BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Kualitas Air

Kualitas air merupakan faktor dalam kehidupan sehari-hari, produktivitas dan kelangsungan hidup sangat ditentukan oleh parameter fisik kualitas air (Bataragoa et al. 2023) beberapa parameter fisik kualitas air antara lain suhu, pH, dan jumlah zat padat terlarut (TDS). Semakin tinggi taraf kehidupan seseorang semakin meningkat pula kebutuhan manusia akan air. Jumlah penduduk dunia setiap hari bertambah, sehingga mengakibatkan jumlah kebutuhan air.

2. pH (*Potential Of Hydrogen*)

Asam adalah zat yang di dalan air melepaskan ion H+ atau bisa dikatakan ion H+ adalah pembawa sifat asam. Basa adalah senyawa yang bisa melepaskan ion hidroksida di dalam air (OH). Jadi, ion OH adalah pembawa sifat basa menurut Arrhenius. Singkatan kata pH adalah "pondus Hydrogenium". pH berarti berat hydrogen. Derajat atau tingkatan keasaman larutan bergantung pada konsentrasi ion H+ dalam larutan. Semakin besar ion H+ , maka semakin asam larutan tersebut. Nilai konsentrasi ion H+ biasanya sangat kecil (Nurhidayat 2021:44).

3. Hardness

Didefinisikan sebagai kandungan terukur dari *logam kation divalen*, kalsium terlarut (Ca++) dan *magnesium* (Mg++) adalah dua *kation divalen* yang ditemukan pada tingkat yang cukup tinggi di sebagian besar perairan, pada

kandungan air yang alami, kalsium dan magnesium terikat pada *bikarbonat*, *sulfate* atau *klorida* (Jasman et al. 2022:394).

4. *Solids Total Dissolved solid* (TDS)

TDS (*total dissolve solid*) merupakan ukuran zat terlarut (baik itu zat organic maupun anorganik, misalnya: garam, dll.) yang terdapat pada sebuah larutan. *TDS* meter menggambarkan jumlah zat terlarut dalam part per *million* (ppm) atau sama dengan milligram per liter (mg/L). Umumnya berdasarkan definisi diatas seharusnya zat yang terlarut dalam air (larutan) harus dapat melewati saringan yang berdiameter 2 mikrometer (2×10-6 meter). Aplikasi yang umum digunakan adalah untuk mengukur kualitas cairan biasanya untuk pengairan, pemeliharaan aquarium, kolam renang, proses kimia, pembuatan air mineral, dll.. Setidaknya, kita dapat mengetahui air minum mana yang baik dikonsumsi tubuh, ataupun air murni untuk keperluan kimia kosmetika, (misalnya obat-obatan, pembuatan makanan, dll.). Sampai saat ini ada dua metoda yang dapat digunakan untuk mengukur kualitas suatu larutan (Kusniawati and Budiman 2021:11).

5. Chloramines

Kloramin dibuat dari sekelompok bahan kimia yang mengandung amonia dan *klorin*, *kloramin* yang paling umum digunakan dalam pengolahan air kota adalah *monokloramin*, *monokloramin* ditambahkan ke air dalam jumlah yang telah diukur, memastikan bahwa *mikroorganisme* telah dibunuh, tetapi air masih aman untuk diminum (Jasman et al. 2022:394).

6. ***Sulfate***

Sulfat adalah kombinasi belerang dan oksigen dan merupakan bagian dari mineral alami di beberapa bagian tanah dan batuan yang mengandung air tanah, mineral larut dari waktu ke waktu dan dilepaskan ke dalam air tanah (Jasman et al. 2022:395).

7. ***Organic Carbon***

Total Organic Carbon adalah jumlah karbon organik yang ada dalam batuan utama yang dinyatakan sebagai persen berat, *TOC* adalah mewakili untuk jumlah total bahan organik yang ada dalam sedimen dan digunakan sebagai indikator kekayaan sumber sehubungan dengan seberapa banyak *hidrokarbon* yang dapat dihasilkan oleh sedimen (Jasman et al. 2022:395).

8. ***Conductivity***

Ukuran kemampuan air untuk mengalirkan arus listrik terutama dipengaruhi dalamnya air oleh adanya padatan terlarut anorganik, seperti *klorida, nitrat, sulfat, natrium, magnesium*, dan sebagainya, hal ini membantu mengetahui kualitas air berdasarkan garam terlarut dan juga membantu untuk menentukan jumlah reaksi kimia atau teknik perawatan yang diperlukan untuk memurnikan air (Jasman et al. 2022:395).

9. ***Trihalomethanes***

THMs adalah cairan yang mudah menguap pada suhu kamar dan berbagai efek racun telah dihubungkan dengan paparan jangka pendek dan jangka panjang dari hewan percobaan pada dosis tinggi, *THM* yang diberikan oleh saluran cairan

pada jagung menyebabkan keracunan yang lebih signifikan daripada dosis setara yang diberikan dalam emulsi cair (Jasman et al. 2022:395).

10. Turbidity

Turbidity atau kekeruhan mengacu pada sifat hamburan cahaya pada sampel, kekeruhan dapat digambarkan sebagai “kabur” atau “putih”, dan disebabkan oleh partikel halus yang menghamburkan cahaya pada kurang lebih 90 derajat ke arah dari cahaya memasuki sampel. Kekeruhan tidak sama dengan warna, atau warna dengan kekeruhan (Jasman et al. 2022:395).

11. Potability

Merupakan kelas dari data ini di mana kelas 0 adalah kelas yang di mana air tersebut tidak dapat diminum. Sedangkan kelas 1 merupakan kelas air yang dapat diminum (Jasman et al. 2022:395)**.**

12. Machine Learning

Machine learning adalah bidang ilmu yang mempelajari bagaimana sistem komputer dapat menyelesaikan tugas tertentu tanpa instruksi eksplisit dengan menggunakan algoritma dan metode statistik, ini adalah bagian dari kecerdasan buatan dan melibatkan pembentukan model matematika dari data pelatihan untuk membuat prediksi atau keputusan tanpa instruksi khusus (Arbain et al., 2022:185). Sedangkan menurut R.H. Zer et al., (2022 :151) mendefinisikan *machine learning* sebagai disiplin ilmu dalam kecerdasan buatan yang menggunakan bahasa pemrograman untuk membuat komputer berperilaku cerdas seperti manusia. Proses ini membutuhkan data yang dianalisis pada kumpulan data besar (*Big Data*) untuk menemukan pola tertentu, *Machine learning* adalah cabang dari

kecerdasan buatan dan ilmu komputer yang berfokus pada penerapan data dan algoritma untuk mengikuti cara belajar manusia dan meningkatkan kinerjanya secara bertahap. Berdasarkan beberapa definisi tersebut, dapat disimpulkan bahwa machine learning adalah cabang ilmu komputer dan kecerdasan buatan yang berfokus pada penerapan algoritma dan data untuk mengajar komputer belajar dari data, mengekstrak pola, dan membuat keputusan tanpa instruksi eksplisit. *Machine learning* mencakup metode pembelajaran *supervised, unsupervised, semi-supervised, reinforcement learning, dan deep learning.*

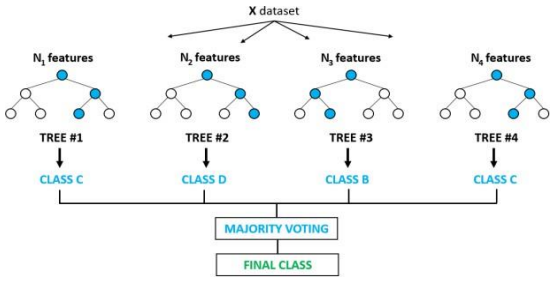
13. Random forest

Random forest adalah model pembelajaran mesin yang menggunakan pembelajaran terawasi dan terdiri dari banyak pohon keputusan, pohon keputusan bertindak sebagai klasifikasi dasar dalam hutan acak dan bekerja bersama sebagai sebuah kesatuan, inilah sebabnya mengapa *random forest* kadang-kadang disebut sebagai *Classifier Ensembles* (Dutta et al., 2020:536) sedangkan berdasarkan pendapat Mekha & Teeyasuksaet (2021:166) *random forest* adalah sebuah metode pembelajaran mesin yang menggabungkan banyak pohon keputusan untuk klasifikasi dan metode lainnya. Dalam algoritma *random forest*, banyak pohon keputusan digunakan untuk melatih dan memberikan hasil prediksi klasifikasi (regresi) dari setiap pohon. Algoritma *random forest* berfungsi sebagai sebuah prediktor yang terdiri dari properti-properti yang didasarkan pada regresi acak dari pohon-pohon tersebut.

Random forest adalah metode klasifikasi (*supervised leraning*) yang melakukan pengelompokan pada variabel dependennya. Metode ini menggabungkan teknik klasifikasi pohon dan *bootstrap aggregating (bagging).*

Random forest dikembangkan oleh Leo Breiman. Model *Random forest* terdiri dari beberapa model klasifikasi pohon yang dibuat dengan mengambil sampel ulang data. Setiap pohon akan memberikan hasil klasifikasi, yang kemudian akan dipilih oleh *Random forest*, hasil ini akan dipilih berdasarkan agregasi dari hasil pohon yang terbentuk.

Ilustrasi proses dalam algoritma *Random Forest* dapat dilihat pada Gambar 2.1 dibawah ini:



Gambar 2.1 Ilustrasi *Random Forest*

Sumber : (Mekha & Teeyasuksaet 2021:166)

Berdasarkan gambar diatas, tahapan persiapan dan estimasi menggunakan *random forest* sebagai berikut:

a. Langkah awal adalah melakukan *bootstrap* hingga mencapai jumlah l, yaitu dengan mengambil sampel secara acak dari data latihan variabel independen yang dimiliki dengan ukuran kembali n.
b. Menggunakan contoh pada setiap bootstrap untuk membangun pohon hingga mencapai ukuran maksimum. Susun pohon berdasarkan data *bootstrap*. Setiap proses pemisahan memilih m < p sebagai variabel independen, dan pemisahan terbaik dilakukan pada tahap sub-setting secara acak.
c. Ulangi langkah 1-2 sebanyak l kali untuk membentuk sebuah hutan yang

terdiri dari pohon-pohon.

d. Melakukan prediksi berdasarkan hasil prediksi pada setiap dataset hasil *Bootstrapping* dengan menggunakan mayoritas suara untuk kasus klasifikasi.

14. Confusion Matrix

Confusion matrix adalah tabel yang berisi data uji yang telah diklasifikasikan dengan benar dan salah, *Confusion matrix* digunakan sebagai cara untuk mengevaluasi kinerja model yang telah dihasilkan. Terdapat empat istilah dalam *confusion matrix*, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

Berikut adalah penjelasan dari empat istilah tersebut:

True Positive (TP) : kasus dimana objek diprediksi sesuai (*Positif*) kelasnya, dan sebenarnya memang sesuai (*True*) kelasnya.

True Negative (TN) : kasus dimana objek diprediksi tidak sesuai (*Negatif*) kelasnya dan sebenarnya memang tidak sesuai (*True*) kelasnya.

False Positive (FP) : kasus dimana objek diprediksi sesuai (*Positif*) kelasnya, dan ternyata tidak sesuai (*False*) kelasnya.

False Negative (FN) : kasus dimana objek diprediksi tidak sesuai (*Negatif*) kelasnya, dan dan ternyata sesuai (*True*) kelasnya.

Untuk memudahkan pemahaman *confusion matrix* dapat dilihat pada Gambar 2.4 berikut:

Gambar 2.2 *Confusion Matrix*

Sumber: (Pradika et al., 2020:134)

Berdasarkan pendapat Pradika et al., (2020:134) *Confusion matrix* digunakan untuk melihat performa dari suatu model yang telah dibuat yaitu *accuracy, precision, recall,* dan *f1 score.*

Accuracy merupakan proporsi atau rasio dari jumlah prediksi yang benar dibandingkan dengan jumlah total data. Dalam hal ini, akurasi digunakan untuk menggambarkan seberapa banyak data objek yang telah diprediksi dengan benar sesuai dengan kelasnya dari keseluruhan data.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Precision dapat diartikan sebagai perbandingan antara hasil prediksi yang benar dengan keseluruhan hasil yang diprediksi sebagai positif. *Precision* mengukur seberapa banyak objek yang sesuai dengan kelas yang diprediksi dari keseluruhan objek yang diprediksi sebagai positif.

$$Precision = \frac{TP}{TP + FP}$$

Recall merupakan rasio antara jumlah prediksi yang benar dengan keseluruhan data yang memang benar. *Recall* menjelaskan seberapa banyak persentase objek yang diprediksi sesuai dengan kelasnya dibandingkan dengan

keseluruhan objek yang memang sesuai dengan kelasnya.

$$Recall = \frac{TP}{TP + FN}$$

F1 Score adalah sebuah metrik evaluasi yang mengkombinasikan rata-rata precision dan recall dengan pemberian bobot yang sama. Dalam *F1 Score*, nilai rata-rata *precision* dan *recall* digabungkan dan dihitung nilai *harmonic mean*-nya, sehingga memberikan nilai yang lebih representatif dari kedua metrik tersebut. Oleh karena itu, *F1 Score* dapat memberikan gambaran yang lebih baik tentang performa suatu model dalam mengklasifikasikan data.

$$F1\ Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)}$$

15. Python

Python adalah bahasa pemrograman yang dinamis dan berorientasi objek yang dapat digunakan untuk berbagai jenis pengembangan perangkat lunak, Bahasa pemrograman ini menyediakan dukungan yang kuat untuk integrasi dengan bahasa pemrograman dan alat bantu lainnya. *Python* dilengkapi dengan pustaka standar yang dapat diperluas dan mudah dipelajari hanya dalam beberapa hari, *Python* dirancang dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode, dan diklaim sebagai bahasa yang menggabungkan kemampuan dan kapabilitas dengan sintaksis kode yang sangat jelas serta fungsionalitas pustaka standar yang luas dan komprehensif (Nugroho et al., 2020:16).

16. Keras

Berdasarkan pendapat Alwanda et al., (2020:49) *Keras* adalah perangkat lunak jaringan terbuka yang dibuat menggunakan bahasa pemrograman *Python*,

Keras dapat dijalankan dengan *MXNet, Tensorflow, Deeplearning4j*, *theano,* atau *CNTK* untuk mempercepat eksperimen yang berhubungan dengan *deep learning*. Keras awalnya dibuat untuk proyek eksperimen bernama *Open-ended Neuro-Electronic Intelligent Robot Operating System*, yang ditulis dan dikembangkan oleh *Francois Chollet*, seorang insinyur di *Google*.

Keras adalah sebuah *library open-source* yang digunakan untuk membangun dan melatih model *neural network*. Keras menyediakan berbagai fitur yang memudahkan proses pembuatan model, termasuk *layer-layer neural network* yang telah diimplementasikan, fungsi aktivasi, *optimizer*, dan fungsi *loss function* yang dapat digunakan untuk menentukan tujuan pelatihan model. Keras juga menyediakan fitur untuk visualisasi performa model seperti *plot learning curves* dan *confusion matrix*.

17. Scikit-learn

Scikit-learn adalah pustaka *Python open source* dari algoritma pembelajaran mesin populer itu akan memungkinkan kami untuk membangun jenis sistem ini. Proyek ini dimulai pada tahun 2007 sebagai proyek *Google Summer of Code* oleh David Cournapeau. Akhir tahun itu, Matthieu Brucher mulai mengerjakan proyek ini sebagai bagian dari tesisnya. Pada tahun 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, dan Vincent Michel dari INRIA mengambil proyek ini kepemimpinan dan menghasilkan rilis publik pertama. Saat ini, proyek sedang dikembangkan dengan sangat aktif oleh komunitas kontributor yang antusias. Itu dibangun pada NumPy (http://www.numpy.org/) dan SciPy (http://scipy.org/), pustaka Python standar untuk komputasi ilmiah (Garreta Raul

and Moncecchi Guillermo 2013:6).

Sedangkan menurut (Hackeling, 2014:16) *Scikit-learn* sangat diminati dalam penelitian akademik karena memiliki API yang terdokumentasi dengan baik, mudah digunakan, dan fleksibel. Pengguna dapat dengan mudah mencoba berbagai algoritma *machine learning* hanya dengan mengubah beberapa baris kode saja. *Scikit-learn* juga mengemas beberapa implementasi-algoritma *machine learning* terpopuler seperti *LIBSVM* dan *LIBLINEAR*. Beberapa pustaka *Python* lainnya, seperti *NLTK*, juga mengemas *scikit-learn* sebagai pembungkus. Selain itu, *scikit-learn* juga menyediakan berbagai dataset yang siap digunakan, sehingga pengguna dapat fokus pada pengembangan algoritma daripada menghabiskan waktu untuk mengumpulkan dan membersihkan data.

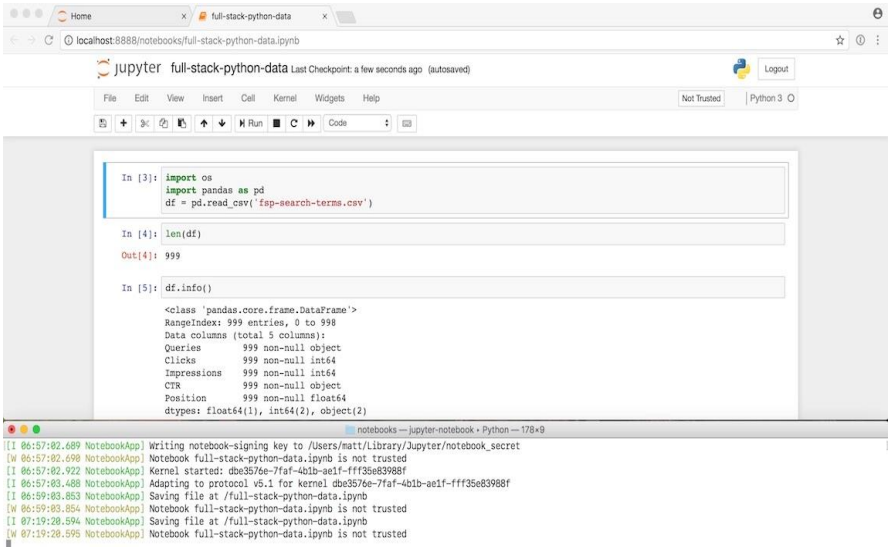
18. Tensorflow

TensorFlow adalah platform *open source end-to-end* untuk *machine learning* yang dibuat oleh tim *Google Brain*. *TensorFlow* memiliki perkakas, pustaka, dan sumber daya komunitas yang komprehensif dan fleksibel yang memungkinkan para peneliti dalam pengembangan *machine learning* dan tim pengembang dapat dengan mudah membangun dan menerapkan aplikasi yang didukung *machine learning*. Arsitekturnya yang fleksibel memungkinkan penerapan komputasi yang mudah di berbagai platform (CPU, GPU, TPU), dan dari perangkat seperti *desktop, cloud,* maupun perangkat *mobile* (Arifianto and Muhimmah 2021). Sedangkan menurut Hikmatia A.E & Ihsan Zul (2021:76) *TensorFlow* adalah platform komputasi untuk membangun model pembelajaran mesin yang menyediakan berbagai *toolkit* untuk membuat model pada berbagai

tingkat abstraksi, serta dapat menjalankan grafik pada berbagai *platform hardware* seperti CPU, GPU, dan TPU.

19. Jupyter Notebook

Jupyter Notebook (dulunya, *IPython Notebook*) adalah sebuah aplikasi yang umum digunakan pada bidang *Data Science*. Biasanya aplikasi ini digunakan untuk membuat dan berbagi dokumen yang berisi *Live code*, Persamaan (*Equalization*), *Visualizations*, dan teks penjelasan tentang dokumen tersebut. *Jupyter Notebook* merupakan aplikasi *server-client* yang mengizinkan kita untuk melakukan *editing* dan *running/compile* dokumen *notebook* melalui *web browser* (Id, Ibnu Daqiqil 2021:39).



Gambar 2.3 *Jupyter Notebook*

Sumber : (Id, Ibnu Daqiqil 2021: 39)

Dengan tool ini kita dapat melakukan banyak hal misalnya *data cleaning*, analisa data, pemodelan, bahkan pelatihan dan evaluasi model. Penggunaan *Jupyter* disukai karena di eksekusi per baris, sehingga kita dapat langsung melihat luaran

yang kita inginkan. Jika dibanding kan dengan cara *konvensional* dimana kita harus mengetik semua kode terlebih dahulu baru dapat menjalankannya, cara ini lebih sederhana dan mudah apalagi bagi yang sedang belajar *Machine Learning*.

20. Flask

Flask merupakan kerangka kerja aplikasi website yang menggunakan bahasa pemrograman python dan menggunakan dependensi *Werkzeug* dan *Jinja2*, Dalam tahap pengemabangan sistem *hoax news detection* ini *flask* berfungsi sebagai pengontrol dalam pengembangan sistem *hoax news detection* (Arbain et al., 2022:187).

Flask adalah sebuah kerangka kerja aplikasi web yang dibuat dengan bahasa pemrograman Python dan dirilis pertama kali pada tahun 2010 oleh *Armin Ronacher*. Flask dibangun dengan konsep sederhana dan fleksibel untuk membangun aplikasi web. *Framework* ini mengikuti paradigma arsitektur web *Model-View-Controller* (MVC) dan tidak memerlukan peralatan atau pustaka tambahan untuk menjalankan aplikasi web. *Flask* bergantung pada dua pustaka *Python*, yaitu *Werkzeug* untuk pengembangan aplikasi web dan *Jinja2* untuk pemrosesan template HTML.

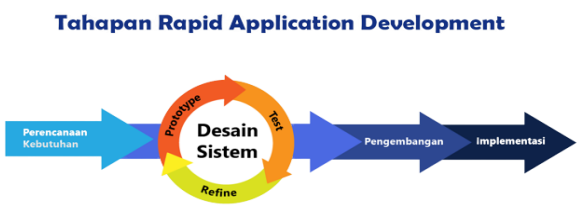
Flask adalah sebuah kerangka kerja aplikasi web yang sederhana dan fleksibel untuk membangun aplikasi web dengan menggunakan bahasa pemrograman *Python*. *Flask* memungkinkan pengembang untuk membangun aplikasi web yang ringan dan mudah di-maintain serta memperoleh fitur-fitur yang dibutuhkan melalui ekosistem ekstensi *Flask* yang kaya dan fleksibel.

21. RAD (Rapid Application Development)

Menurut Nur Fitrianingsih Hasan et al. (2023) Metode RAD yang muncul pada tahun 1990an bertujuan untuk mengatasi kekurangan metode desain terstruktur dengan mengadaptasi fase SDLC. Hal ini memungkinkan bagian tertentu dari sistem dikembangkan dengan cepat dan memungkinkan pengguna untuk lebih memahami sistem dan menyarankan perubahan. Sistem menyesuaikan agar sesuai dengan kebutuhan pengguna. Kebanyakan metodologi berbasis RAD mendorong analis untuk menggunakan metode dan alat komputasi khusus seperti alat untuk mempercepat tahap analisis, desain, dan implementasi. Seperti alat Computer aided software engineering (CASE).

Rapid Application Development (RAD) adalah metode pengembangan perangkat lunak yang menekankan siklus pengembangan perangkat lunak dalam waktu yang singkat. Menurut definisi tambahan, RAD adalah metode pengembangan perangkat lunak yang menggunakan pendekatan pengembangan sistem beorientasi objek, yang mencakup pengembangan perangkat lunak dan pengembangan perangkat lunak (I Kadek Dwi Gandika Su, S.T. et al. 2023).

Berikut ini tahapan metode ini seperti yang dapat dilihat pada gambar 2.4 dibawah ini :



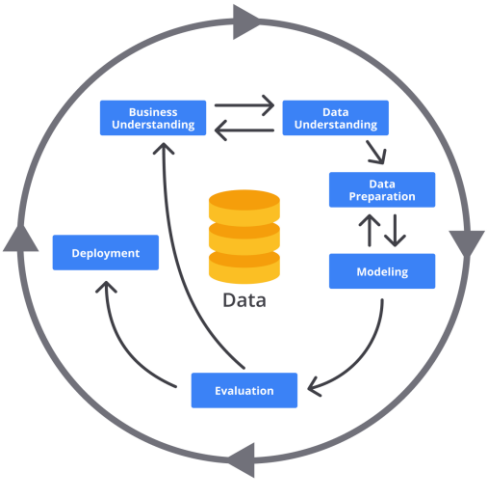
Gambar 2.4 RAD (*Rapid Application Development*)

Sumber : (I Kadek Dwi Gandika Su, S.T. et al. 2023)

Dari uraian diatas Metode RAD (Rapid Application Development) merupakan metode pengembangan sistem yang muncul pada tahun 1990-an untuk mengatasi kelemahan metode desain terstruktur. Metode RAD berusaha mempercepat proses pengembangan sistem dengan melibatkan pengguna secara aktif dan menggunakan alat bantu komputer.

22. CRISP-DM (Croos-Industry Standard Process)

Dalam penelitian ini metode *data mining* yang digunakan adalah metode *CRISP-DM* atau *Cross Industry Standard Process* untuk *Data Mining* yaitu merupakan standarisasi proses data mining sebagai strategi pemecahan masalah secara umum dari bisnis atau unit penelitian (Wibisono, Ulum, and Megawaty 2022:113). CRISP-DM memiliki alur penelitian yang terbagi dalam enam fase, berikut tahapan seperti yang dapat dilihat pada gambar 2.5 :



Gambar 2.5 Alur penelitian *CRISP-DM*

Sumber : (Brzozowska et al. 2023:87)

a) *Business Understanding Phase*, yaitu menentukan sasaran atau tujuan bisnis, menyiapkan strategi untuk mencapai tujuan, sampai memahami situasi bisnis.
b) *Data Understanding Phase*, yaitu mengumpulkan data dan kemudian mempelajari data untuk dapat mengetahui data seperti apa yang akan digunakan dalam penelitian.
c) *Data Preparation Phase*, yaitu data mentah yang sudah dikumpulkan sebelumnya akan dipersiapkan untuk dilakukan pembersihan data sehingga data siap digunakan untuk fase selanjutnya.
d) *Modeling Phase*, yaitu menentukan dan menerapkan teknik pemodelan yang paling sesuai untuk mendapatkan hasil yang optimal.
e) *Evaluation Phase*, yaitu melakukan evaluasi apakahmodel yang sudah diterapkan pada fase sebelumnya sudah mencapai tujuan yang ditetapkan diawal yaitu pada fase Business Understanding.
f) *Deployment Phase*, yaitu penyusunan laporan atau presentasi dari hasil dan informasi yang didapat sebelumnya yaitu pada tahap Evaluation.

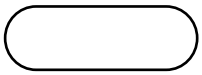
23. Flowchart

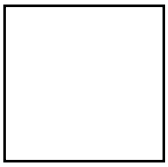
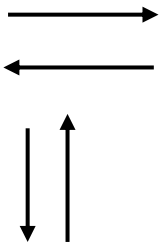
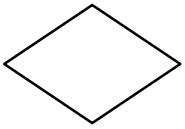
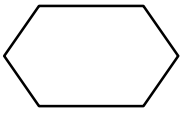
Flowchart merupakan bagan simbol yang memberikan penjelasan rinci tentang serangkaian proses dalam suatu program dan hubungan antara proses tersebut (instruksi) dan proses lainnya. *Flowchart* adalah representasi grafis dari serangkaian proses yang dilakukan oleh suatu program. Setiap penanda memiliki simbol pada diagram blok. *Flowchart* menampilkan langkah dan keputusan dalam melakukan sebuah proses dari suatu program. Setiap langkah digambarkan dalam

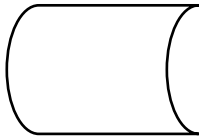
bentuk diagram dan dihubungkan oleh garis atau arah. *Flowchart* membantu pengguna untuk menentukan langkah-langkah dan fungsi proyek perangkat lunak yang melibatkan banyak orang pada saat yang bersamaan. *Flowchart* membuat program menjadi lebih jelas dan ringkas, sehingga mengurangi kemungkinan kesalahpahaman. Dalam dunia pemrograman, diagram alur adalah cara yang bagus untuk menghubungkan kebutuhan teknis dan non-teknis. Fungsi utama *flowchart* adalah untuk memberikan gambaran aliran perangkat lunak dari satu proses ke proses lainnya. Dengan menggunakannya alur program menjadi lebih mudah untuk dipahami oleh semua orang. Selain itu *flowchart* dapat menyederhanakan rangkaian prosedur agar memudahkan pemahaman terhadap informasi tersebut (Huda et al., 2021:7).

Simbol-simbol *flowchart* dapat dilihat pada tabel dibawah :

Tabel 2.1 Tabel Simbol-simbol *Flowchart*

No	Simbol	Nama	Keterangan
1.		Terminator	Simbol yang menyatakan awal atau akhir suatu program.
2.		*Input*/*Output*	Merepresentasikan *input* data atau *output* data yang diproses atau Informasi.

3.		Proses	Simbol yang menyatakan suatu proses yang dilakukan komputer.
4.		*Flow*	Simbol yang digunakan untuk menggabungkan antara simbol yang satu dengan simbol yang lain. Simbol ini juga disebut dengan *Conecting line*
5.		*Decision*	Simbol yang menunjukkan kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, yaitu ya atau tidak.
6.		*Preparation*	Simbol yang menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberikan nilai awal.
7.		Dokumen	Simbol yang menyatakan bahwa input berasal dokumen dalam bentuk fisik, atau output yang perlu dicetak.

7.		*Database*	Simbol yang menunjukkan penyimpanan data dalam sistem atau proses dan digunakan untuk mempresentasikan tempat atau lokasi penyimpanan data dalam suatu sistem atau database.

24. UML (Unifinied Modelling Languange)

UML adalah bahasa visual yang digunakan untuk pemodelan dan komunikasi mengenai sebuah sistem melalui diagram dan teks pendukung, Pemodelan UML mencakup beberapa jenis diagram seperti diagram *use case*, diagram *class*, diagram aktivitas, dan diagram urutan (Syarif & Nugraha (2020:65).

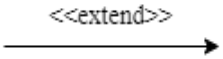
UML atau *Unified Modeling Language* merupakan sebuah bahasa yang digunakan untuk menentukan, visualisasi, konstruksi, dan dokumentasi suatu artifact dalam proses pembuatan perangkat lunak, Bahasa ini menyediakan notasi yang lengkap untuk membuat visualisasi model suatu sistem yang terdiri dari informasi dan fungsi, dan biasanya digunakan untuk pemodelan sistem komputer. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, tetapi juga dapat digunakan di hampir semua bidang yang memerlukan pemodelan (Febriani et al., (2020:125).

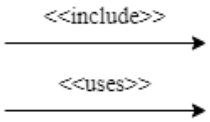
Berikut adalah penjelasan singkat tentang beberapa jenis diagram dalam *Unified Modeling Language* (UML) :

a. *Use case diagram*

Diagram ini digunakan untuk menggambarkan fungsionalitas sebuah sistem dari sudut pandang pengguna (*actor*) dan aktivitas yang dilakukan oleh sistem. *Use case diagram* sering digunakan dalam tahap awal perancangan sistem untuk memahami kebutuhan fungsional sistem dari sudut pandang pengguna. Tabel 2.1 adalah simbol-simbol yang dapat diterapkan dalam diagram *use case* :

Tabel 2.2 Simbol-simbol *use case diagram*

No.	Simbol	Nama	Keterangan
1.		Actor	Pengguna sistem atau yang berinteraksi secara langsung dengan sistem.
2.		*Assocation*	Interaksi yang terjadi terhadap *actor* dan *user* .
3.		*Extend*	Relasi tambahan *use case* terhadap *use case* lain.
4.		Generalization	Menunjukan hubungan kearah *use case* yang lebih umum.

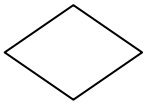
5.		*Uses/Include*	Relasi dua *use case, use case* yang ditambahkan membutuhkan tambahan untuk menjalankan *use case.*

b. *Activity diagram*

Diagram ini digunakan untuk menggambarkan urutan aktivitas dan alur logika dari suatu proses atau fungsi pada sebuah sistem. *Activity diagram* sering digunakan untuk menggambarkan proses bisnis, alur aplikasi, atau algoritma. Berikut adalah Tabel 2.3 yang menjabarkan simbol-simbol yang ada pada diagram aktivitas :

Tabel 2.3 Simbol-simbol *activity diagram*

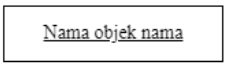
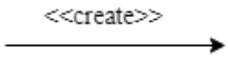
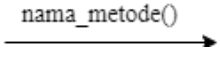
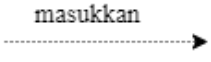
No.	Simbol	Nama	Keterangan
1.		Status Awal/akhir	Merupakan status awal dan akhir, setiap aktivitas diagram memiliki status awal dan status akhir

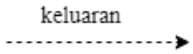
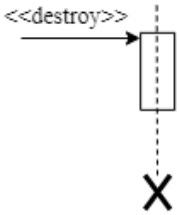
2.		Aktivitas	Merupakan kegiatan yang dilakukan sistem, yang dimulai dengan kata kerja.
3.		*Decision*	Merupakan hubungan untuk keputusan aktivitas yang memiliki lebih dari satu pilihan.
4.		*Join*	Merupakan hubungan jika satu atau lebih aktivitas menjadi satu.
5.		*Swimlane*	Merupakan pemisah organisasi bisnis yang memiliki tanggung jawab aktivitas yang terjadi.

c. Sequence diagram

Diagram ini digunakan untuk menggambarkan interaksi antara objek pada sebuah sistem dalam bentuk urutan pesan yang dikirimkan antara objek. Sequence diagram sering digunakan untuk memodelkan alur proses atau interaksi antara komponen pada sebuah sistem. Simbol-simbol *sequence diagram* dapat dilihat pada Tabel 2.4:

Tabel 2.4 Simbol-simbol *sequence diagram*

No.	Simbol	Nama	Keterangan
1.		*Actor*	Merupakan orang atau sistem atau proses diluar sistem yang dibuat, yang berhubungan dengan sistem yang dibuat.
2.		*Lifeline*	Garis hidup objek yang menerangkan kehidupan dari objek tersebut.
3.		Objek	Merupakan interaksi pesan yang dilakukan oleh objek.
4.		Waktu aktif	Menandakan interaksi obejek.
5.		Pesan tipe *create*	Pernyataan suatu objek membuat objek lain.
6.		Pesan *tipe call*	Merupakan penyataan satu objek memanggil metode atau proses pada objek lain.
7.		Pesan tipe *send*	Merupakan penyataan bahwa objek mengirimkan informasi atau masukan atau data ke objek lain.

8.		Pesan tipe *return*	Pernyataan objek menjalankan suatu perintah dan memberi keluaran ke objek tertentu.
9.		Pesan tipe *destroy*	Pernyataan objek yang dimatikan oleh objek lainnya.

25. Blackbox Testing

Pengujian fungsional atau *black box testing* adalah metode pengujian perangkat lunak yang digunakan untuk menguji fungsionalitas perangkat lunak tanpa memperhatikan detail struktur internal kode atau program yang digunakan. Dengan metode ini, pengguna dapat menguji perangkat lunak dari luar tanpa perlu mengetahui bagaimana perangkat lunak tersebut diimplementasikan secara internal. Pengujian fungsional umumnya digunakan dalam tahap pengembangan perangkat lunak untuk memastikan bahwa fungsi-fungsi yang diharapkan dari perangkat lunak berjalan dengan baik. Metode ini sangat berguna untuk memastikan kehandalan dan kualitas perangkat lunak sebelum dirilis ke publik (Hardika et al. 2024).

B. Kajian Empiris

Sebagai referensi untuk menyelesaikan penelitian ini, penulis menggunakan penelitian pengelolaan kualitas air yang berbeda sebelumnya. Penelitian tersebut dilakukan dengan menggunakan berbagai jenis data dan metode klasifikasi yang

disesuaikan dengan kebutuhan peneliti.

Penelitian sebelumnya telah menunjukkan bahwa penggunaan sensor berbasis IoT dalam sistem akuaponik mampu memberikan data yang akurat dan real-time tentang berbagai parameter kualitas air, seperti pH, suhu, kadar oksigen terlarut, dan konduktivitas listrik. Misalnya, penelitian oleh (Alfia et al. 2021) yang berjudul "Sistem Monitoring Kualitas Air Pada Sistem Akuaponik Berbasis *IoT*" menunjukkan bahwa sistem pemantauan berbasis *IoT* dapat mendeteksi perubahan kualitas air dengan cepat dan membantu dalam pengambilan keputusan untuk menjaga kesehatan ikan dan tanaman.

Persamaan penelitian yang dilakukan oleh (Wahy dan Muhamad Bahrul 2021) dengan kajian ini terfokus Pada pengujian alat sistem *monitoring* kualitas air pada budidaya ikan hias air tawar, dimana sistem ini dapat mendeteksi kualitas air dan dapat menampilkan hasil monitoring pada *smartphone android*, *Mikrokontroler NodeMCU ESP32* dapat mengirim data sesuai nilai sensor ke *firebase*, Dapat diperoleh sistem monitoring kualitas air, dimana sistem dapat mendeteksi pH air menggunakan PH-4502C sebagai pendeteksi nilai pH dengan *action* pompa ph *up* atau ph *down* akan menyala, *turbidity SENO189* sebagai pendeteksi nilai NTU dengan *action buzzer* akan menyala jika air keruh, *DS18b20* sebagai pendeteksi suhu pada air. Serta *monitoring* yang dapat dilakukan melalui aplikasi *android*.

Pada penelitian yang dilakukan oleh (Putri et al. 2024) Dari penelitian yang telah dilakukan membuahkan hasil akurasi algoritma Decision Tree jauh lebih baik dibandingkan dengan algoritma SVM dalam dataset stunting yang digunakan

dalam riset ini. Pada algoritma Decision Tree didapatkan akurasi sebesar 96.15% sedangkan pada algoritma SVM akurasi yang didapatkan memiliki selisih 33.67% dengan algoritma Decision Tree yaitu sebesar 62.48%. Apabila pada penelitian selanjutnya terdapat pola data yang hampir sama dapat menggunakan Decision Tree untuk pengolahannya.

C. Kerangka Berfikir

Kerangka berpikir merupakan suatu model abstrak yang digunakan untuk membantu peneliti dalam merancang suatu penelitian. Kerangka berpikir tersebut dapat meliputi berbagai aspek seperti perumusan masalah, pendekatan penelitian, pengembangan sistem, dan hasil penelitian. Pada sub bab ini, akan dijelaskan kerangka berpikir yang berkaitan dengan jenis-jenis air dan solusi yang ditawarkan untuk mengatasinya.

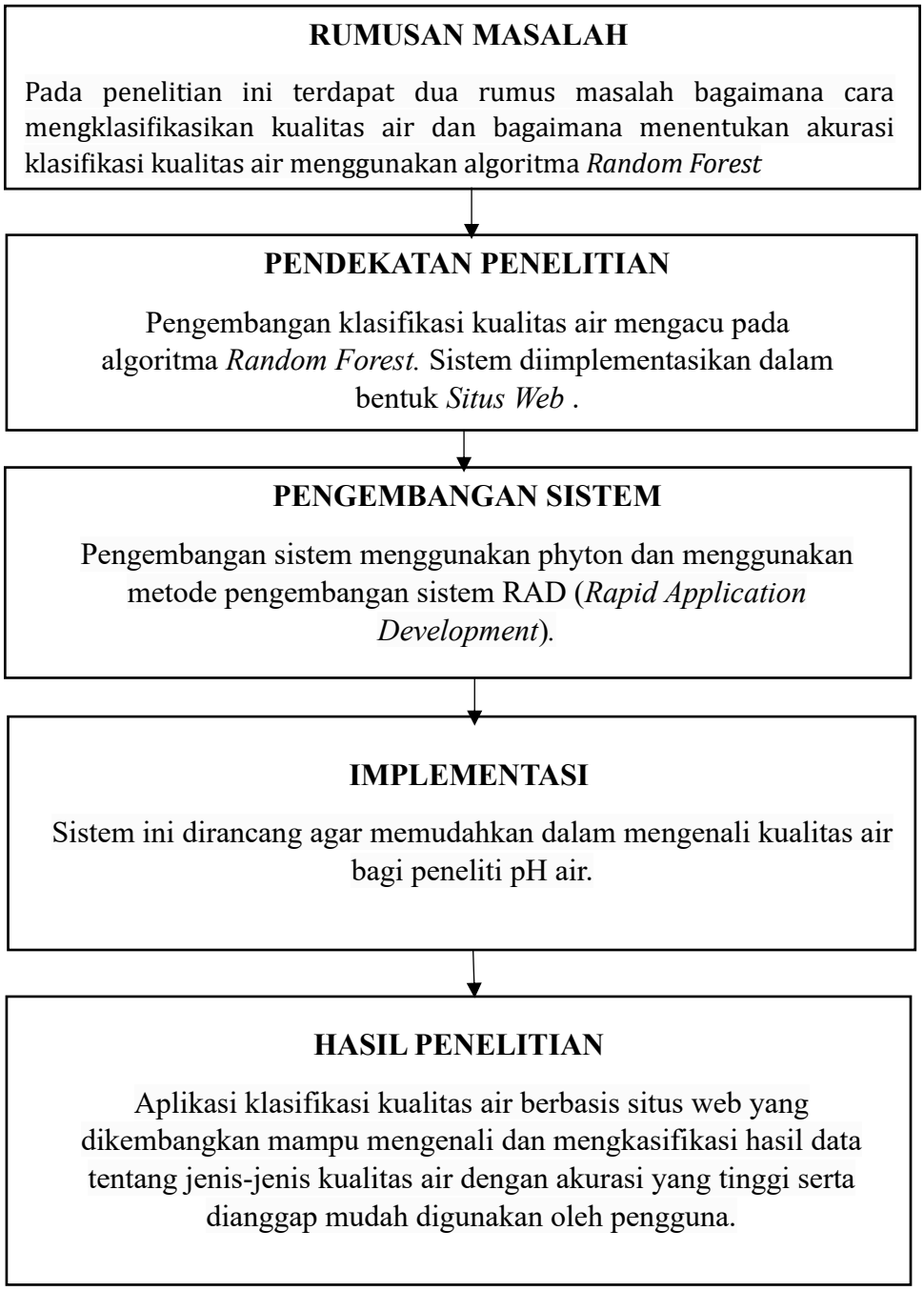
Rumusan masalah pada penelitian ini adalah variasi pada jenis-jenis kualitas air. Hal ini cukup menyulitkan bagi seseorang yang ingin mengetahui jenis-jenis kualitas air untuk penggunaan air dalam kehidupan sehari-hari. Dalam konteks ini, penelitian bertujuan untuk mengembangkan suatu sistem yang dapat membantu seseorang dalam mengenali jenis-jenis kualitas air dengan lebih mudah. Pendekatan penelitian yang digunakan dalam penelitian ini adalah perancangan dan pembangunan klasifikasi kualitas air mengacu pada algoritma *random forest*. Sistem dibangun dalam bentuk aplikasi website yang dapat diakses secara *online* oleh pengguna. Selain itu, pengembangan sistem dalam penelitian ini menggunakan pemodelan pengembangan sistem RAD (*Rapid Application Development*). Pendekatan ini dipilih karena memungkinkan pengembangan

sistem yang lebih responsif terhadap perubahan kebutuhan pengguna.

Setelah tahap perancangan dan pembangunan, dilakukan implementasi sistem. Sistem ini dirancang agar memudahkan seseorang dalam mengenali atau mengetahui jenis-jenis kualitas air yang baik untuk digunakan. Implementasi dilakukan dengan menguji coba aplikasi pada sejumlah pengguna untuk mendapatkan umpan balik terkait kegunaan dan sistem. Hasil penelitian menunjukkan bahwa aplikasi klasifikasi kualutas air yang dikembangkan mampu mengenali dan mengklasifikasikan jenis-jenis air dengan akurasi yang tinggi. Selain itu, aplikasi ini juga dianggap mudah digunakan dan dapat membantu pengguna dalam mempelajari kualitas air.

Dalam kesimpulan, penelitian ini mampu mengembangkan suatu sistem yang dapat membantu seseorang dalam mengenali jenis-jenis kualitas air dengan lebih mudah. Hasil penelitian menunjukkan bahwa aplikasi klasifikasi kualitas air yang dikembangkan mampu mengenali dan mengklasifikasikan jenis-jenis air dengan akurasi yang tinggi serta dianggap mudah digunakan oleh pengguna.

Gambar kerangka berpikir dalam penelitian ini dapat dilihat pada gambar 2.3 dibawah ini :



Gambar 2.6 Kerangka Berpikir