BAB II
LANDASAN TEORI

2.1 Pertanian Presisi

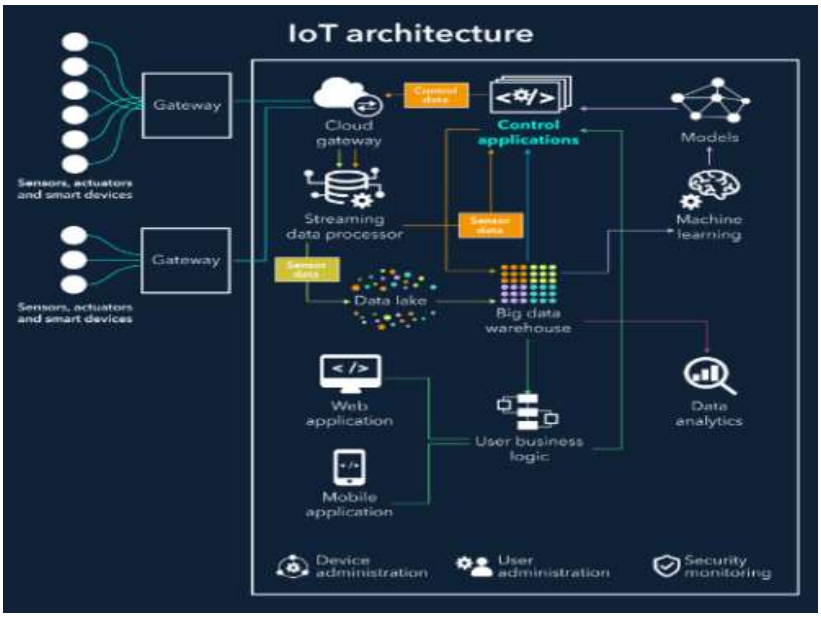
Pertanian presisi adalah suatu konsep manajemen pertanian berdasarkan pengamatan, pengukuran, dan response terhadap variabilitas dalam dan antar bidang pada tanaman. Pertanian presisi merupakan integrasi teknologi sistem informasi dan sistem pertanian dimaksudkan guna mendukung efisiensi, produktifitas dan profitabilitas pertanian. Terdapat tiga aspek produksi yang diterapkan pada teknologi informasi dalam pertanian presisi yaitu koleksi input data, analisis atau pengolahan proses untuk menghasilkan informasi, dan aplikasi untuk menampilkan informasi kepada pengguna [5]. Berikut dibawah ini gambaran dari teknologi IoT yang dipakai dalam *precision agriculture*.

Gambar II-1 Penerapan IoT pada pertanian presisi

2.2 Internet of Thing

Internet Of Thing atau dikenal sebagai IoT ini merupakan sebuah konsep yang digunakan perangkat untuk berkomunikasi dengan internet, baik untuk mengirim data atau menerima data. Banyak daerah-daerah yang telah didukung oleh teknologi ini, misalnya untuk keperluan dibidang transportasi, pertanian, kesehatan dan kota pintar [7]. Tujuan utama IoT adalah memungkinkan segala

sesuatu dapat diakses kapan saja dan dimana saja dengan menggunakan layanan jaringan apapun. Dibawah ini merupakan gambaran dari arsitektur IoT [8].



Gambar II-2 IoT Architecture

Cara kerja IoT yaitu setiap benda harus memiliki sebuah alamat *Internet Protocol* (IP). Alamat IP adalah sebuah identitas dalam jaringan yang membuat benda tersebut bisa diperintahkan dengan benda lainnya didalam sebuah jaringan yang sama. Setelah itu, alamat IP dalam benda tersebut akan disambungkan ke jaringan internet.

2.3 Kecerdasan Buatan

Kecerdasan Buatan adalah salah satu cabang ilmu pengetahuan yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal ini biasanya dilakukan dengan mengikuti/mencontoh karakteristik dan analogi berpikir dari kecerdasan/intelegensia manusia, dan merapkannya sebagai algoritma yang dikenal oleh komputer. Dengan suatu pendekatan yang kurang lebih fleksibel dan efisien dapat diambil tergantung dari keperluan, yang mempengaruhi bagaimana wujud dari perilaku kecerdasan buatan. AI (*Artificial Intelegence*) biasanya dihubungkan dengan Ilmu Komputer, akan tetapi juga terkait erat dengan bidang

Dengan bidang lain seperti Matematika, Psikologi, Biologi, Filosofi, dan masih banyak lagi. Kemampuan untuk mengkombinasikan pengetahuan dari semua bidang ini pada akhirnya akan bermanfaat bagi kemajuan dalam upaya menciptakan suatu kecerdasan buatan [9].

2.4 UML

Unified Modeling Language (UML) adalah bahasa pemodelan untuk system dan perangkat lunak yang berparadigma berorientasi objek. Abstraksi konsep dasar UML terdiri dari *structural classification*, *dynamic behavior*, dan model *management* dapat kita pahami main *concepts* sebagai *term* yang akan muncul pada saat membuat diagram dan *view* adalah kategori dari diagram tersebut. UML mendefinisikan diagram-diagram sebagai *Use case diagram*, *Class diagram*, *Statechart diagram*, *Activity diagram*, *Sequence diagram*, *Collaboration diagram*, *Component diagram*, dan *Deployement diagram*. Dan berikut beberapa jenis diagram UML, yaitu:

1. Diagram Kelas : diagram kelas ini bersifat statis. Pada diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram jenis ini umumnya bias kita jumpai pada pemodelan system berorientasi objek. Meskipun sifatnya itu statis tapi diagram jenis ini mampu memuat kelas-kelas aktif.
2. Diagram paket : bersifat statis diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram Usecase : diagram ini sifatnya sama seperti diagram kelas. Pada diagram ini memperlihatkan himpunan usecase dan actor-aktor suatu jenis khusus dari suatu kelas. Diagram ini sangat penting untuk mengorganisasi dan memodelkan perilaku suatu system yang dibutuhkan serta diharapkan pengguna.
4. Diagram *Sequence*. Merupakan diagram yang menggambarkan kolaborasi dinamis antar sejumlah object. Fungsinya untuk menunjukkan rangkaian pesan yang akan dikirim antara object juga interaksi antara object.

5. Diagram *Activity*. Diagram jenis ini lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram aktivitas ini menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi [10].

2.5 OOP

Menurut Priyanto Hidayatullah *Object Oriented Programming* atau biasa dikenal dengan nama OOP adalah sebuah paradigma atau teknik pemrogramman yang berorientasikan pada objek. Semua data dan fungsi dalam OOP dibagi kedalam kelas-kelas atau objek-objek [11]. Sedangkan menurut Hartanto Kurniawan, OOP merupakan paradigma pemrogramman yang berorientasi pada objek. Semua data dan fungsi pada paradigma ini dibungkus dalam *class-class* atau *object-obejct* [12]. Berdasarkan definisi OOP menurut Priyanto Hidayatullah dan Dharma, Akhmad dapat ditarik kesimpulan bahwa OOP merupakan suatu konsep pemrogramman yang mengacu pada object dimana semua data atau fungsi dibungkus dalam objek tersebut. Untuk lebih memahami konsep OOP berikut ilustrasi dari OOP:

Misalnya saja disebuah ruangan kelas perkuliahan terdapat meja, kursi, papan tulis, penggaris dan spidol. Dalam hal ini ruang kelas menjadi kelas sedangkan meja, kursi, papan tulis, penggaris dan spidol adalah sebuah objek. Akan tetapi ruang kelas pun bisa saja menjadi sebuah objek jika pada contoh ini kita mendeklarasikan suatu gedung, dimana gedung tersebut memiliki banyak kelas [11].

2.6 PHP

Menurut Priyanto Hidayatullah PHP *Hypertext Preprocessor* atau yang lebih dikenal dengan PHP adalah suatu bahasa *scripting* khususnya digunakan untuk *web development*. Karena sifatnya yang *server side scripting*, dimana untuk bisa menjalankan PHP dibutuhkan nya *web server* [11].

Dan menurut Andre Pratama PHP adalah bahasa pemrogramman web yang digunakan untuk men-generate atau menghasilkan kode HTML [13].

Berdasarkan dari definisi PHP menurut Priyanto Hidayatullah dapat diambil kesimpulan bahwa PHP adalah sebuah bahasa pemrogramman yang berjalan disisi server dan berfungsi untuk kegiatan manipulasi data.

Kelebihan dari PHP berbasis *server side scripting* ini adalah dapat melakukan tugas-tugas yang dilakukan dengan mekanisme CGI seperti mengambil dan mengumpulkan data dari database, meng-*generate* halaman dinamis, atau bahkan menerima dan mengirim *cookie*. Dan yang menjadi keutamaan PHP adalah bisa digunakan diberbagai *operating system*, diantaranya Linux, Unix, Windows, Mac OsX, dan operating system lainnya [11].

Selain dari kelebihan PHP diatas, PHP juga sudah mendukung *framework*. *Framework* bisa diartikan sekumpulan perintah/program dasar dimana perintah dasar tersebut dapat digunakan lagi untuk menyelesaikan masalah yang lebih kompleks sehingga dapat digunakan untuk membantu untuk membuat aplikasi baru atau aplikasi yang sudah kompleks tanpa harus membuat ulang program dari awal (BUKU Priyanto Hidayatullah). Berikut beberapa *framework* yang popular digunakan oleh pengembang *web apps* :

a. Laravel
b. Codeigniter
c. Symfony
d. Yii
e. CakePHP

2.7 Javascript

Javascript adalah sebuah bahasa pemrogramman yang termasuk kedalam varian *script language*, *scripting* memerlukan program *interpreter* agar bisa dimengerti oleh computer. Secara historisnya penciptaan javascript digunakan agar sebuah halaman *web* bisa menjadi lebih hidup. Javascript adalah suatu bahasa yang paling popular didunia pemrogramman. Bahasa javascript adalah bahasa yang *versatile* karena bisa digunakan untuk membuat berbagai macam aplikasi [14].

Sama seperti PHP, javascript juga telah didukung oleh beberapa *framework*, berikut ini beberapa contoh *framework* javascript :

a. Node.Js
b. React.Js
c. Angular.Js
d. Vue.Js
e. Ember.Js

2.8 MySQL

Menurut Priyanto Hidayatullah MySQL adalah sebuah aplikasi DBMS (*Database Management System*) yang dipakai untuk mengelola basis data. DBMS sendiri biasanya menawarkan beberapa kemampuan yang terintegrasi seperti:

1. Membuat, menghapus, menambah, dan memodifikasi basis data.
2. Pada beberapa DBMS pengelolaannya berbasis *windows* (berbentuk jendela-jendela) sehingga lebih mudah digunakan.
3. Tidak semua orang bisa mengakses basis data yang telah kita buat, sehingga memberikan keamanan bagi data.
4. Kemampuan berkomunikasi dengan program aplikasi lain. Misalnya dimungkinkan untuk mengakses basis data MySQL menggunakan aplikasi yang dibuat menggunakan PHP.
5. Kemampuan pengaksesan melalui komunikasi antarkomputer (*client server*).

Kelebihan dari MySQL sendiri adalah gratis, handal, selalu *update* dan banyak forum yang memfasilitasi pengguna jika memiliki kendala. MySQL juga menjadi DBMS yang sering dibundling dengan *web server* sehingga proses instalasinya jadi lebih mudah [11].

2.9 Protocol MQTT

Protocol MQTT (*Message Queuing Telemetry Transport*) merupakan sebuah protocol komunikasi data machine to machine (M2M) yang berada pada layer aplikasi. Protocol ini bersifat lightweight message artinya MQTT ini

berkomunikasi dengan mengirimkan data pesan yang memiliki header berukuran kecil yaitu sekitar 2bytes untuk setiap jenis data, sehingga dapat bekerja didalam lingkungan yang terbatas sumber dayanya seperti kecilnya *bandwidth* data terbatasnya sumber listrik, selain itu protocol ini juga menjamin terkirimnya semua pesan walaupun koneksi terputus sementara, protocol MQTT ini menggunakan suatu metode *publish/subscribe* untuk metode komunikasinya [15].

2.10 Mosquitto Broker

Mosquitto broker adalah salah satu *opensource* broker pesan yang mengimplementasikan protocol MQTT versi 3.1 dan 3.1.1. broker mosquito juga mendukung pengimplementasian server lightweight dari MQTT maupun MQTT-SN. Mosquito broker ini ditulis dalam bahasa pemrogram C dengan alasan agar dapat tetap bekerja pada mesin yang tidak mendukung JVM (Java Virtual *Machine*) [15].

2.11 Paho MQTT

Paho MQTT adalah sebuah library untuk MQTT *client* yang telah dikembangkan oleh eclipse. Paho library ini tersedia dalam berbagai bahasa pemrogramman seperti Lua, Python, C++ dan Javascript. Paho MQTT menggunakan 3 level Qos MQTT untuk pengiriman/*publish* pesan dan juga untuk penerimaan/*subscribe* pesan [15].

2.12 Python

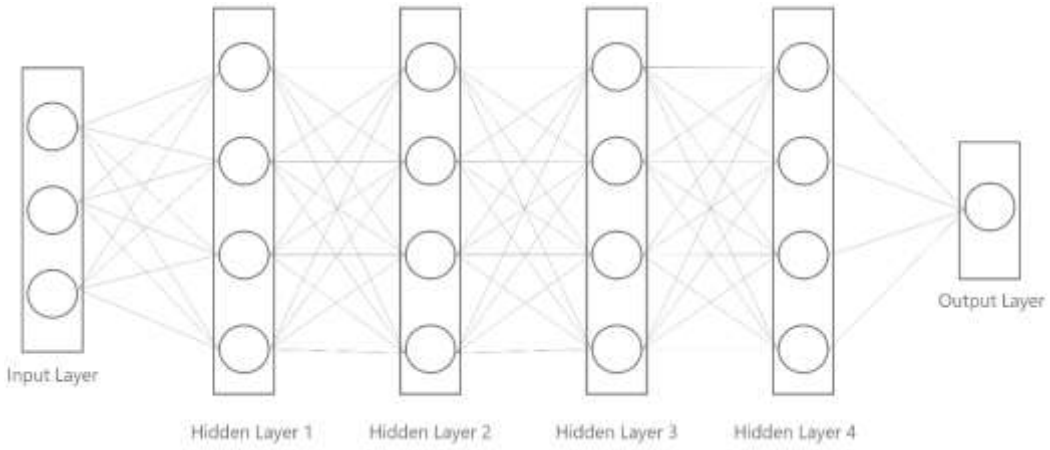
Menurut Muhammad Fuad Muttaqin dalam e-Proceeding nya Python merupakan bahasa pemrogramman *interpretative* multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas Pustaka standar yang besar serta komprehensif. Python merupakan bahasa pemrogramman yang bersifat *open source* dan *multiplatform*, ada beberapa *feature* yang dimiliki Python[16].

2.13 *Scikit-learn*

Scikit-learn adalah salah satu modul Python yang menyediakan berbagai macam algoritma *machine learning* yaitu *scikit-learn* yang tersedia dalam bentuk *library* python. *Scikit-learn* menggunakan *task-oriented interface* yang konsisten sehingga mudah dalam melakukan perbandingan antar metode [17].

2.14 *Recurrent Neural Network*

RNN merupakan bagian dari *machine learning* yang berbasis ekstraksi fitur dari sebuah data secara lebih detail. RNN merupakan pengembangan dari Jaringan Syaraf Tiruan (JST) dan arsitekturnya hampir sama dengan Multi Layer Perception (MLP) cara kerja *Deep Learning* meniru cara kerja dari otak manusia sehingga dapat mengirimkan informasi dari sebuah *neuron* menuju *neuron* lain [18]. *Recurrent Neural Network* memiliki variasi jenisnya seperti *Long Short Term Memory* (LSTM). LSTM ini digunakan untuk mengatasi masalah yang ada dalam RNN yaitu ingatan memori jangka pendek. Berikut gambaran arsitektur RNN dapat dilihat pada Gambar II.3 dibawah ini.

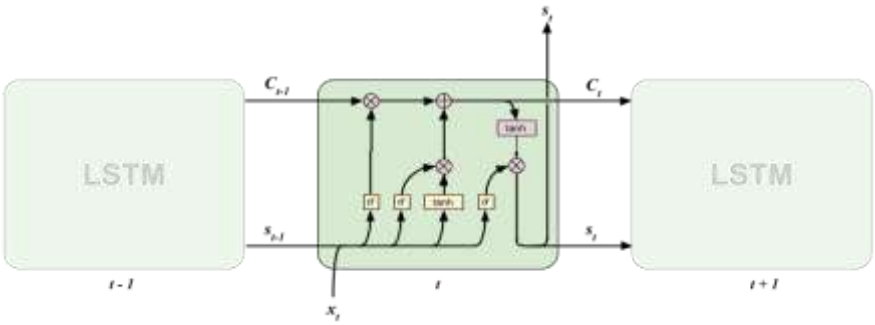


Gambar II-3 Arsitektur Recurrent Neural Network dengan 4 hidden layer

Long Short Term Memory (LSTM) jaringan yang diciptakan dengan tujuan mengatasi masalah *hidden layer*. Kunci utama dalam desain LSTM adalah menggabungkan control *non-linier,* data *dependent* ke dalam sel RNN, yang dapat dilatih untuk memastikan bahwa gradien fungsi tujuan dengan memperhatikan

sinyal (kuantitas berbanding lurus dengan pembaruan parameter yang dihitung selama pelatihan oleh *Gradient Descent*) tidak menghilang.

LSTM digunakan untuk mengatasi masalah *vanishing gradient* atau situasi dimana nilai *gradient* bernilai 0 atau mendekati 0 dengan mekanisme *gate*. LSTM merupakan cara lain untuk menghitung *hidden state*. Memori dalam LSTM disebut dengan *cells* yang mengambil *input* dari *state* sebelumnya (h_{t-1}) dan *input* saat ini (x_t). Kumpulan *cells* tersebut memutuskan apa yang akan disimpan dalam memori dan kapan yang akan dihapus dari memori. LSTM menggabungkan *state* sebelumnya, memori saat ini, dan *input*. LSTM sangat begitu efisien untuk merekam *long-term dependencies*. Berikut gambaran dari Arsitektur *Long Short Term Memory*.



Gambar II-4 Arsitektur Long Short Term Memory

Dilihat dari Gambar *II-4*, LSTM mempunyai 3 *gate*, yaitu *gate* yang pertama adalah *forget gate* untuk menentukan informasi yang akan dihilangkan dari *cell* menggunakan *sigmoid layer* seperti terlihat pada persamaan 10 dengan fungsi aktivasi yang digunakan adalah Relu. *Gate* yang kedua adalah *input gate* yang mana dari *sigmoid layer* yang akan diperbaharui dan *tanh layer* akan dibuat sebagai sebuah vector dari nilai yang diperbaharui. *Gate* yang ketiga adalah *output gate* untuk memaparkan isi sel memori pada *output* LSTM.

LSTM diperlukan sebagai kotak hitam yang diberikan *current input* dan *hidden state* sebelumnya untuk menghitung *hidden state* selanjutnya. Memori yang digunakan dalam LSTM disebut dengan *cells* yang mengambil *input* dari *state* sebelumnya (h_{t-1}) dan input saat ini (x_t) [19].