningkatan produksi yang signifikan dalam beberapa tahun terakhir (Wang dan Lu 2016).

Permintaan komoditas ikan nila yang semakin meningkat berkorelasi terhadap intensifikasi budidaya yang akan mempengaruhi kebutuhan pakan sebagai salah satu faktor pembatas dalam pertumbuhan. Biaya kebutuhan pakan untuk ikan budidaya sekitar 60-70% dari total biaya produksi, maka pengembangan pakan dengan bahan baku lokal yang berkelanjutan diperlukan, dan menjadi tantangan bagi pembudidaya (Sarker et al. 2018).

Budidaya ikan dengan sistem tradisional plus semi intensif maupun intensif banyak menggunakan pakan pabrikan jenis pelet dan bergantung pada tingkatan stadia. Pakan pelet disusun berdasarkan bahan baku dan penghitungan komposisi yang tepat sehingga fungsi pakan untuk meningkatkan performa pertumbuhan ikan lebih optimal. Pakan merupakan

salah satu kendala utama yang sering dialami pembudidaya. Kualitas dan kontinuitas yang berkorelasi dengan harga menyebabkan komponen biaya pakan menyumbang 60–70% total biaya produksi (Naseem et al. 2021).

2.2 Pakan Benih Nila

Pakan merupakan salah satu pokok penunjang yang berperan meningkatkan Pertumbuhan organisme sehingga sangat penting memperhatikan kualitas pakan dan kuantitas pakan yang akan di berikan pada ikan Nila salin. Namun bila kualitas pakan sudah baik maka yang harus diperhatikan kuantitas dalam pemberian pakan, dosis yang tepat akan mempengaruhi pertumbuhan terutama berat ikan Nila salin, penggunaan dosis yang tepat juga akan berdampak pada hasil kegiatan usaha budidaya perikanan.

Dosis pemberian pakan untuk ikan Nila berkisar 3-7 % dari berat biomassa. Dosis pemberian pakan serta frekuensi pemberian yang berlebihan akan mengurangi nilai dari konversi pakan dan efisiensi pakan, sehingga penting penentuan dosis pemberian pakan yang sesuai dengan kebutuhan ikan agar tumbuh optimal namun dalam segi ekonomi masih dapat terkontrol. Selain itu, ikan nila salin di beri pakan buatan yaitu berupa pelet. pakan tersebut diperlukan pada saat ukuran ikan nila salin saat fase benih.

Upaya yang dapat dilakukan untuk meningkatkan produksi benih ikan nila salin yaitu dengan mengetahui cara pemeliharaan dan Pemberian pakan

yang baik tepat waktu, tepat ukuran dan tepat dosis, sehingga dapat meningkatkan laju pertumbuhan dan kelangsungan hidup (Angriani, 2020).

2.3 Deep Learning

Deep Learning adalah salah satu cabang dari ilmu pembelajaran mesin (*Machine Learning*) yang terdiri algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linier yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam *deep learning* dapat digunakan baik untuk kebutuhan pembelajaran terarah (*supervised learning*), pembelajaran tak terarah (*unsupervised learning*) dan semi-terarah (*semi-supervised learning*) dalam berbagai aplikasi seperti pengenalan citra, pengenalan suara, klasifikasi teks, dan sebagainya. *Deep learning* disebut sebagai *deep* (dalam) karena struktur dan jumlah jaringan saraf pada algoritmanya sangat banyak bisa mencapai hingga ratusan lapisan. *Deep learning* adalah salah satu jenis algoritma jaringan saraf tiruan yang menggunakan *metadata* sebagai *input* dan mengolahnya menggunakan sejumlah lapisan tersembunyi (*hidden layer*) transformasi non-linier dari data masukan untuk menghitung nilai *output*. Algoritma pada *deep learning* memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis. Hal ini berarti algoritma yang dimilikinya secara otomatis dapat menangkap fitur yang relevan sebagai keperluan dalam pemecahan suatu masalah. Algoritma semacam ini sangat penting dalam sebuah kecerdasan buatan karena mampu mengurangi beban pemrograman dalam memilih fitur

yang eksplisit. Algoritma ini dapat digunakan untuk memecahkan permasalahan yang perlu pengawasan (*supervised*), tanpa pengawasan (*unsupervised*), dan semi terawasi (*semi supervised*).

Dalam jaringan saraf tiruan tipe *deep learning* setiap lapisan tersembunyi bertanggung jawab untuk melatih serangkaian fitur unik berdasarkan *output* dari jaringan sebelumnya. Algortima ini akan menjadi semakin kompleks dan bersifat abstrak ketika jumlah lapisan tersembunyi (*hidden layer*) semakin bertambah banyak. Jaringan saraf yang dimiliki oleh *deep learning* terbentuk dari hierarki sederhana dengan beberapa lapisan hingga tingkat tinggi atau banyak lapisan (*multi layer*). Berdasarkan hal itulah *deep learning* dapat digunakan untuk memecahkan masalah kompleks yang lebih rumit dan terdiri dari sejumlah besar lapisan transformasi non-linier (Putri, 2020).

2.4 Sistem Tersemat

Sistem tersemat (*embedded system*) adalah sistem komputer tujuan-khusus dengan seluruh bagian yang diperlukan dimasukkan menjadi satu dalam perangkat tersebut. Kata tersemat (*embedded*) menunjukkan bahwa sistem ini merupakan perangkat lengkap termasuk bagian sistem mekanik dan elektrik. Sebuah sistem tertanam memiliki kebutuhan tertentu dan melakukan tugas yang telah diset sebelumnya, tidak seperti komputer pribadi serba guna. Contoh sistem atau aplikasinya antara lain adalah instrumentasi medis, *process control*, *automated vehicles control*, dan perangkat

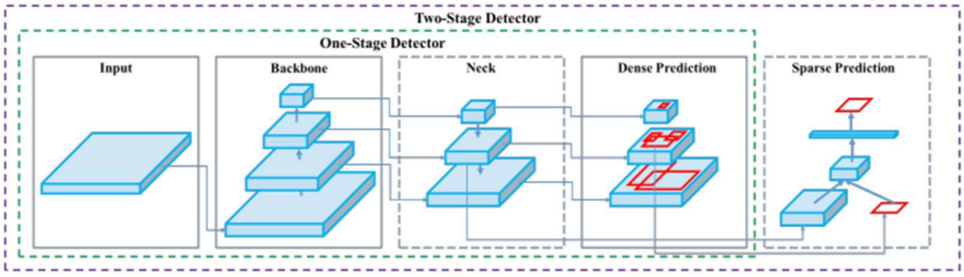
komunikasi. Sistem tersemat biasanya diimplementasikan dengan menggunakan mikrokontroler.

2.5 *Internet of Things* (IoT)

Internet of Things (IoT) adalah struktur di mana objek orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke komputer. *Internet of Things* merupakan perkembangan keilmuan yang sangat menjanjikan untuk mengoptimalkan kehidupan berdasarkan sensor cerdas dan peralatan pintar yang bekerja sama melalui jaringan internet.

2.6 Object detection

Object detection merupakan salah satu metode dari *image processing* yang digunakan untuk menentukan keberadaan objek tertentu di dalam suatu citra. Proses *object detection* umumnya mendeteksi citra dengan melakukan



Gambar 2.1 *Object Detector* (Bochkovskiy, 2020)

pembacaan fitur-fitur dari seluruh objek pada citra *input*. Proses *object detection* dapat dilihat pada gambar berikut :

Object detection saat ini umumnya terdiri dari dua bagian, bagian pertama yaitu *backbones* yang merupakan *pre-trained* dari *ImageNet* dan *head* yang digunakan untuk memprediksi kelas dan pembatas *bounding boxes*. Untuk *detector* yang dijalankan menggunakan GPU, *backbone* yang digunakan bisa berupa VGG, ResNet, ResNeXt,atau DenseNet. Sedangkang untuk *detector* yang dijalankan menggunakan *CPU Platform* umumnya *backbones*

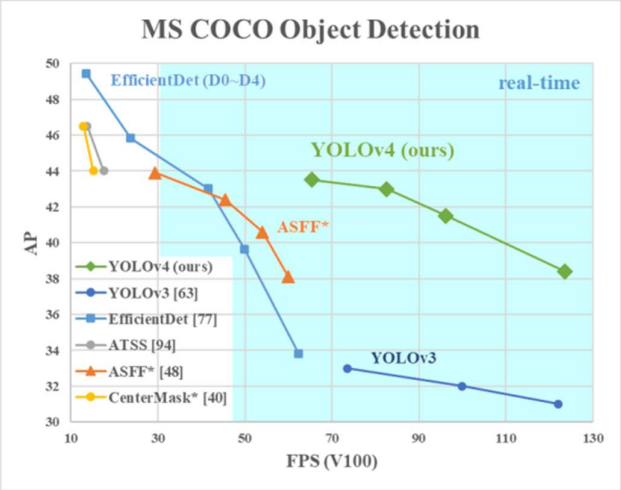
2.7 YOLO

YOLO atau *You Only Look Once* merupakan sebuah algoritma yang menggunakan *Convolutional neural networks* untuk pendeteksian objek. YOLO merupakan salah satu algoritma pendeteksi objek tercepat bila dibandingkan dengan algoritma lain (Kelvin, 2020).

YOLO merupakan salah satu metode yang memprediksi *bounding box* dan probabilitas pada kelas secara langsung dalam sekali evaluasi. Sistem deteksi pada YOLO sangat mudah yaitu apabila telah mendapatkan *input* citra, sistem akan melakukan perubahan terhadap citra menjadi 416 x 416. Setelah itu diproses dengan *single Convolutional neural network* dan dilakukan *non-max suppression* agar menghasilkan *bounding box* yang menentukan kelas dari tiap objeknya. *Non-max suppression* memiliki peran penting dalam memilih *bounding box* dengan nilai *confidence* yang lebih

tinggi karena dalam algoritma deteksi objek pasti ada kemungkinan terdapat lebih dari satu *bounding box* (Kusuma, Usman, & Saidah, 2021).

YoloV4 adalah pengembangan dari versi sebelumnya, yaitu YoloV3. Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao mengusulkan algoritma YoloV4 dengan perubahan signifikan dari versi sebelumnya dan akurasi yang jauh lebih baik yaitu dengan peningkatan AP (*Average Precision*) dan FPS (*Frame Per Second*) pada YoloV4 dengan masing-masing 10% dan 12% (Bochkovskiy, Wang, & Liao, 2020). Perbandingannya dapat dilihat pada **Gambar 2.2**.

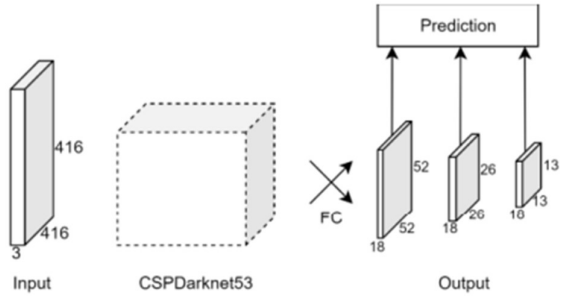


Gambar 2.2 Perbandingan YOLOv4 dan objek detektor lain (Bochkovskiy, 2020)

YOLOv4 menambahkan sejumlah implementasi tambahan yang meningkatkan akurasi dan efisiensi pendeteksian seperti *Weighted Residual Connection* untuk mengombinasikan residu dari lapisan neural *network* secara efektif dan efisien. *Cross Stage Partial Connections* untuk memisahkan *feature map* dari lapisan dasar dengan menyalinnya dan mengirim satu salinan melalui *dense block* dan mengirimkan yang lain

langsung ke tahap berikutnya, sehingga mengurangi komputasi di DenseNet dan meningkatkan pembelajaran dengan meneruskan versi *feature map* yang sebelumnya. *Cross Mini-Batch Normalization* untuk memungkinkan *training* pada satu GPU, karena sebagian besar teknik normalisasi *batch* memanfaatkan beberapa kekuatan GPU. *Self Adversarial Training*, mengaburkan wilayah gambar yang paling bergantung pada jaringan untuk memaksa jaringan mempelajari fitur-fitur baru. *Mosaic Data Augmentation*, menggabungkan empat gambar *training* bersama-sama, yang membantu model dalam belajar menemukan objek yang lebih kecil dan kurang fokus pada lingkungan yang tidak langsung berada di sebelah objek. Mish *Activation*, Fungsi aktivasi ini digunakan untuk mendorong sinyal ke kiri & kanan, yang tidak mungkin dilakukan dengan fungsi seperti ReLU. Mish memberikan hasil empiris yang lebih baik jika dibandingkan dengan ReLU, Swish & Leaky ReLU. Juga berbagai teknik Augmentasi Data berperilaku konsisten dengan penggunaan Mish. *DropBlock regularization* adalah teknik regularisasi untuk mengatasi *over-fitting*. Ini menjatuhkan blok piksel. Ia bekerja pada lapisan *Convolution,* tidak seperti putusnya *pixel* yang tidak bekerja pada lapisan ini. *CIoU loss*, sebuah fungsi *loss* yang mencapai

konvergensi lebih cepat dan akurasi yang lebih baik pada permasalahan regresi *bounding box* (Solawetz, 2020).



Gambar 2.3 Arsitektur sederhana YOLOv4 (Dhiaegana & Munir, 2020)

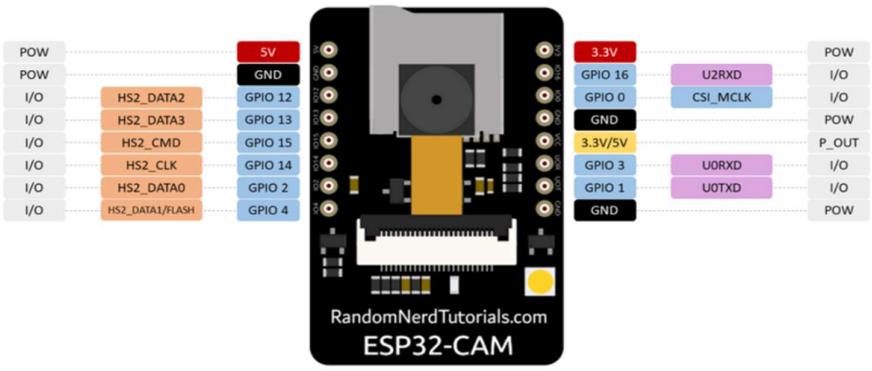
Pada **Gambar 2.3**, dapat dilihat arsitektur yang sangat sederhana dari YOLOv4. Arsitektur ini menggunakan arsitektur *backbone* CNN CSPDarknet53. Arsitektur ini berisi 162 lapisan. Masukan atau *input* merupakan citra yang sudah di *resize* sesuai ukuran yang diatur pada konfigurasi. Selanjutnya, lapisan *input* dimasukkan ke dalam arsitektur *backbone* CSPDarknet53 (Dhiaegana & Munir, 2020).

2.8 ESP32-Cam

Modul *Al-Thinker* ESP32-CAM dilengkapi dengan chip ESP32-S, kamera OV2640 berukuran sangat kecil dan slot kartu *micro SD*. Slot kartu *micro SD* dapat digunakan untuk menyimpan gambar yang diambil dari kamera atau untuk menyimpan *file*. Modul ESP32-CAM ini dapat digunakan

secara luas di berbagai aplikasi IoT. ESP32-CAM yang digunakan dalam penelitian ini disajikan dalam **Gambar 2.4**.

Gambar 2.4 ESP32-Cam
(Sumber : hwlibre.com)



Gambar 2.5 Dataset ESP32-Cam
Sumber: randomnerdtutorials.com

ESP32-CAM merupakan salah satu mikrokontroler yang memiliki fasilitas tambahan berupa *bluetooth*, wifi, kamera, bahkan sampai ke slot mikroSD. ESP32-CAM ini biasanya digunakan untuk *project* IoT (*Internet of Things*) yang membutuhkan fitur kamera. Modul ESP32CAM memiliki lebih sedikit pin I/O dibandingkan modul ESP32 produk sebelumnya, yaitu ESP32 *Wroom*. Hal ini dikarenakan sudah banyak pin yang digunakan secara internal untuk fungsi kamera dan fungsi slot kartu microSD. Selain itu, modul

ESP32CAM juga tidak memiliki port USB khusus (mengirim program dari *port* USB komputer). Jadi untuk memprogram modul ini Anda harus menggunakan USB TTL atau kita dapat menambahkan modul tambahan berupa *downloader* khusus untuk ESP32-CAM.

Modul ESP32CAM memiliki 2 sisi dalam rangkaian modulnya. Di bagian atas terdapat modul kamera yang dapat dibongkar pasang dan ada microSD yang dapat diisi, serta *flash* sebagai lampu tambahan untuk kamera jika diperlukan. Di bagian belakang modul, terdapat antena internal, konektor untuk antena eksternal, pin male untuk I/O dan ESP32S sebagai otaknya. Lebih jelasnya, kita dapat melihat spesifikasinya sebagai berikut:

- 802.11b/g/n *Wi-Fi*
- *Bluetooth* 4.2 *with* BLE
- UART, SPI, I2C *and* PWM *interfaces*
- *Clock speed up to* 160 MHz
- *Computing power up to* 600 DMIPS
- 520 KB SRAM plus 4 MB PSRAM
- *Supports WiFi Image Upload*
- *Multiple Sleep modes*
- *Firmware Over the Air* (FOTA) *upgrades possible*
- 9 GPIO *ports*
- *Built-in Flash* LED
- Kamera

2.9 Servo Motor MG996R

Gambar 2.6 Servo Motor MG996R (Baihaqi, 2019)

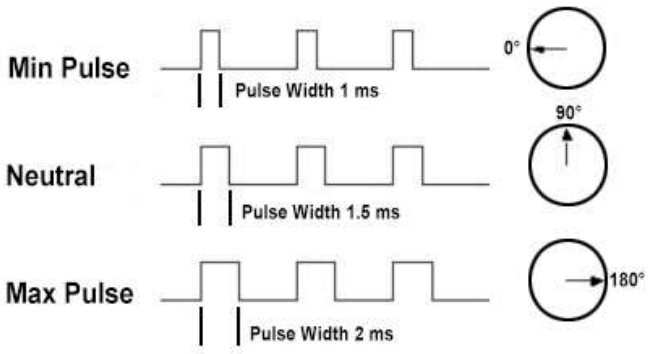
Motor Servo MG996R adalah sebuah perangkat atau aktuator putar (motor) yang dirancang dengan sistem kontrol umpan balik *loop* tertutup (servo), sehingga dapat di set-up atau di atur untuk menentukan dan memastikan posisi sudut dari poros *output* motor. motor servo merupakan perangkat yang terdiri dari motor DC.

Motor ini terdiri dari sebuah motor DC, rangkaian *gear*, *ponsiometer*, dan rangkaian kontrol. Potensiometer berfungsi sebagai penentu batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor servo. Ada beberapa keunggulan motor servo yang diantaranya yaitu:

1. Tenang saat beroperasi
2. Daya yang dihasilkan sebanding dengan ukuran motor
3. Arus listrik sebanding dengan penggunaan

Motor servo dikendalikan dengan sinyal PWM dari encoder/potentiometer. Lebar sinyal (pulsa) yang diberikan inilah yang akan menentukan posisi sudut putaran dari poros motor servo. Sebagai contoh, lebar sinyal dengan waktu 1,5 ms (mili second) akan memutar poros motor

servo ke posisi sudut 90º. Bila sinyal lebih pendek dari 1,5 ms maka akan berputar ke arah posisi 0º atau ke kiri (berlawanan dengan arah jarum jam), sedangkan bila sinyal yang diberikan lebih lama dari 1,5 ms maka poros motor servo akan berputar ke arah posisi 180º atau ke kanan (searah jarum jam) dapat dilihat pada **Gambar 2.7**. Ketika sinyal PWM telah diberikan, maka poros motor servo akan bergerak ke posisi yang telah ditargetkan dan berhenti pada posisi tersebut serta akan tetap bertahan pada posisi tersebut.



Gambar 2.7 Cara kerja servo
Sumber: https://www.jameco.com/

2.10 TCP/IP

TCP/IP merupakan Komponen penting di dalam jaringan. Beberapa bagian TCP/IP yang penting dan mencakup di seluruh jaringan yang digunakan sekarang ini adalah :

a. Protokol TCP/IP

Protokol (Harry:2013) merupakan sekumpulan aturan yang mengatur dua atau lebih mesin dalam suatu jaringan dalam melakukan interaksi pertukaran format data. Protokol memiliki suatu fungsi yang

spesifik satu sama lain pada sebuah hubungan telekomunikasi. TCP/IP merupakan sekumpulan protokol yang dikembangkan untuk mengizinkan komputer-komputer agar dapat saling membagi sumber daya yang dimiliki masing-masing melalui media jaringan. Protokol protokol TCP/IP dikembangkan sebagai bagian dari riset yang dikembangkan oleh *Defense Advanced Research Projects Agency* (DARPA). Pertama kalinya TCP/IP dikembangkan untuk komunikasi antar jaringan yang terdapat pada DARPA. Selanjutnya, TCP/IP dimasukkan pada distribusi *software* UNIX. Sekarang TCP/IP telah digunakan sebagai standar komunikasi *internetwork* dan telah menjadi protokol *transport* bagi internet, sehingga memungkinkan jutaan komputer berkomunikasi secara global.

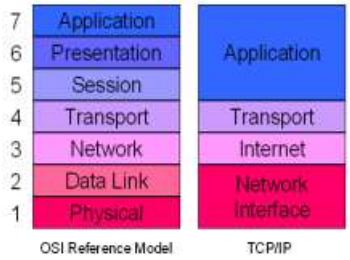
Protokol *Transmission Control Protocol*(TCP) adalah protokol yang berada pada model OSI dan menyediakan keandalan pengiriman paket secara *stream* dari layanan ke aplikasi dengan menerapkan beberapa mekanisme pengakuan(*acknowledgement*) dan re-transmisi paket pada kasus spesifik(Agus:2012).

Pengiriman aliran ini, kita dapat mengirim data melalui TCP/IP dengan format sendiri atau struktur data yang bebas. TCP/IP juga merupakan kelompok protokol berorientasi koneksi. Protokol TCP juga menyediakan operasi dupleks penuh di mana pengirim dan penerima dapat mengirim dan menerima data pada waktu yang bersamaan. Selain itu, TCP juga memunyai fitur *multiplexing* yang memungkinkan

beberapa komunikasi yang terjadi menjadi satu koneksi atau di-multipleks-kan.

b. Arsitektur TCP/IP

Seperti telah disebutkan sebelumnya, TCP/IP berisi kumpulan dari protokol-protokol yang melakukan fungsinya masing-masing secara spesifik. Protokol-protokol ini dikumpulkan berdasarkan fungsinya dalam lapisan-lapisan tertentu. TCP/IP memiliki 4 lapisan yang antara satu dengan lainnya memiliki protokol dengan fungsi yang saling melengkapi satu sama lain.



Gambar 2.8 Perbandingan OSI Layer dengan model TCP/IP (Sari, 2016)

c. Datagram IP

Sebuah datagram IP berisi *header* IP dan data, dan dikelilingi oleh *Media Access Control* (MAC) *header* dan MAC trailer. Satu pesan dapat dikirim dalam urutan datagram-datagram yang disusun kembali menjadi pesan asli pada sisi penerima.

d. TCP dan UDP

TCP dan UDP menggunakan nomor *port* (atau soket) untuk melewatkan informasi ke lapis yang lebih atas. Nomor *port* digunakan untuk membedakan aplikasi yang berbeda yang melewati jaringan pada saat yang bersamaan. Pengembang *software* aplikasi telah sepakat

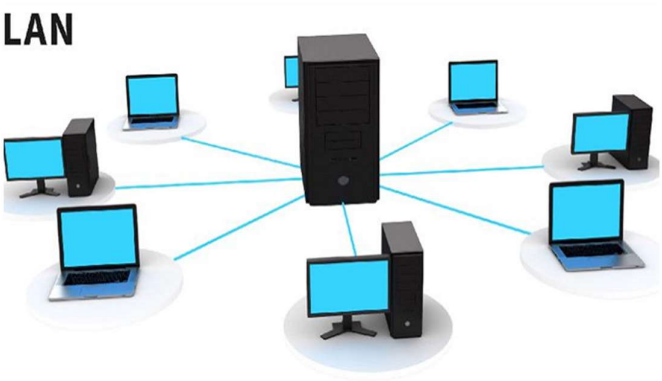
untuk menggunakan nomor nomor *port* yang didefinisikan dalam RFC 1700. Sebagai contoh, semua percakapan yang digunakan untuk aplikasi FTP menggunakan nomor *port* standar 21.

2.11 *Local Area Network* (LAN)

Local Area Network adalah sebuah sambungan komputer yang memiliki sambungan antara komputer satu dengan yang lainnya yang sambungan tidak terlalu jauh seperti dalam sebuah gedung yang sama. Dan dengan sambungan jaringan LAN itu akan menjadikan satu komputer bisa berhubungan dengan komputer lainnya. Beberapa keuntungan menggunakan jaringan LAN adalah:

1. Pertukaran *file* dapat dilakukan dengan mudah (*File Sharing*).
2. Pemakaian printer dapat di lakukan oleh semua *client* (*Printer Sharing*).
3. *File-file* dapat disimpan pada server ,sehingga data dapat diakses dari semua *client* menurut otorisasi sekuritas dari semua karyawan ,yang dapat di buat berdasarkan struktur organisasi perusahaan sehingga keamanan data terjamin.
4. *File* data yang keluar/masuk dari atau ke server dapat di kendalikan.
5. Proses *backup* data lebih mudah dan cepat.
6. Risiko kehilangan data oleh virus sangat kecil sekali.

7. Komunikasi antar karyawan dapat di lakukan dengan menggunakan E-mail dan Chat.

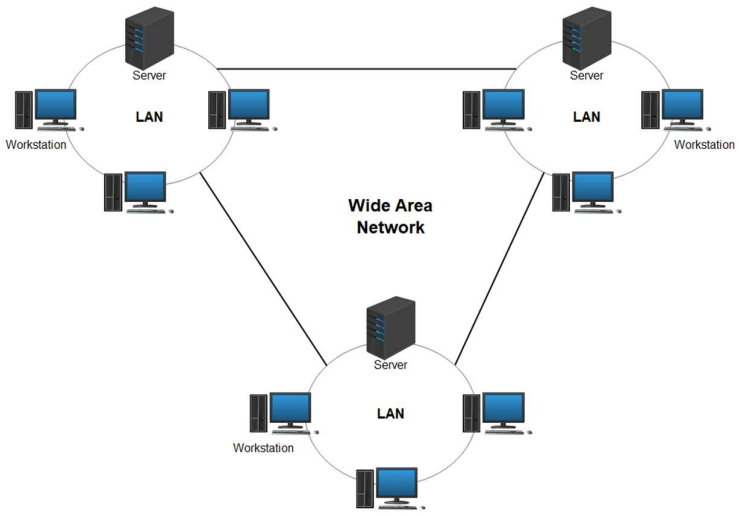


Gambar 2.9 *Local Area Network* (LAN)
Sumber : diengcyber.com

2.12 *Wide Area Network* (WAN)

Wide Area Network (WAN) adalah sebuah jaringan yang jangkauannya mencakup daerah geografis yang lebih luas, sering kali mencakup sebuah negara bahkan benua. WAN terdiri dari kumpulan LAN, MAN, dan mesin-mesin yang bertujuan untuk menjalankan program aplikasi pemakai. Contoh penggunaan WAN adalah hubungan antara kantor pusat dengan kantor cabang yang ada didaerah-daerah. Jaringan WAN memiliki beberapa kelebihan ,yaitu :

1. Apabila terhubung dengan jaringan internet maka transfer *file* pada tempat yang saling berjauhan dapat dilakukan dengan cepat menggunakan e-mail dan FTP (*File TransferProtocol*).
2. Memiliki sistem jaringan yang luas sehingga dapat mencapai Negara , benua ,bahkan seluruh dunia.



Gambar 2.10 *Wide Area Network* (WAN)

Sumber: edrawsoft.com

2.13 REST API

Salah satu metode yang dapat digunakan untuk pengembangan teknologi *web service* adalah REST. REST merupakan singkatan dari *Representational State Transfer*. Metode REST *web service* menerapkan konsep perpindahan antar *state*. *State* yang dimaksud di sini dapat digambarkan apabila *browser* melakukan permintaan suatu *web*, maka *server* akan melakukan pengiriman *state* halaman *web* yang sekarang ke *browser*. Ide dasar dari metode REST adalah bagaimana menghubungkan jalur komunikasi antar mesin atau aplikasi melalui HTTP sederhana. Istilah REST atau RESTful pertama kali diperkenalkan oleh Roy Fielding pada disertasinya di tahun 2000. REST merupakan interaksi antara *client* dan server difasilitasi oleh sejumlah tipe operasional yang unik bagi setiap sumber daya. Tipe operasional tersebut dapat berupa POST, GET, PUT dan DELETE dll. Istilah yang biasa digunakan untuk menyatakan prinsip *uniform interface* pada

REST adalah CRUD (*Create, Read, Update, Delete*). Berikut ini ulasan detail mengenai method-method tersebut :

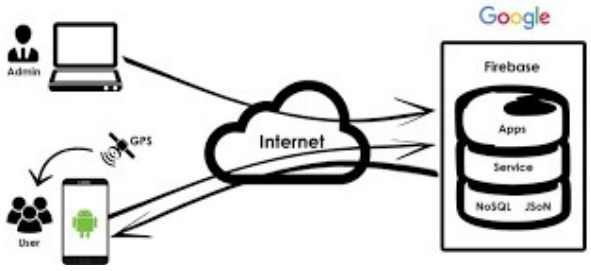
a. GET, operasi *read-only* yang digunakan untuk meminta informasi spesifik pada server dalam bentuk *query*. Karakteristik dari operasi GET adalah *idempotent* dan *safe*.
b. PUT, operasi untuk meminta kepada *server* agar membuat sebuah *resource* baru.
c. DELETE, operasi untuk menghapus *resource* tertentu.
d. POST, operasi untuk membuat *resource* baru atau memodifikasi *resource* yang telah ada.
e. HEAD, operasi yang mirip dengan GET, namun *response message* yang dikembalikan hanyalah berupa *response code* dan *message header*.
f. OPTIONS, operasi yang mengembalikan informasi mengenai berbagai HTTP *methods* yang didukung oleh suatu *server*, berguna untuk mengecek fungsionalitas sebuah *server* sebelum melakukan operasi sesungguhnya.

Application Programming Interface atau biasa disingkat API adalah suatu penghubung yang memungkinkan suatu aplikasi untuk berinteraksi dengan aplikasi lainnya dan berbagi data (Ahmadi, 2017). API dirancang dengan konsep *Representational State Transfer* (REST). REST memungkinkan *client* dapat melakukan *request* melalui protokol HTTP

dengan mudah menggunakan *Uniform Resource Identifier* (URI) (Kurniawan dkk, 2013).

2.14 *Firebase*

Firebase yakni model layanan yang bekerja di belakang layar dan menghubungkan aplikasi *mobile* ke *cloud storage*. *Firebase* Realtime *Database* adalah *database* yang di-*host* di *cloud*. Data disimpan sebagai JSON dan disinkronkan secara *realtime* ke setiap klien yang terhubung. Ketika anda membuat aplikasi lintas-platform dengan SDK Android, iOS, dan JavaScript, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis. (*Firebase*, 2018).



Gambar 2.11 Arsitektur sistem Firebase

Sumber: eprints.utdi.ac.id

Semua data *Firebase* Realtime Database disimpan sebagai objek JSON. Bisa dianggap basis data sebagai JSON *tree* yang di-*host* di awan. Tidak seperti basis data SQL, tidak ada tabel atau rekaman. Ketika ditambahkan ke JSON *tree*, data akan menjadi simpul dalam struktur JSON yang ada. Meskipun basis data menggunakan JSON tree, data yang tersimpan dalam basis data bisa diwakili sebagai tipe bawaan tertentu yang sesuai dengan tipe

JSON yang tersedia untuk membantu untuk menulis lebih banyak kode yang bisa dipertahankan.

2.15 Web server

Web server adalah *software* yang memberikan layanan data yang mempunyai fungsi untuk menerima permintaan HTTP (*HyperText Transfer Protocol*) atau HTTPS yang dikirim oleh klien melalui web browser dan mengirimkan kembali hasilnya dalam bentuk halaman web yang umumnya berbentuk dokumen HTML (*HyperText Markup Language*). *Web server* berguna sebagai tempat aplikasi *web* dan sebagai penerima *request* dari *client* (Indra Warman & Zahni, 2013).

Saat mengambil halaman *website*, browser mengirimkan permintaan ke server yang kemudian diproses oleh *web server*. HTTP *request* dikirimkan ke *web server*. Sebelum memproses HTTP *request*, *web server* juga melakukan pengecekan terhadap keamanan. Pada *web server*, HTTP *request* diproses dengan bantuan HTTP *server*. HTTP server merupakan perangkat lunak yang bertugas menerjemahkan URL (alamat situs *website*) serta HTTP (protokol yang digunakan browser untuk menampilkan halaman *website*). Kemudian *web server* mengirimkan HTTP *response* ke browser dan memprosesnya menjadi halaman situs *website*.

2.16 Flask

Dalam sejarah perkembangannya, *flask* pertama kali dirilis pada April 2010 dibuat oleh Armin Ronacher. *Flask* adalah sebuah *web framework* yang ditulis dengan bahasa *python* yang menyediakan *library* yang dapat digunakan untuk membangun sebuah *website*. *Flask* tersedia di *repository* publik *python* yang semua orang dapat mengunduh dan menggunakannya. *Flask* juga memiliki fungsi yang bisa dikembangkan sesuai kebutuhan (Samudera dkk, 2015).

Flask berfungsi sebagai kerangka kerja aplikasi dan tampilan dari suatu web. Dengan menggunakan *Flask* dan bahasa *Python*, pengembang dapat membuat sebuah web yang terstruktur dan dapat mengatur *behaviour* suatu web dengan lebih mudah. *Flask* termasuk pada jenis *microframework* karena tidak memerlukan suatu alat atau pustaka tertentu dalam penggunaannya. Sebagian besar fungsi dan komponen umum seperti validasi *form*, *database*, dan sebagainya tidak terpasang secara *default* di *Flask*. Hal ini dikarenakan fungsi dan komponen-komponen tersebut sudah disediakan oleh pihak ketiga dan *Flask* dapat menggunakan ekstensi yang membuat fitur dan komponen-komponen tersebut seakan diimplementasikan oleh *Flask* sendiri.

Selain itu, meskipun *Flask* disebut sebagai *microframework*, bukan berarti *Flask* mempunyai kekurangan dalam hal fungsionalitas. *Microframework* di sini berarti bahwa *Flask* bermaksud untuk membuat *core* dari aplikasi ini sesederhana mungkin tapi tetap dapat dengan mudah

ditambahkan. Dengan begitu, fleksibilitas serta skalabilitas dari *Flask* dapat dikatakan cukup tinggi dibandingkan dengan *framework* lainnya.

Web framework Flask ditulis menggunakan bahasa *Python*, sehingga sebelum *Flask* dapat digunakan, maka developer harus menginstall *Python* pada perangkat yang akan digunakan. Oleh sebab itu, *web developer* yang akan menggunakan *Flask* sebagai *web framework* untuk *web development* harus setidaknya mempelajari bahasa pemrograman *Python* terlebih dahulu, sebelum dapat menggunakan *Flask* seutuhnya.

Dalam melakukan instalasi *Flask* pada sebuah perangkat, dibutuhkan PIP yang biasanya sudah terinstal pada *Python* versi 3.4 ke atas. PIP adalah sebuah *package management system* yang biasa digunakan untuk mengatur dan mengunduh *package* yang berisi modul-modul *Python*. PIP digunakan untuk mengunduh *Flask* karena *Flask* ditulis dan dikembangkan dengan bahasa dan modul-modul pemrograman Python. Dengan menggunakan PIP, semua hal yang dibutuhkan untuk instalasi Flask akan diunduh dan dipasang dalam satu perintah.