Integrasi *Flower Care Sensor* dengan Logika Fuzzy-Random Forest sebagai Sistem Monitoring Tanaman dalam Kondisi Ekstrim dengan *Multi-platform Dashboard*

Tugas Akhir

diajukan untuk memenuhi salah satu syarat

memperoleh gelar sarjana

dari Program Studi S1 Informatika

Fakultas Informatika

Universitas Telkom

1301228479

Prasidya Pramadresana Saftari

Program Studi Sarjana S1 Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2024

Lembar Persetujuan

Integrasi *Flower Care Sensor* dengan Logika Fuzzy-Random Forest sebagai Sistem Monitoring Tanaman dalam Kondisi Ekstrim dengan *Multi-platform Dashboard*

Flower Care Sensor and Fuzzy-Random Forest Integration as Plant Monitoring System on Extreme Condition with Multi-platform Dashboard

NIM: 1301228479

Prasidya Pramadresana Saftari

Tugas akhir ini telah diterima dan disahkan untuk memenuhi sebagian syarat memperoleh gelar pada Program Studi Sarjana S1 Informatika

Fakultas Informatika

Universitas Telkom

Bandung, 1 Desember 2023

Menyetujui

Pembimbing I,

Dr. Bayu Erfianto, S.Si., M.Sc.
NIP: 05730058

Ketua Program Studi
Sarjana S1 Informatika,

Dr. Erwin Budi Setiawan, S.Si., M.T.
NIP: 00760045

Daftar Isi

Daftar Gambar

Daftar Tabel

ABSTRAK

Tugas Akhir ini membahas integrasi *Flower Care Sensor* yang memanfaatkan logika *Fuzzy-Random Forest* sebagai sistem pemantauan tanaman dalam kondisi ekstrim dengan menggunakan aplikasi klien berupa *multi-platform dashboard*. Tujuan utama dari tugas akhir ini adalah meningkatkan efektivitas pemantauan kondisi tanaman, terutama dalam menghadapi keadaan dan variabel ekstreme, baik lingkungannya maupun nutrisinya. Sensor tanaman berupa *Flower Care* digunakan untuk mengukur parameter lingkungan, sedangkan implementasi *Fuzzy-Random Forest* secara *sequential* digunakan untuk menginterpretasikan informasi sensor dengan lebih akurat dan responsif. Implementasi *multi-platform dashboard* memungkinkan petani untuk memantau kondisi tanaman secara *real-time* dan mengambil tindakan yang diperlukan melalui berbagai perangkat seperti *smartphone* ataupun *tablet*. Hasil pengujian tugas akhir membuktikan integrasi *monitoring system* ini dapat mendeteksi dan menentukan keadaan tanaman dalam kondisi ekstrem maupun optimal berdasarkan parameter ukur temperatur, konduktivitas listrik pada tanah, kelembaban tanah, serta intensitas cahaya yang didapatkan.

Kata Kunci: *Fuzzy Logic*, *IoT Integration, Sensor Fusion, Smart System*

Bab I
PENDAHULUAN

1.1. Latar Belakang

Dalam beberapa tahun terakhir, pertanian modern semakin menggantikan metode konvensional dengan teknologi canggih untuk meningkatkan produktivitas dan efisiensi [1]. Salah satu aspek penting dalam pertanian adalah pemantauan kondisi tanaman, terutama Tomat Ceri untuk memastikan kesehatan dan pertumbuhan yang optimal. Tanaman tomat ceri ini sangat rentan apabila berada di kondisi sekitar yang ekstrim.

Pada kondisi ekstrim, seperti suhu dan kelembaban yang tidak sesuai, paparan cahaya yang berlebih, tanaman dapat mengalami stres, penyakit, dan kematian [2]. Oleh karena itu, diperlukan sistem pemantauan yang dapat memberikan informasi secara *real-time* tentang kondisi tanaman agar petani dapat mengambil tindakan yang diperlukan pada saat terjadinya kondisi ekstrim. Namun, perlu diketahui bahwa gangguan pada tanaman tidak hanya dipengaruhi oleh lingkungannya saja, nutrisi juga berperan dalam kesehatan pertumbuhan tanaman. Maka dari itu, digunakanlah *Flower Care Sensor* yang memantau perubahan nilai intensitas cahaya, temperatur udara, kelembaban tanah, dan kesuburan tanah melalui (*Electrical Conductivity* dan pH) pada area tanam [3]. Tanaman seperti tomat ceri ini adalah sayuran basah yang kesehatannya sangat bergantung pada parameter yang diukur oleh sensor tersebut, maka dari itu pengujian ini dilakukan untuk menguji keadaannya dimulai pada saat *early-stage*.

Penelitian dan pengujian tugas akhir ini terinsirasi oleh riset yang pernah dilakukan oleh salah satu jurnal. Dalam jurnal tersebut, penulis mengeksplorasi integrasi sensor tanaman untuk memantau parameter lingkungan media tanam tumbuhan [4]. Perbedaannya terdapat di integrasi *Flower Care Sensor* dengan bantuan pemodelan *Fuzzy-Random Forest* dengan metode *sequential modelling*, sehingga selain memonitor kondisi lingkungan, juga melakukan klasifikasi kondisi lingkungan tanaman tersebut. Dengan demikian, tugas akhir ini dapat memberikan kontribusi untuk pengembangan sistem pertanian pintar.

1.2. Perumusan Masalah

Rumusan masalah pada tugas akhir ini adalah sebagai berikut.

1. Bagaimana sensor dapat diintegrasikan untuk mengukur parameter lingkungan tanam?
2. Bagaimana sistem monitoring yang dibangun dapat menentukan kondisi tanaman berdasarkan parameter yang diukur oleh sensor?
3. Bagaimana pengguna dapat memantau perubahan parameter kondisi tanaman secara *real-time*?

1.3. Tujuan

Tugas akhir ini memiliki beberapa tujuan akhir sebagai berikut.

1. Membuat sistem yang dapat memonitoring tanaman berdasarkan parameter suhu, kelembaban tanah, intensitas cahaya, dan konduktivitas listrik pada tanah dengan memanfaatkan fusi *Flower Care Sensor.*
2. Membuat *early system* yang mendeteksi kondisi tanaman melalui parameter lingkungan ekstrem, optimal, *caution* dengan memanfaatkan model prediktif *Fuzzy-Random Forest.*

1.4. Batasan Masalah

Tugas akhir memiliki batasan riset serta kebutuhan sebagai berikut.

1. Pengujian hanya dilakukan menggunakan tanaman Tomat Ceri (*Solanum lycopersicum var. cerasiforme*).
2. Proses pengambilan data dilakukan selama satu bulan penuh pada saat masa pancaroba untuk mendapatkan variasi data.
3. Proses pemodelan *Fuzzy-Random Forest* hanya menggunakan sampel yang didapatkan pada pagi sampai sore hari untuk menghasilkan dataset yang bervariasi.
4. Parameter yang diukur hanya intensitas cahaya (lux), temperatur (ºC), kelembaban tanah (%), dan electrical conductivity (µS/cm).
5. Aplikasi *multi-platform dashboard* atau *multi-platform mobile app* yang dibangun adalah aplikasi dapat dijalankan dengan ponsel Android dan iOS.

1.5. Hipotesis

Hipotesis tugas akhir ini adalah bahwa integrasi *Flower Care Sensor* dengan *Fuzzy-Random Forest* sebagai sistem pemantauan tanaman dalam kondisi ekstrem dengan *multi-platform mobile dashboard* dapat meningkatkan efektivitas pemantauan pertanian. Hipotesis ini didasarkan pada keyakinan bahwa penggunaan *Flower Care Sensor* akan memberikan parameter ukur dari lingkungan tanam, sedangkan pemanfaatan logika Fuzzy dan *ensemble Random Forest model* meningkatkan kemampuan sistem untuk menganalisis data dari sensor yang bervariasi pada saat kondisi ekstrem.

1.6. Rencana Kegiatan

Adapun langkah-langkah yang dilakukan dalam penyusunan tugas akhir ini.

1. **Pengumpulan Kajian Pustaka**, yaitu menelaah literatur terkait metode dan proses yang akan digunakan pada tugas akhir ini, seperti alat yang digunakan, parameter yang diuji dan teknologinya.
2. **Pengumpulan Data**, yaitu memperoleh data dari sistem ataupun perangkat yang dirancang. Pada kegiatan ini perangkat sensor dan mikrokontroler sudah mulai digunakan.
3. **Analisis Data**, yaitu menganalisis dan mengolah dataset yang didapatkan dari hasil pengumpulan data, pada tahap ini *Fuzzy logic* dan *Random Forest model* mulai digunakan untuk mengklasifikasi tiap-tiap hasil pengujian.
4. **Hasil dan Kesimpulan**, yaitu data hasil analisis dan model *Fuzzy logic* dan *Random Forest model* akan dicatat dan disimpulkan ke dalam laporan tugas akhir ini.
5. **Pembuatan Buku Tugas Akhir**, menyusun buku laporan akhir yang mencakup keseluruhan aspek Tugas Akhir mulai dari pendahuluan sampai hasil dan temuan.

1.7. Jadwal Kegiatan

Table 1. 1 *Timeline* Kegiatan TA

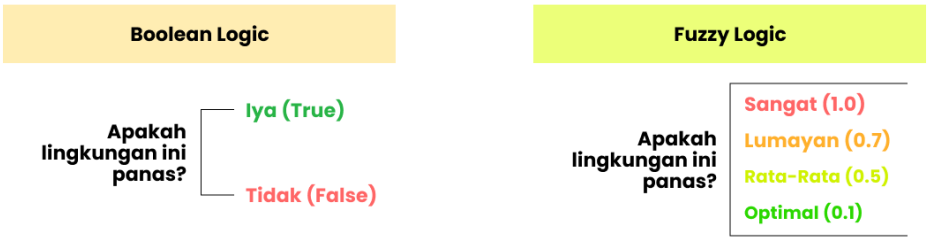
Kegiatan	Bulan				
	1	2	3	4	5
Pengumpulan Kajian Pustaka					
Pengumpulan Data					
Pemodelan Random Forest Model					
Analisis dan Hasil					
Pembuatan Buku Tugas Akhir					

Bab II

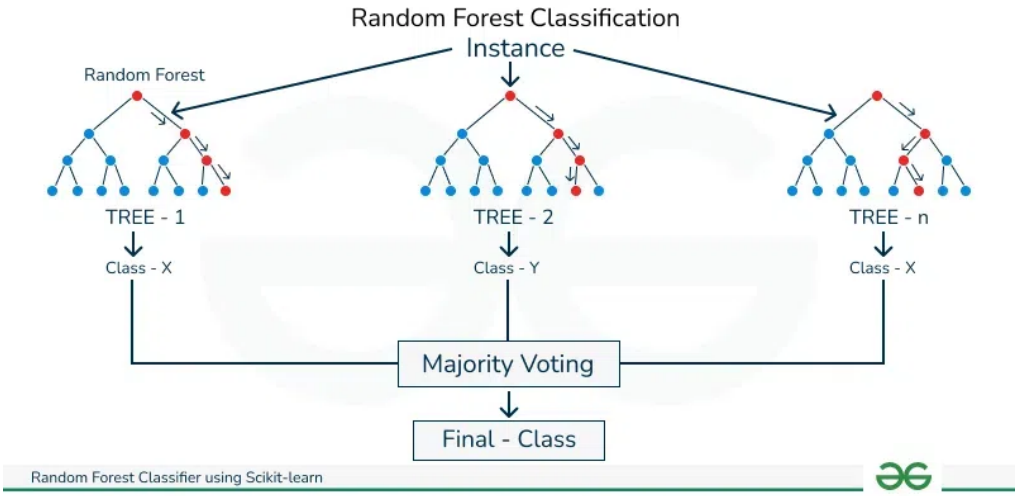
KAJIAN PUSTAKA

2.1. Fuzzy-Random Forest

Metode yang menjadi fondasi dan digunakan dalam *learning* di pengujian ini untuk menganalisis data adalah *Fuzzy*. *Fuzzy* secara bahasa diartikan sebagai kabur atau samar yang artinya suatu nilai dapat bernilai benar atau salah secara bersamaan. Dalam *fuzzy* dikenal derajat keanggotan yang memiliki rentang nilai tidak hanya false hingga true seperti boolean [5]. Dalam fuzzy logic, istilah *Fuzzy* ini berarti ketidakpastian atau kekurangan dalam suatu informasi. Konsep ini mencoba menangkap sifat fleksibel atau kabur dalam cara manusia berpikir. Contoh dari konsep Fuzzy dapat dilihat saat kita menyatakan apakah sebuah suhu dingin atau panas. Logika ini menggambarkan bahwa kita mengklasifikasikan 2 hasil yang mungkin dapat diperoleh dari 1 variable, yaitu suhu. Kelebihan logika *Fuzzy* adalah kemampuannya untuk menangani ketidakpastian dan kompleksitas saat dihadapkan dengan berbagai masalah, seperti kendali otomatis, sistem pemantauan, dan kecerdasan buatan.



Gambar 2. 1 Perbedaan Boolean dan Fuzzy



Gambar 2. 2 Cara kerja sederhana Random Forest

Sumber: https://www.geeksforgeeks.org/random-forest-classifier-using-scikit-learn/

Sedangkan *Random Forest* merupakan metode *ensemble machine learning*. *Random Forest* (RF) adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon (tree) dengan melakukan *training* pada sampel data yang dimiliki [6]. Salah satu algoritma *Random Forest* yang cepat dan efisien ini disebut *ensemble model*, yaitu model yang sifatnya menggabungkan beberapa *machine learning model* lainnya. Secara sederhana, basis dari RF model adalah menggunakan beberapa *decision tree* atau pohon keputusan yang melakukan seleksi klasifikasi atau regresi, maka dari itu diberi nama *forest* atau hutan. Banyaknya jumlah pohon menentukan seberapa akurat model dapat mencapai keputusan akhir dan penentuan hasil akhir keputusan dilakukan berdasarkan hasil voting dan *tree* yang terbentuk. Selain menggunakan beberapa pohon keputusan, pelatihan *Random Forest* juga umumnya mengimplementasikan metode *bagging*. Metode *bagging* ini adalah kombinasi model pembelajaran untuk meningkatkan hasil keseluruhan.

Ada beberapa cara untuk menggunakan integrasi Fuzzy-Random Forest, seperti menggunakan metode *sequential modelling* maupun *decision tree fuzzification*. Pada penelitian tugas akhir ini, metode yang digunakan adalah *sequential*. Proses ini menjelaskan bahwa saat *modelling* terdapat *sequence* atau tahapan yang saling berurutan, di mana logika Fuzzy dilakukan untuk memberikan *output label* setelah melakukan *Fuzzification* terhadap empat parameter ukur. Proses ini diikuti oleh pemodelan *Random Forest Classifier* model yang dilatih berdasarkan dataset yang sudah difiltrasi serta memiliki *output label* tersebut. Berbeda dengan metode *decision tree fuzzification*, proses logika Fuzzy akan melakukan *fuzzification* pada saat membuat Decision Tree untuk

mengendalikan ketidakpastian secara lebih baik [7]. Namun, alasan metode kedua ini tidak dilakukan pada penelitian tugas akhir ini adalah dataset yang dimiliki tidak ada *output label*, sehingga dilakukan metode *sequential* yang menggunakan logika Fuzzy untuk membantu proses *labeling* dan yang memberikan *borderline* berupa *Fuzzy Rule*.

2.2. *Electrical Conductivity* (EC) dan *pH Level*

EC adalah ukuran dari jumlah garam yang terlarut dalam larutan nutrisi atau kepekatan pupuk dalam larutan hidroponik. Nilai EC dalam larutan mempengaruhi metabolism tanaman, yaitu dalam hal kecepatan fotosintesis, aktivitas enzim, dan potensi penyerapan ion-ion oleh akar. EC dari tiap tumbuhan berbeda nilainya dan nilai tersebut disebut nilai EC optimal. Misalnya tanaman selada, selada dengan EC 1800 µs/cm memiliki metabolisme terbaik, sedangkan saat di EC 2100 µs/cm tanaman akan tumbuh semakin tinggi namun pada hari ke-5 sampai ke-35 akan mulai rontok dan layu [8].

NAMA SAYURAN	pH	PPM
LOBAK	6.0-7.0	840-1540
SELADA	6.0-7.0	560-840
CAULIFLOWER	6.5-7.0	1050-1400
PAK CHOI	7	1050-1400
KETIMUN	5.5	1190-1750
EGGPLANT	6	1750-2450
TOMAT	6.0-6.5	1400-3500
SAWI PAHIT	6.0-6.5	840-1680
STRAWBERRY	6	1260-1540
KANGKUNG	5.5-6.5	1050-1400
SAWI	5.5-6.5	1050-1400
KAILAN	5.5-6.5	1050-1400
BAYAM	6.0-7.0	1260-1610
BAWANG PUTIH	6.O	980-1260
SELEDRI	6.5	1260-1680
CABE	6.0-6.5	1260-1540
WORTEL	6.3	1120-1400
marjoram	6	1120-1400
Peterseli	5,5-6,0	560-1260
Peas	6.0-7.0	980-1260
Jagung manis	6	840-1680
Kentang	5.0-6.0	1400-1750

Gambar 2. 3 List Kadar pH dan Nutrisi Tanaman

Sumber: http://petaniteguh.blogspot.com/2014/03/tabel-untuk-ukuran-ppm-dan-ph-hidroponik.html

Table 2. 1 EC Optimal Tomat Ceri

Sumber: https://ohioline.osu.edu/factsheet/hyg-1437

***Stage* Tomat Ceri**	***Electrical Condcutivity* (dS/m)**	***Electrical Conductivity* (mS/cm)**
Stage 1 – Kotiledon	1.8 - 2.0	1800 - 2000
Stage 2 – Tanaman muda	2.0	2000
Stage 3 – Berbunga	2.4	2400

Selain EC, nutrisi dan kondisi pada tanaman yang harus diteliti adalah *pH level* atau kadar pH. Kadar pH (*Potential of Hydrogen*) adalah derajat keasaman yang digunakan untuk menyatakan tingkat keasaman atau kebasaan yang dimiliki oleh suatu larutan [9]. Kadar pH sendiri memiliki nilai 0 sampai 14, semakin mendekati 0 artinya tingkat keasamannya tinggi. Semakin pH mendekati tengah atau sekitar 7, maka disebut netral. Selanjutnya apabila mendekati nilai 14, maka disebut basa. Setiap tanaman memiliki kadar pH optimal ini yang berbeda-beda.

Gambar 2.3 adalah daftar kadar pH dan PPM atau kesuburan tanah. Tabel 2.1 merupakan rangkuman detail tahapan tanaman dan juga EC optimal yang dibutuhkan oleh Tomat Ceri. Penelitian tugas akhir ini menggunakan Tomat Ceri yang masih dalam *early stage*, yaitu pada tahap Stage 2.

2.3. Flower Care Sensor

Flower Care adalah sensor yang terhubung melalui Bluetooth ke sebuah perangkat untuk memantau tanaman yang diciptakan oleh perusahaan teknologi Beijing, bernama *HuaHuaCaoCao Plant Technology Co*. Sensor ini mengukur intensitas cahaya, suhu di sekitar tanaman, kelembaban lingkungan, serta nutrisi dalam tanah [10]. Sensor ini biasanya ditanam di tanah tanaman yang ingin dipantau. Kemudian sensor ini mengumpulkan data mengenai kondisi tanah dan udara sekitar tanaman tersebut. Informasi ini dapat diakses oleh pengguna secara langsung melalui aplikasi *Flower Care Mobile App* pada ponsel Android ataupun iOS.

Gambar 2. 4 HuaHuaCaoCao Flower Care Sensor

Gambar 2.4 merupakan *Flower Care Sensor* yang digunakan untuk penelitian tugas akhir ini. Walaupun ukurannya kecil, *Flower Care Sensor* memiliki teknologi *Bluetooth Low Energy* (BLE), sehingga memberikan fleksibilitas bagi para pengembang *IoT Smart System* yang ingin membuat sistem pemantaunya sendiri. *Flower Care* ini dapat meng-*update* data secara otomatis setiap 30 menit sampai 1 jam, namun *update* data dapat diminta secara *on-demand* apabila diperlukan.

Keempat nilai yang diukur oleh sensor ini digunakan sebagai parameter ukur atau input yang akan diproses oleh model. Masing-masing parameter ukur juga harus diketahui *optimal borderline* atau rentang optimalnya untuk tanaman Tomat Ceri yang diteliti. Berdasarkan riset dan penelitian yang dilakukan sebelumnya, kriteria optimal setiap parameter dapat dituliskan seperti pada Tabel 2.2 berikut.

Table 2. 2 Nilai Optimal Parameter Tomat Ceri

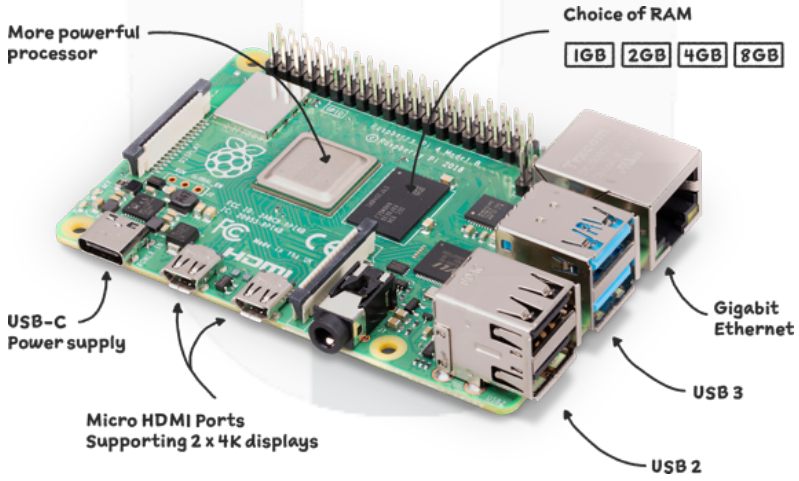
Sumber: https://www.mdpi.com/2073-4395/13/9/2417

Parameter Ukur	**Nilai Optimal**	**Nilai digunakan**	**Keterangan**
Temperature (ºC)	21 – 29 ºC	20 – 25 ºC	Toleransi adanya pengaruh suhu ruangan
Light Intensity (lux)	10000+ lux	4500 - 7500 lux	Tanaman masih muda, pengujian cahaya berlebih menyebabkan pertumbuhan terhambat, dan kebutuhan pengujian kondisi ekstrem

Soil Moisture (%)	45%+	30% – 55%	Toleransi cahaya yang lebih sedikit, juga kebutuhan pengujian kondisi ekstrem

2.4. *Raspberry Pi 4 Model B*

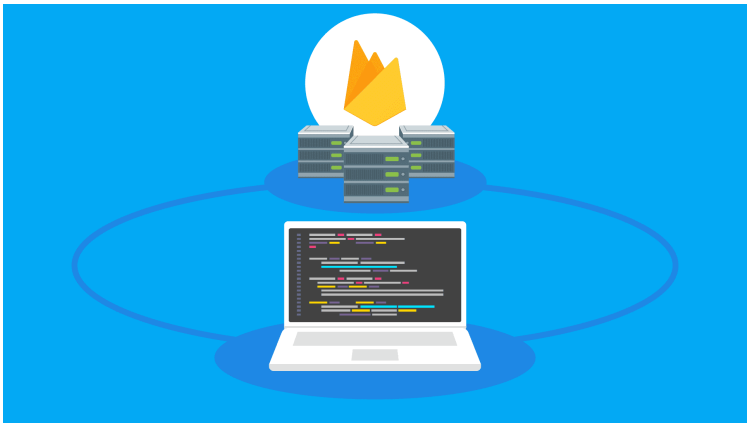
Raspberry Pi 4 Model B merupakan mikrokontroler yang sekaligus menjadi komputer kecil (*mini-PC*) ciptaan *Raspberry Foundation*. Komputer seukuran kartu bridge ini sebenarnya adalah sebuah papan sirkuit yang menggunakan prosesor *Quad core Cortex-A72 (ARM v8) 64-bit* berkemampuan 1.8 GHz. Komputer memiliki RAM sebesar 4GB, sehingga cukup bertenaga untuk perangkat kecil, seperti ponsel pintar saat ini. Perangkat ini juga menyediakan ruang untuk penyimpanan eksternal menggunakan MicroSD [11]. Khusus tugas akhir ini, untuk sistem operasi sekaligus penyimpanan berkelanjutan, *Raspberry Pi 4 Model B* ini menggunakan MicroSD berkapasitas 16GB. Dengan kapasitas penyimpanan 16GB ini, komputer kecil ini mampu untuk menyediakan beberapa ruang untuk menyimpan *cache* dan *swap memory* pada saat menggunakan *text editor* dan IDE untuk mengembangkan program Python ataupun terutama pada saat melakukan *streaming data* dari sensor.



Gambar 2. 5 Spesifikasi Raspberry Pi 4 Model B

Seperti yang digambarkan pada Gambar 2.3, *Raspberry Pi 4 Model B* ini merupakan bagian dari keluarga model *Raspberry Pi 4* yang memiliki fitur tambahan yaitu *dual HDMI support* sehingga cocok untuk penggunaan sehari-hari. Pada tugas akhir ini, perangkat Raspberry akan digunakan sebagai alat pertukaran informasi dan otak dari pemrosesan data. Data dari sensor akan langsung diteruskan ke perangkat ini menggunakan koneksi *Bluetooth Low Energy*. Selanjutnya, *Raspberry Pi 4 Model B* akan mengirim data yang didapat ke *Firebase Realtime Database.*

2.5. *Firebase Realtime Database*



Gambar 2. 6 Firebase Realtime Database

Firebase *Realtime Database* atau disingkat Firebase RTDB adalah basis data alternatif berbasis *NoSQL* milik Google yang di-*hosting* di *cloud*, sehingga pengembang tidak perlu lagi memasang server. Firebase RTDB ini dapat menyimpan, menghubungkan serta menyinkronkan data dengan seluruh pengguna secara langsung (*real time*). Selain kemampuannya dalam menyinkron data secara *real time*, pengembang dapat mengimplementasikannya dengan mudah ke semua perangkat lunak, tanpa perlu mempertimbangkan bahasa pemrograman yang digunakannya.

Dalam penelitian tugas akhir ini Firebase RTDB digunakan untuk tiga proses pengembangan, yaitu pada saat perangkat Raspberry mengirim data yang didapat dari sensor, saat melakukan data *preprocessing* model, serta pada saat pengembangan aplikasi dashboard *multi-platform mobile app*. Pengumpulan dan pengiriman data menggunakan Python yang ditanam dalam perangkat Raspberry. Seluruh data yang terkumpul di *Firebase Realtime Database* ini kemudian digunakan sebagai dataset oleh RF model yang dikembangkan menggunakan Scikit-Learn Python. Selanjutnya model yang di-*hosting* sebagai API digunakan untuk menampilkan data dan prediksi secara *real time* pada aplikasi yang dikembangkan menggunakan Flutter.

2.6. *Bluetooth Low Energy*

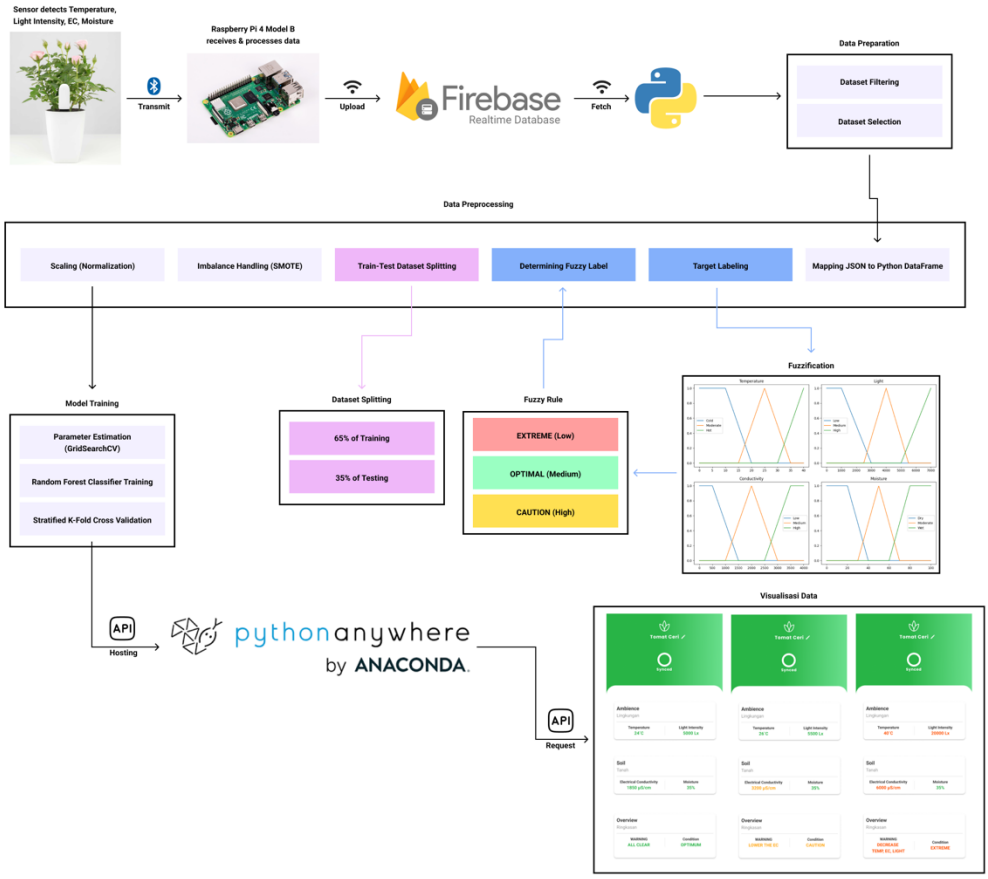
Bluetooth Low Energy (BLE) adalah salah satu standar komunikasi nirkabel yang dirancang dengan penggunaan daya rendah untuk perangkat pintar. Teknologi ini memungkinkan perangkat untuk mentransmisikan data secara efisien. Teknologi ini sangat cocok untuk implementasi yang tidak menggunakan daya listrik besar, seperti perangkat *IoT (Internet of Things)*, sensor-sensor kecil, dan perangkat *wearable* ataupun *smartwatch* [12].

Gambar 2. 7 Beragam Perangkat BLE

Dalam implementasinya, BLE sering digunakan dalam berbagai perangkat dan sistem, termasuk perangkat olahraga dan aplikasi rumah pintar. Selain digunakan untuk perangkat IoT dan *Smart System*, BLE juga teknologi yang cocok untuk penelitian. Bahkan, pada beberapa jurnal membandingkan BLE dengan teknologi Wi-Fi dan Zigbee, menyoroti kelebihan dan kelemahan masing-masing dalam konteks pengaplikasian dan permasalahan tertentu [13]. Sama seperti *Bluetooth* klasik yang umum digunakan oleh perangkat yang dapat menggunakan sumber tenaga lebih besar, BLE memiliki kemampuan komunikasi nirkabel jarak dekat yang memudahkan penggunanya. Namun, terdapat beberapa perbedaan antara *Bluetooth* LE dengan *Bluetooth* klasik selain dari tahun penemuannya. Jarak optimal BLE yang sekitar 5 – 10 meter sedangkan *Bluetooth* klasik sekitar 10 – 100 meter, kecepatan BLE sekitar 25 kbps dibandingkan 0,5 – 2 Mbps, serta maksimal perangkat yang terkoneksi di BLE tidak dibatasi.

Bab III
PERANCANGAN SISTEM

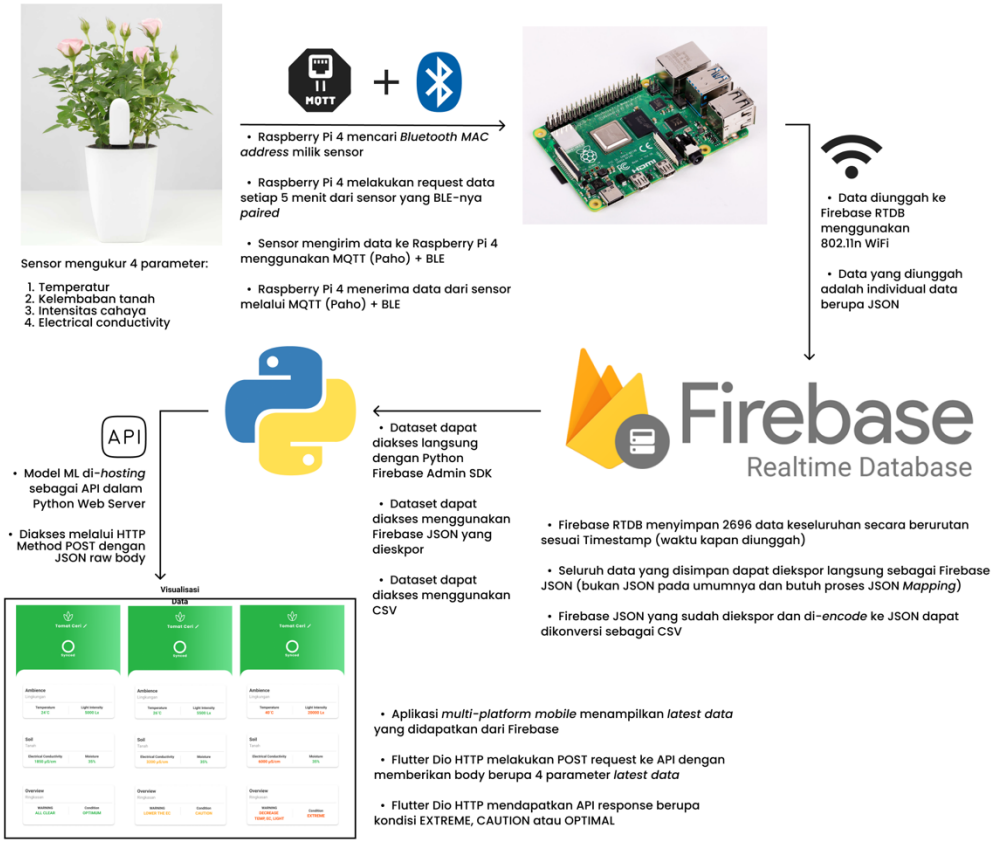
3.1 Deskripsi Umum



Gambar 3. 1 Arsitektur Sistem

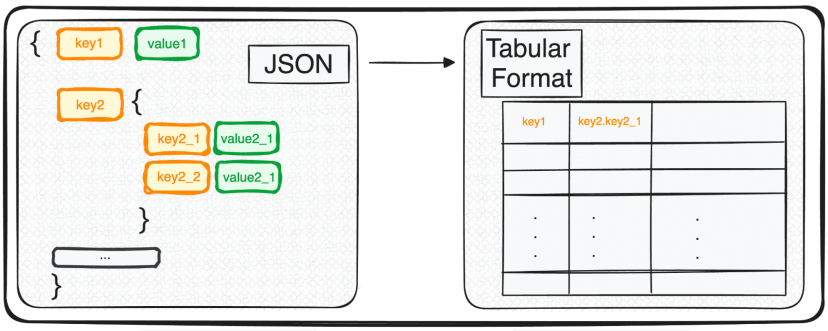
Berdasarkan Gambar 3.1, sistem perangkat yang akan dibangun terintegrasi dengan sensor tanaman, yaitu *Flower Care Sensor*. Sensor ini akan mendeteksi nutrisi tanaman melalu *Electrical Conductivity* (EC) pada tanah, suhu dan kelembaban lingkungan, temperatur udara, serta kelembaban tanah. Informasi-informasi yang didapat oleh sensor akan dikirim ke sebuah mikrokontroler, yaitu Raspberry Pi 4 Model B. Mikrokontroler ini akan bekerja sebagai “otak” dari keseluruhan sistem yang dibangun. Tujuannya adalah sebagai pengelola, pengolah serta pusat transaksi data yang menerima informasi dari sensor dan mengirimnya ke *Firebase Realtime Database* di *Cloud*.

Sistem yang terintegrasi dengan Raspberry Pi 4 Model B akan memungkinkan pengelolaan data sensor dengan lebih efisien dan responsif. Informasi dari Raspberry Pi 4 Model B, dikirimkan ke *Firebase Realtime Database* melalui koneksi WiFi. Informasi yang tersimpan adalah EC, suhu, intensitas cahaya, dan kelembaban lingkungan. *Firebase Realtime Database* merupakan penyimpanan *Cloud* yang bersifat *NoSQL* yang dimiliki oleh *Google Firebase*. Sesuai namanya, *Firebase Realtime Database* memberikan *real-time database communcation* dan aksesibilitas yang fleksibel dalam transaksi data. Sifatnya yang *real-time* dan *open source* memudahkan sistem ini untuk dibangun secara cepat dan efisien. Selanjutnya, setelah disimpan di *Cloud*, data ini akan digunakan untuk melatih *Random Forest* model, yang pada akhirnya model ini akan di-*hosting* pada *Web Server* untuk diakses sebagai API (*Application Programming Interface*), yang kemudian dapat diakses oleh *multi-platform mobile app* yang menampilkan visualisasi data tersebut. Proses transaksi data lebih teknisnya dijelaskan pada Gambar 3.2 berikut.



Gambar 3. 2 Proses Transaksi Data

Seperti yang digambarkan dalam Gambar 3.1 dan Gambar 3.2 sebelumnya mengenai arsitektur dari sistem yang akan dibangun, seluruh data yang terdapat di *Firebase Realtime Database* digunakan sebagai dataset. Namun, dari total 2696 dataset yang ada, hanya sebagian saja yang digunakan. Proses *dataset filtering* dan *dataset selection* inilah yang menyeleksi dataset sample yang akan digunakan untuk melatih model prediksi. Dataset yang digunakan sebagai input model adalah 2000 data pertama yang diambil pada saat pagi hari sampai sore hari, ditandai dengan intensitas cahaya 600 lux atau lebih. Setelah filtrasi dan seleksi didapatkan total sampel adalah 1099 sampel. Sebelum melakukan pelatihan model (*model training*), perlu diingat bahwa dataset yang diterima dari *Firebase* adalah JSON (*Javascript Object Notation*) atau (Notasi Obyek Javascript) sehingga harus diubah ke dalam bentuk *Python Dataframe* terlebih dahulu.

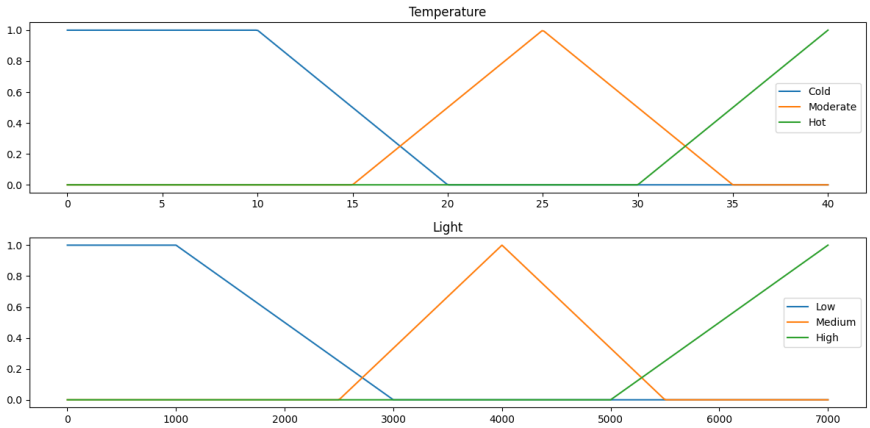


Gambar 3. 3 Ilustrasi Konversi JSON dan Dataframe

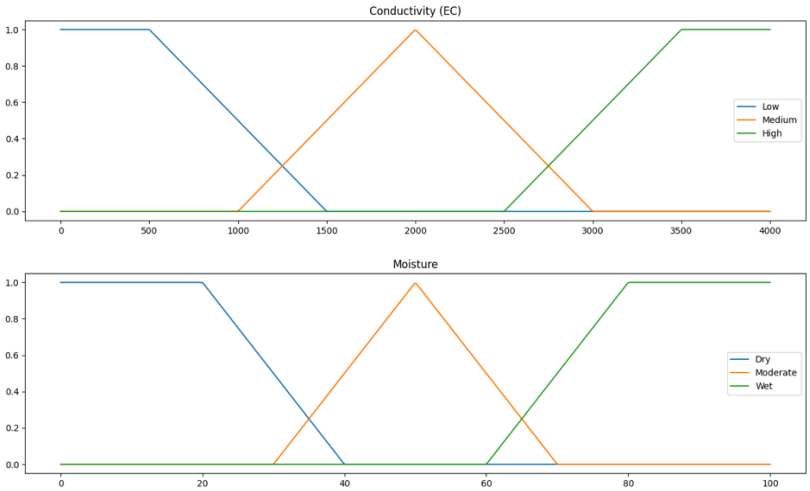
Sumber: https://www.kdnuggets.com/converting-jsons-to-pandas-dataframes-parsing-them-the-right-way

Proses ekstraksi fitur dan konversi JSON ke dalam bentuk Dataframe ini disebut *Mapping*. Alasan dibutuhkannya proses *mapping* adalah sifat kedua struktur data yang berbeda. JSON adalah struktur data yang didesain untuk transaksi data dengan detail, namun sangat ringan karena hanya menggunakan format teks dengan struktur *Key*/*Field* dan *Value* [14]. Berbeda dengan JSON, *Dataframe* adalah struktur data dari librari Python, yaitu Pandas, yang fungsinya untuk menyimpan data jumlah banyak dalam bentuk tabular (disusun menjadi *table*) [15]. Setelah *mapping* dan konversi JSON ke Python Dataframe, maka *data preprocessing* dilanjutkan dengan *target labeling* dan *Fuzzy determining*, ini adalah proses menambahkan *field* untuk target pengujian ke dalam Dataframe, lalu melakukan *Fuzzification* sesuai dengan *Fuzzy membership* yang sudah dibuat. Untuk melakukan *Fuzzification*, setidaknya dibutuhkan beberapa *Fuzzy membership* sebanyak parameter yang digunakan untuk menghasilkan *Fuzzy rule*. *Parameter* yang digunakan dalam penelitian ini untuk membuat *Fuzzy membership function* adalah temperatur, konduktivitas listrik di tanah (kesuburan), intensitas cahaya, dan kelembapan udara.

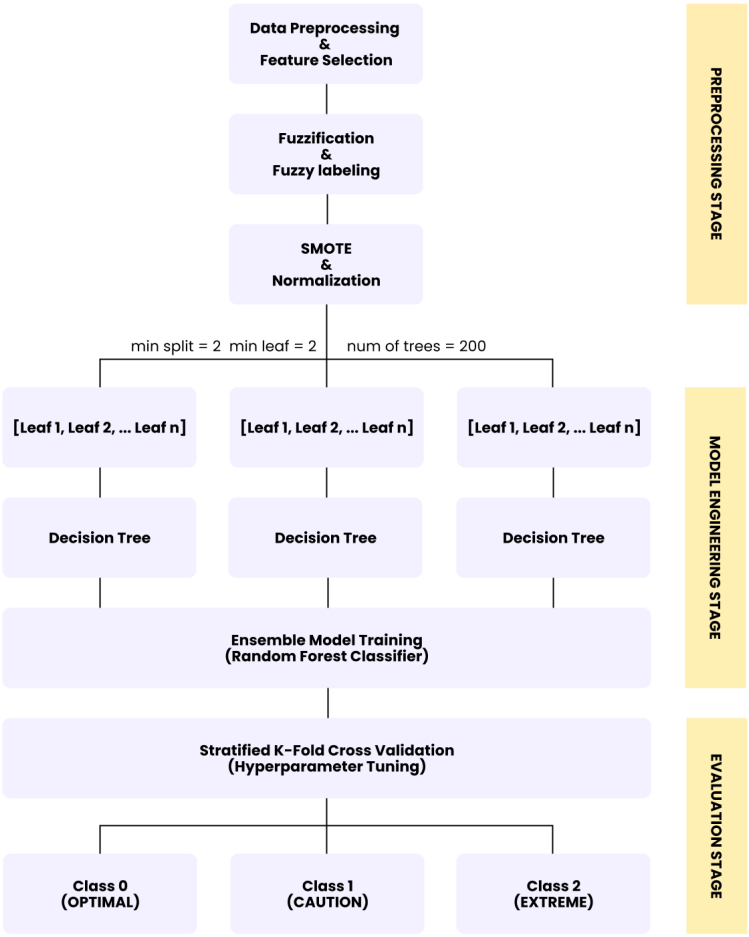
Adapun *Fuzzy rule* yang dari setiap parameter dapat dilihat pada Gambar 3.3 untuk parameter lingkungan dan Gambar 3.4 untuk parameter tanah.



Gambar 3. 4 Environment Parameter Fuzzification



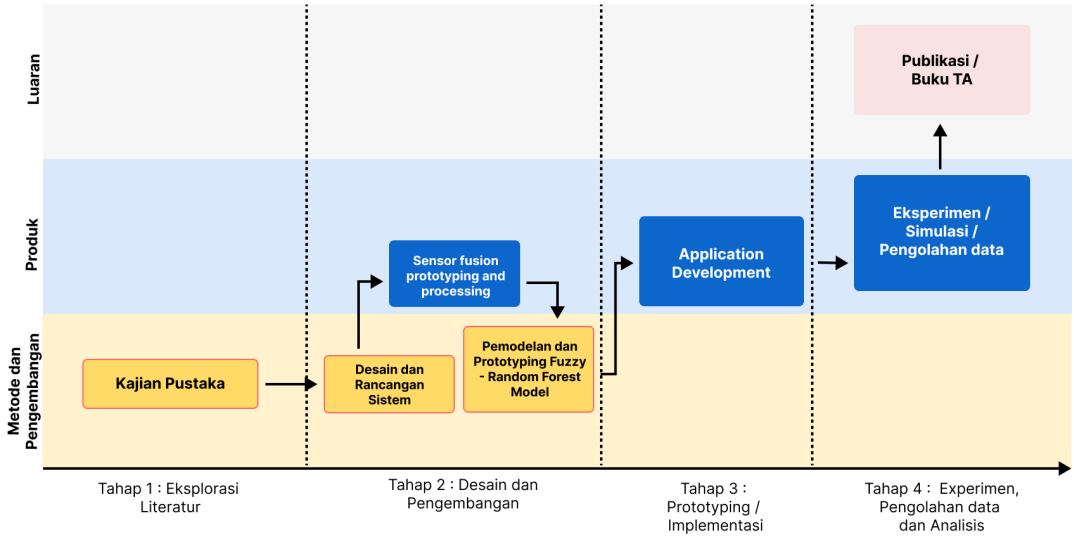
Gambar 3. 5 Soil Parameter Fuzzification



Gambar 3. 6 *Fuzzy-Random Forest Block*

Gambar 3.6 merupakan *Fuzzy-Random Forest block diagram*, menggambarkan proses algoritma *Fuzzy Logic* di-*ensemble* dengan *Random Forest* untuk melakukan klasifikasi kondisi tanaman menjadi normal, ekstrim, atau kurang nutrisi. Tahapan ini memungkinkan identifikasi pola kompleks dalam data sensor dan memberikan wawasan mendalam mengenai kondisi tanaman. Kondisi tanaman inilah yang nantinya akan ditampilkan dalam aplikasi monitoring. Aplikasi yang akan dikembangkan merupakan *multi-platform mobile app*, yaitu aplikasi yang dapat diakses melalui tablet maupun ponsel. Aplikasi ini akan memberikan visualisasi data dari keseluruhan informasi yang sebelumnya diproses oleh model dari dataset yang ada di *Firebase Realtime Database*.

3.2 Tahapan Pengerjaan TA



Gambar 3. 7 Block Diagram Tahapan Pengerjaan TA

Berdasarkan Gambar 3.7 di atas dilakukan beberapa tahapan dalam pengerjaan TA penelitian ini sebagai berikut.

Tahap 1: Kajian Pustaka

Tahap ini bertujuan dalam memahami dan menjelaskan secara mendalam konsep-konsep, definisi serta teori mengenai parameter penelitian yang akan dilakukan. Memahami perangkat yang akan digunakan seperti *Mi Flower Care Sensor* dan mikrokontroler Raspberry Pi 4 Model B, mencari konsep dan teori mengenai deteksi nutrisi menggunakan Electrical Conductivity (EC) dan pH pada tanah, serta kelembaban dan suhu udara. Selain itu, pengumpulan kajian pustaka juga bertujuan untuk memahami dampak serta pengaruh parameter pengukuran terhadap tanaman yang akan dipantau.

Tahap 2: Desain dan Rancangan Sistem

Pada tahap ini, hasil kajian pustaka digunakan sebagai dasar utama dalam membangun sistem pengukuran dan sensor fusion penelitian ini. Kajian pustaka yang sebelumnya dikumpulkan juga digunakan sebagai fondasi dalam membuat sistem analisis yang lebih baik lagi dalam membangun sistem pemantauan dan identifikasi kondisi lingkungan. Tahapan ini juga dijadikan sebagai rancangan pemodelan arsitektur sistem yang akan dibangun, rekayasa teknologi keseluruhan yang akan digunakan pada saat *prototyping* maupun implementasi di lapangan.

Tahap 3: *Sensor Fusion Prototyping* dan *Processing*

Pada tahapan ini, rancangan sistem diimplementasikan untuk membangun prototipe *sensor fusion* dari Mi Flower Care dan Raspberry Pi Zero W. Alat prototipe ini akan digunakan untuk membaca EC dan pH pada tanah, serta suhu dan kelembaban pada lingkungan tanaman. Mikrokontroler Raspberry Pi Zero W akan menjadi "otak" dari sistem perangkat ini, di mana akan menerima informasi dari sensor yang terhubung menggunakan *Bluetooth Low Energy* dan mengirimkan informasi tersebut ke *Firebase Realtime Databse* (*Cloud*) menggunakan koneksi internet.

Tahap 4: Pemodelan dan *Prototyping Fuzzy–Random Forest Model*

Tahapan ini adalah tahapan paling penting kedua setelah prototipe dalam menyusun penelitian ini. Pada tahap ini akan dilakukan pemodelan dan penyusunan *Fuzzy – Random Forrest machine learning model*. Model akan menggunakan *real-time* dataset yang tersimpan di Firebase, selanjutnya model ini akan mengklasifikasikan dataset tersebut ke dalam 3 kategori, yaitu Kondisi *Extreme*, Kondisi *Optimal* dan Kondisi *Caution* (*Undernutrition*/*Overnutrition*).

Tahap 5: Application Development

Selanjutnya pada tahapan ini dilakukan pembuatan *multi-platform dashboard*. Aplikasi ini akan dibangun menggunakan *Streamlit Python* untuk memberikan *multi-platform mobile app* yang dapat dibuka dengan ponsel pengguna, baik yang memiliki platform OS Android ataupun iOS. Tahapan ini akan mengimplementasikan model *Fuzzy–Random Forest* sebelumnya yang sudah dibuat ke dalam bentuk visualisasi data pemantauan parameter tanaman.

Tahap 6: Experimen dan Pengolahan Data

Pengujian yang mengukur efektifitas serta efisensi model dan sistem yang dibangun. Pada tahap ini juga pengolahan data diimplementasikan secara real-time dan dievaluasi secara berkala untuk menghasilkan model yang cepat, akurat, efektif dan efisien dalam membaca perubahan parameter. Selain itu, pada tahap ini juga dilakukan reevaluasi model secara berkala untuk memastikan model memberikan hasil yang presisi pada saat perubahan parameter terjadi secara tiba-tiba dan sangat signifikan. Proses prediksi akan dipastikan berjalan dengan optimal sebelum melanjutkan ke publikasi buku Tugas Akhir (TA).

Tahap 7: Publikasi Buku TA

Proses publikasi dan penyusunan buku Tugas Akhir (TA) yang merupakan langkah akhir dalam kontribusi penelitian ini dan pemahaman ilmiah ini. Hasil eksperimen, penelitian serta simulasi dan analisis data yang sudah diolah akan dipublikasikan secara ilmiah, secara teliti dan merinci terhadap konteks materi yang sudah dipelajari. Publikasi ini juga bertujuan untuk memberikan dan membagikan pengetahuan baru, memperbaiki serta mengonfirmasi temuan dari literatur sebelumnya yang menjadi fondasi penelitian dan bidang ini.

3.3 Rancangan Algoritma

Implementasi algoritma yang digunakan pada penelitian ini adalah *Fuzzy Logic* yang didukung oleh salah satu metode pembelajaran mesin berupa *ensemble learning*, yaitu *Random Forest*. Tujuan dari menggabungkan *Fuzzy Logic* dengan *Random Forest* ini adalah mendapatkan klasifikasi yang lebih akurat. Dalam menggunakan logika *Fuzzy*, digunakan 4 parameter, yaitu *Electrical Conductivity* (EC), *temperature*, intensitas cahaya, dan kelembaban tanah yang berpengaruh dalam memberikan label klasifikasi. Secara garis besar, berikut merupakan *pseudocode* atau rancangan algoritma dari Fuzzy-Random Forest (FRF) yang digunakan.

```
FUZZY_LABELING(in: Map<String, Any> value; out: Integer rule) -> Integer
begin
    DECLARE temp, light, ec, moisture, rule

    temp : value["temperature"]
    light : value["light"]
    ec : value["conductivity"]
    moisture : value["mositure"]


    IF  temp is less than 10 OR ec is less than 10 OR moisture is less than 30 OR light is less than 100
        rule <-- 2

    IF  temp is between 22 and 27 INCLUSIVE AND light is between 3500 and 5600 INCLUSIVE AND
        ec is between 1500 to 2500 INCLUSIVE AND moisture is between 35 and 50 INCLUSIVE
        rule <-- 0

    IF  temp is between 20 and 22 EXCLUSIVE OR temp is between 27 and 30 EXCLUSIVE OR
        light is between 1500 and 3500 EXCLUSIVE OR light is between 5600 and 6000 INCLUSIVE OR
        ec is between 950 and 1500 EXCLUSIVE OR ec is between 2500 and 3000 INCLUSIVE OR
        moisture is between 30 and 35 INCLUSIVE OR moisture is between 50 and 60 INCLUSIVE
        rule <-- 1

    RETURN rule
end
```

Gambar 3. 8 Algoritma *Basic Fuzzy Labeling*

Baris kode di Gambar 3.8 merupakan fondasi utama logika *Fuzzification yang* digunakan dalam penelitian ini. Logika ini menentukan label yang diberikan kepada sampel dataset sebelum digunakan oleh model saat melakukan pelatihan. Logika ini memiliki tiga *rules* yang didapatkan dari proses *Fuzzification*. Penelitian ini terdapat 3 klasifikasi, yaitu ektrim, optimal dan *caution* (*undernutrition* / *overnutrition*). Proses *Fuzzification* inilah yang digunakan untuk menentukan batas-batas dari tiap-tiap parameter yang ada, membagi ketiga *rules* dari dengan batas minimum dan maksimumnya masing-masing. Penerapan Fuzzy ini dibagi menjadi 3 proses, yaitu:

1. ***Triangular Function***

Fungsi ini dibangun untuk mengukur sejauh mana nilai parameter termasuk dalam rentang tertentu dengan pendekatan berbentuk segitiga. *Membership* ini menghasilkan nilai 0 sampai 1 yang menunjukkan derajat keanggotaan suatu nilai dalam rentang yang telah ditentukan. Selain itu, fungsi ini memudahkan penilaian parameter dengan memberikan skor yang merepresentasikan seberapa sesuai dengan rentang optimal atau *caution*.

2. **Penilaian *Optimal* dan *Caution Score***

Pada penilaian *optimal*, algoritma menghitung skor dalam rentang optimal. Misalnya, suhu di antara 20 hingga 30 derajat dianggap dalam kondisi optimal. Nilai keanggotaan akan mendekati 1 jika parameter berada di tengah rentang optimal, dan mendekati 0 jika berada di luar rentang tersebut.

3. **Penentuan *Output Label***

Jika semua parameter menunjukkan *membership* dalam rentang optimal, maka label ditetapkan sebagai 0. Apabila ada yang tidak dalam rentang optimal, namun ada yang

berada dalam rentang caution, maka label ditetapkan sebagai 1. Jika salah satu parameter berada di bawah batas kritis. Kondisi dianggap ekstrem, label ditetapkan sebagai 2.

Selanjutnya untuk membuat prediksi yang lebih baik lagi dalam menentukan *decision* untuk klasifikasi, digunakanlah model *Random Forest*. *Random Forest* atau RF merupakan metode *Ensemble Learning* yang dapat mengurangi *variance error* dan menghasilkan *classifier model* yang lebih efisien dan akurat. Dengan bantuan Fuzzy rule sederhana yang telah dibangun, model RF dapat dilatih sesuai *condition labeling* yang dilakukan pada saat *dataset preprocessing*.

```
FUNCTION RFModel(in: Integer n_estimators, Integer max_depth, Matrix X_train, Vector y_train; out: RF_Model model) ->
RandomForestClassifier
begin
    DECLARE n_decision_tree: List of Decision Trees, RF_Model

    RF_Model_Estimators <-- n_estimators
    RF_Model_Depth <-- max_depth

    FOR i FROM 1 TO n_estimators DO
        DECLARE X_bootstrap, y_bootstrap

        X_bootstrap, y_bootstrap <-- BootstrapSample(X_train, y_train)

        decision_tree <-- DecisionTree(max_depth)
        decision_tree.train(X_bootstrap, y_bootstrap)

        n_decision_tree.Add(decision_tree)
    END FOR

    RF_Model.decision_trees <-- n_decision_tree
    RETURN RF_Model
end

Function DecisionTree(in: Integer max_depth; out: DecisionTree DT) -> DT
begin
    DECLARE decision_tree: DT

    decision_tree <-- Tree
    decision_tree.depth <-- max_depth

    RETURN decision_tree
end

Function BootstrapSample(in: Matrix X, Vector y; out: (Matrix, Vector)) -> (Matrix, Vector)
begin
    DECLARE X_bootstrap: Matrix, y_bootstrap: Vector

    FOR i FROM 1 TO LENGTH(X) DO
        index <-- RandomInteger(1, LENGTH(X))

        X_bootstrap[i] = X[index]
        y_bootstrap[i] = y[index]
    END FOR

    RETURN (X_bootstrap, y_bootstrap)
end
```

Gambar 3. 9 Algoritma Random Forest

Baris kode di Gambar 3.9 di atas adalah *Random Forest* melakukan prediksi kondisi lingkungan tanaman. Sesuai namanya, *Forest* atau hutan memiliki banyak pohon di dalamnya. Model *Random Forest* menggunakan kumpulan pohon yang disebut *Decision Tree*, pohon-pohon ini yang secara individual menilai dan menentukan klasifikasi sebuah kondisi berdasarkan input yang diberikan. Berikut merupakan penjelasan fungsi-fungsi yang ada dalam Gambar 3.9.

1. **Fungsi *DecisionTree***

Pohon yang membuat keputusan dan melakukan tugas klasifikasi. Dalam model *Random Forest*, jumlah pohon ini yang nantinya menentukan performa model dalam menentukan klasifikasi lingkungan tanaman. Jumlah pohon yang terlalu banyak dapat menghasilkan model RF yang terlalu kompleks dalam menentukan komputasi. Jumlah model yang terlalu sedikit juga perlu dipertimbangkan karena dapat menghasilkan model yang *underfitting*, yaitu tidak mampu menangkap kompleksitas data, sehingga memberikan hasil klasifikasi yang buruk.

2. **Fungsi *RandomForestClassifier***

Merupakan *machine learning model* utama yang digunakan dalam penelitian ini. Model *RandomForestClassifier* ini adalah variasi *Random Forest* yang ditujukan untuk prediksi masalah klasifikasi, sedangkan variasi lainnya adalah prediksi regresi (matematis). Kedua variasi model ini bergantung pada jumlah *Decision Tree* yang dimilikinya, menentukan performa model tersebut optimal ataupun tidak.

3. **Fungsi *BootstrapSample***

Metode yang diberikan kepada model RF untuk membentuk *predictive model* yang optimal. *Bootstrap* ini adalah metode statistik yang diberikan untuk memperkirakan distribusi sampel dengan mengambil sampel berulang kali dari data asli. Tujuan dari implementasi *bootstrap* adalah meningkatkan kekuatan generalisasi suatu model, karena dengan ini setiap pohon dilatih pada *bootstrap sample* yang berbeda, sehingga setiap pohon memiliki pandangan yang sedikit berbeda terhadap data pelatihan.

3.4 Rencana dan Skenario Eksperimen

Table 3. 1 Skenario Eksperimen

No	Tujuan Eksperimen / Simulasi	Variabel atau Konstanta	Parameter yang diukur
1	Memahami dan meneliti pengaruh EC, suhu lingkungan, dan intensitas cahaya terhadap tanaman	Electrical Conductivity (EC), suhu, intensitas cahaya	Pengaruh parameter terhadap kondisi optimal, *caution* serta *extreme* oleh FRF Model
2	Proses m*odel preprocessing*, evaluasi dan analisis pengaruh SMOTE *imbalanced handling* yang diimplementasi ke FRF model	*Train & test dataset,* SMOTE *imbalanced handling*, FRF model	Performa, akurasi, presisi, error FRF model baik sebelum maupun sesudah *imbalanced handling*
3	Evaluasi dan analisis *learning curve* yang didapat dari model selama *training*	FRF Model sebelum dan sesudah *hyperparameter-tuning*	Mengenali performa dari FRF model serta mendeteksi *overfitting* dan *underfitting*
4	Evaluasi waktu respon dalam membaca perubahan parameter yang kompleks	FRF Model yang sudah di-*hosting* sebagai API	Performa prediksi dalam membaca variasi parameter lingkungan tanaman
5	Evaluasi Performa Aplikasi *Monitoring* dalam menangkap respon API	*Multi-Platform Mobile App*	Performa *Multi-Mobile App* dalam menangkap respon API dan menampilkannya

BAB IV
HASIL DAN ANALISIS

4.1 Dataset Pengujian

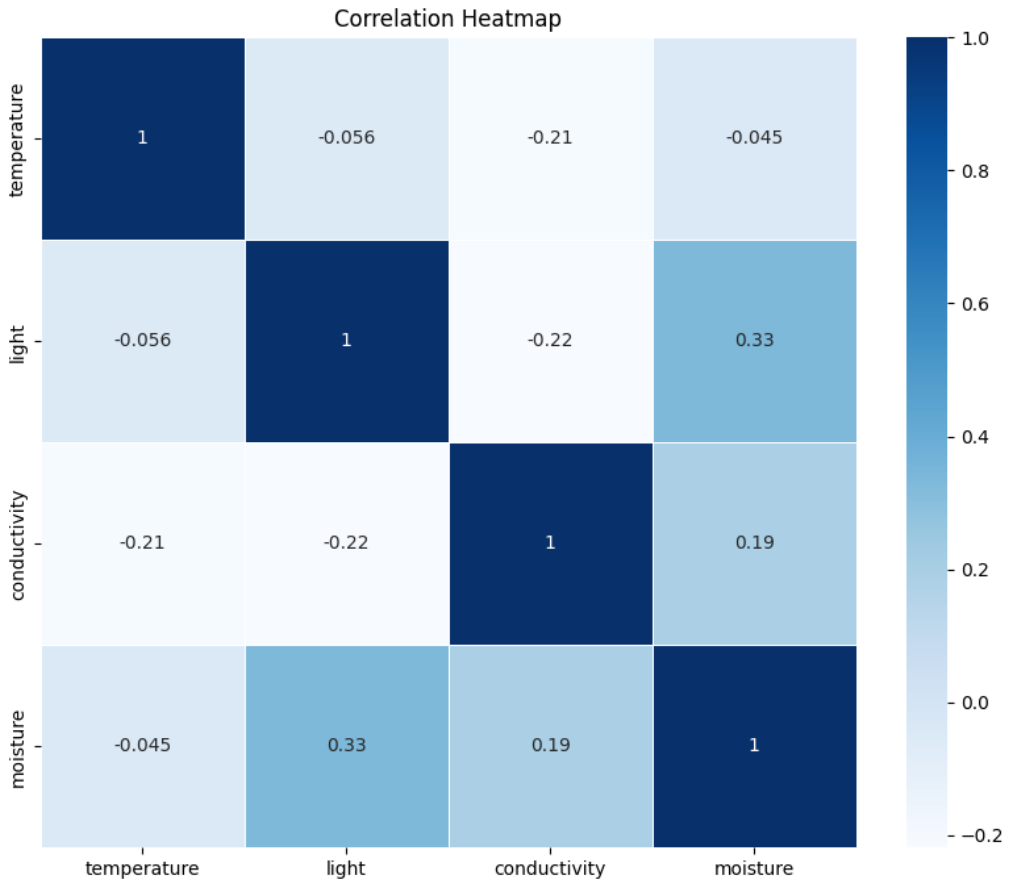
Dataset pengujian yang digunakan adalah dataset sampel yang didapatkan dari hasil *data preprocessing*, tepatnya pada proses *dataset splitting*. Dari 1099 sampel yang terseleksi, sebanyak 35% porsi digunakan untuk dataset pengujian (*test dataset*). Dataset pengujian ini juga melalui proses yang sama pada melakukan validasi dan *hyperparameter tuning* saat melakukan pelatihan. Dataset pengujian ini berisi tiga jenis data yang sudah teracak, di dalamnya ada data yang mewakili klasifikasi *Optimal*, *Caution*, dan *Extreme*.

Data dengan kondisi *Optimal* adalah data yang memiliki parameter sesuai batas *optimum level* dari *Fuzzification* dan tidak melebihi atau kurang dari batas *moderate level*. Data-data *optimal* didapat dari hasil eksperimen di mana tanaman Tomat Ceri dimonitor dan dikontrol keadaannya selama 8 jam dalam jangka waktu dua minggu setiap pagi sampai sore hari. Selanjutnya, kondisi *Caution* didapatkan pada waktu yang sama saat proses pengujian kondisi *Optimal* berlangsung. Data-data dengan kondisi *Caution* diperoleh pada saat dua atau lebih parameter pengujian, yaitu naik ataupun turun melebihi garis *optimal*. Hal ini sering terjadi pada saat perubahan cuaca dan pada saat matahari tepat berada di atas kepala, tepatnya pukul 12 sampai 1 siang. Selain itu, ada data-data dengan kondisi *Extreme* yang artinya data tersebut memiliki parameter yang jauh lebih tinggi atau lebih rendah dari batas *optimal*-nya. Data-data *Extreme* didapatkan dengan cara *artificial* atau dengan campur tangan, seperti mengurangi sinar matahari yang mengarah ke tanaman dan mengurangi atau melebihi jumlah nutrisi serta kadar air untuk mengubah kondisi lembab atau kering.

Berdasarkan cara mendapatkan data, pengujian untuk mendapatkan data *Extreme* dilakukan terakhir setelah kondisi *Optimal* dan *Extreme* sudah terpenuhi karena sifatnya yang *destructive* dan sangat berpengaruh terhadap perkembangan tanaman. Total dataset yang didapatkan selama satu bulan penuh adalah 2696 data, digunakan sebanyak 1099 sampel setelah diseleksi, lalu dilakukan *dataset splitting* dengan porsi 65% pelatihan dan 35% pengujian. Dataset pengujian ini memuat 863 *Caution*, 132 *Extreme*, dan 104 *Optimal*.

4.2 Korelasi antara Suhu, Intensitas Cahaya, Kelembaban Tanah, dan *Electrical Conductivity*

Pengujian pertama merupakan *data exploration* dan *data preprocessing*, di mana korelasi hubungan keempat parameter lingkungan dianalisis. Variabel yang digunakan dalam eksplorasi data ini adalah temperatur, intensitas cahaya, kelembaban tanah, dan konsentrasi EC. Parameter yang diukur difokuskan kepada korelasi antara temperatur, intensitas cahaya, kelembaban tanah, dan konsentrasi EC tersebut. Dengan adanya eksplorasi korelasi parameter ini, maka dapat diperoleh pemahaman yang lebih mendalam tentang perubahan nilai dan konsentrasi dari suhu, cahaya, kelembaban tanah dan EC dengan pengaruhnya terhadap kondisi lingkungan tanam secara keseluruhan.



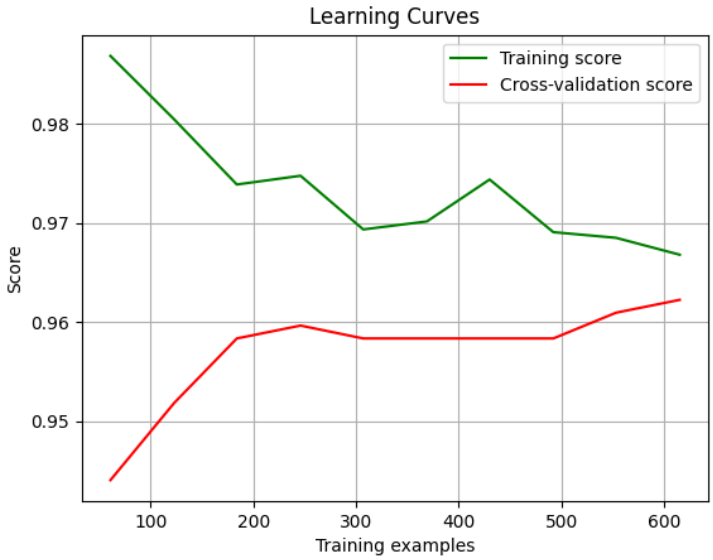
Gambar 4. 1 *Heatmap* Korelasi Parameter

Berdasarkan Gambar 4.1, dapat dilihat bahwa sebagian besar parameter yang dianalisis tidak menunjukkan korelasi yang signifikan satu sama lain. Terdapat korelasi positif lemah antara variabel light dan moisture, yang mengindikasikan adanya kecenderungan peningkatan kadar kelembaban seiring dengan peningkatan intensitas cahaya. Namun, hubungan ini tidak terlalu kuat. Selain itu, ditemukan beberapa korelasi

negatif lemah, seperti antara temperature dan moisture. Hal ini menunjukkan bahwa variabel-variabel yang dianalisis cenderung independen satu sama lain. Korelasi ini digunakan untuk menguji hipotesis serta teori mengenai hubungan antara variabel-variabel tertentu, sebelum kemudian menentukan proses *modelling*.

4.3 Evaluasi Pengaruh *Imbalanced Handling* pada saat *Data Preprocessing*

Dataset yang digunakan didapatkan pada saat bulan cuaca pancaroba, di mana kemungkinan untuk variasi data yang diperoleh sangat besar dan perubahan signifikan juga pasti terjadi. Maka dari itu, pada dataset ini, jumlah data dengan label *Caution* lebih banyak dibandingkan data *Optimal* dan *Extreme*, menciptakan dataset yang jumlah sebaran datanya tidak seimbang. Sebaran data yang tidak seimbang ini dapat menyebabkan model overfit terhadap salah satu atau beberapa klasifikasi saja. Selain itu, perlu diingat bahwa *Random Forest* menggunakan metode *majority vote* atau voting terbanyak ialah yang dipilih sebagai hasil akhir. Akibat *imbalanced dataset* terhadap *majority vote* ini adalah bias terhadap kelas terbanyak dalam dataset, artinya, dalam penelitian ini model cenderung lebih mengenal kondisi 0 (*Optimal*) karena sebarannya paling sedikit dan 1 (*Caution*) karena sebarannya paling banyak. Gambar 4.2 berikut merupakan grafik *learning curve* dari pelatihan model tanpa *imbalanced handling*, diikuti oleh Tabel 4.1 yang merupakan *performance metrics*.



Gambar 4. 2 *Learning Curve* sebelum *SMOTE*

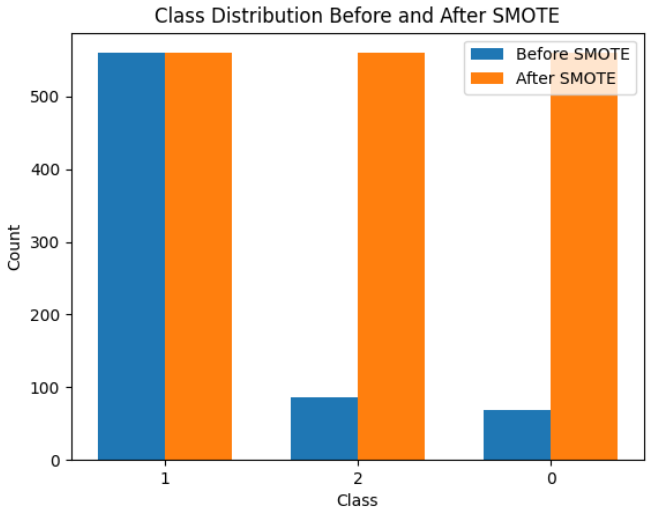
Table 4. 1 *Performance Metrics* sebelum SMOTE

Class	Precision	Recall	F1-Score	Support
0	1.0	1.0	1.0	36
1	1.0	1.0	0.99	302
2	0.7	0.7	0.8	47
Accuracy				0.95

Berdasarkan grafik *learning curve* Gambar 4.2, dapat dilihat bahwa garis hijau adalah *training score* dan garis merah adalah *validation score* sesuai legenda. Grafik ini mengindikasikan *overfitting*, yaitu kondisi di mana model terlalu terpaku dan menghafal *training dataset*, yang seharusnya model ini mempelajari hubungan yang mendasari dan pola yang dapat digeneralisasi ke data lain yang belum dilihat. Model yang *overfitting* seperti ini mungkin dapat mencapai akurasi tinggi pada saat *training*, tetapi akan gagal dalam memprediksi data baru yang tidak pernah muncul di *training dataset*. Berdasarkan grafik, indikasi *overfitting* ini dapat dikenali karena *training score* dimulai sangat tinggi mendekati skor 1.0, kemudian turun dengan stabil ke sekitar 0.972, kemudian berfluktuasi sangat bebas dan menutup di sekitar 0.966. Melihat *validation score*, dimulai dari sangat rendah yang kemudian naik ke sekitar 0.959 namun mengalami penurunan kembali walaupun kecil dan stabil. Selain itu, dapat kita lihat juga, gap dari kedua garis sangat besar yang menandakan gejala *overfitting* yang sangat besar.

Grafik tersebut didukung dengan Tabel 4.1 yang menggambarkan performa dari pelatihan model, di mana nilai *precision*, *recall* dan *F1-score* di kelas 0 dan 1 adalah sempurna. Namun, nilai sempurna ini sangat berbeda dengan nilai *recall* atau kemampuan model mengingat kembali pada kelas 2, serta nilai *precision* atau ketepatannya yang adalah 0.7. Hal ini menunjukkan bahwa model mempelajari dataset latihnya namun hanya menghapal dataset tersebut, sehingga tidak dapat mempelajari data baru yang diberikan untuk diprediksi. Maka dari itu, untuk menghindari dataset latih yang tidak seimbang, dilakukanlah *imbalanced handling* atau penanganan data yang tidak seimbang menggunakan metode SMOTE.

Metode *Synthetic Minority Oversampling Technique* (SMOTE) adalah metode *oversampling* untuk menyeimbangkan distribusi kelas dalam kumpulan data. Metode ini memilih minoritas terdekat dari *feature space* [16]. Secara sederhana, metode ini mencetak sampel baru dari kelas minoritas untuk menyeimbangkan dataset. Implementasi SMOTE menghasilkan dataset yang jauh lebih seimbang dan relevan, seperti yang divisualisasikan pada Gambar 4.3 berikut.



Gambar 4. 3 Grafik sebelum dan sesudah *SMOTE*

Gambar 4.3 menunjukkan bahwa SMOTE menyeimbangkan kelas minoritas atau kelas yang sedikit jumlah sebarannya dengan cara *oversampling*. Seperti yang ada pada Gambar 4.3, batang berwarna biru adalah sebaran kelas sebelum SMOTE dan batang berwarna oranye adalah sebaran kelas sesudah SMOTE. Setelah ketiga kelas ini dibuat seimbang, dataset dapat kemudian digunakan sebagai *training dataset*. Namun, untuk mencegah *overfitting* terjadi kembali, sebelum digunakan sebagai *training dataset*, dilakukanlah normalisasi. Dalam penelitian ini, digunakanlah Python StandardScaler, yaitu metode statistika untuk mengubah ukuran distribusi nilai sehingga rata-rata nilai yang diamati adalah 0 dan deviasi standarnya adalah 1 [17]. Setelah dataset sudah *balanced* dan *normalized*, maka dilanjutkan untuk menjadi *training dataset* untuk melatih model yang menghasilkan *learning curve* seperti pada Gambar 4.4.

Gambar 4. 4 *Learning Curve* setelah *SMOTE* dan *Scaling*

Berdasarkan Gambar 4.4, *training score* (garis hijau) menunjukkan tren peningkatan yang hampir stabil, mengindikasikan bahwa model mulai mempelajari pola-pola *training dataset*. Selain itu, mendekati *training example* 1400, *gap* atau jarak antara *training* dengan *cross-validation score* mulai berdekatan, namun belum dapat dikatakan bahwa model mampu mengeneralisasikan data. Grafik *validation score* mengalami fluktuasi, namun mengarah mendekati *training score* dan hampir bertemu. Namun, perlu dicatat bahwa pada *training example* <1000, *gap* antara *training* dengan *cross-validation score* sangat jauh yang artinya masih mengalami *overfitting*, namun grafiknya sudah lebih baik daripada sebelumnya. Walaupun sudah performa pelatihan saat ini sudah mengalami peningkatan, model belum bisa dikatakan optimal dan akurat. Performa tersebut dapat dilihat dari Tabel 4.2 berikut, di mana metriks yang didapatkan sangat tinggi, bahkan akurasinya hampir mencapai nilai 1.0. Apabila didukung dengan Gambar 4.4, maka dapat disimpulkan model mengalami *overfitting* hebat.

Table 4. 2 *Performance Metrics* sesudah SMOTE dan Normalisasi

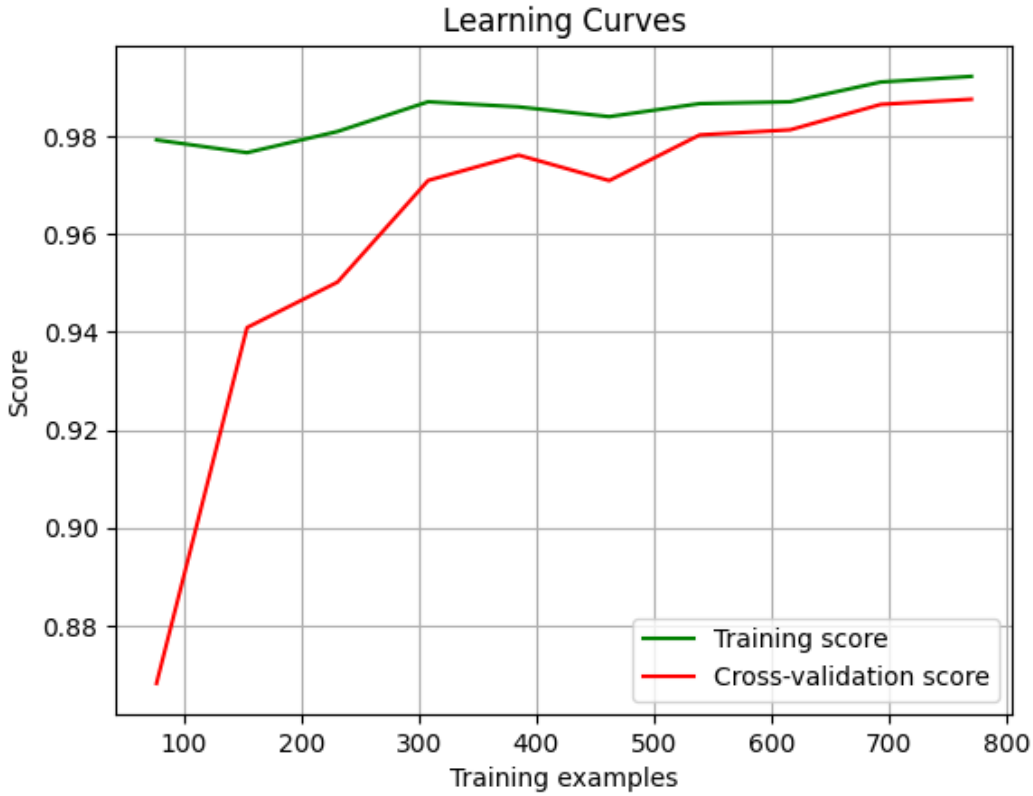
Class	Precision	Recall	F1-Score	Support
0	1.0	1.0	1.0	36
1	0.98	1.0	1.0	302
2	0.98	0.97	0.95	47
Accuracy				0.99

4.4 Evaluasi Pengaruh *Hyperparameter Tuning* menggunakan *Stratified K-Fold Cross Validator*

Setelah mengamati hasil eksperimen dengan mengimplementasikan *imbalanced handling* dan normalisasi, diketahui bahwa *learning curve* mengindikasikan model masih mengalami *overfitting*, namun sudah ada perkembangan dari pengujian sebelum *balancing*. Hal ini tentunya menjelaskan bahwa model belum mengeneralisasikan data dengan baik dan masih menghafal *training dataset*. Maka itu, pengujian selanjutnya adalah mengimplementasikan *Hyperparameter Tuning*, yaitu proses mencari nilai optimal dari *hyperparameter* suatu model untuk memperbaiki performa model *machine learning* [18]. Proses ini adalah bagian penting dari proses *model engineering* suatu algoritma *machine learning*.

Pada penelitian ini, proses *hyperparameter tuning* dilakukan untuk meningkatkan performa *Random Forest* yang sebelumnya sudah dilakukan *imbalanced handling* dan normalisasi. Model RF yang saat ini diperoleh masih bersifat fundamental, sehingga masih "model mentah" yang belum diperbaiki *hyperparameter*-nya. Maka dari itu, pada proses inilah dilakukan integrasi *Stratified K-Fold Cross Validation* (SKCV), sebuah metode *K-Fold Cross Validation* yang menjaga keseimbangan distribusi dan sebaran kelas di setiap *fold*, terutama untuk dataset yang tidak seimbang [19]. Sedangkan *K-Fold Cross Validation* sendiri merupakan metode untuk menghindari overfitting serta memastikan model mampu mengeneralisasikan data dengan baik dengan cara membagi data latih menjadi beberapa *K-subset* yang disebut *K-Fold* dengan ukuran yang sama [20].

Pada penelitian ini, SKCV digunakan untuk membagi data latih menjadi 5 *K-Fold* yang menghasilkan grafik *learning curve* seperti pada Gambar 4.5. Grafik *learning curve* menunjukkan peningkatan signifikan ke arah yang lebih baik setelah melakukan *hyperparameter tuning*. Peningkatan performa dapat dilihat dengan cara membandingkan *training score* saat ini dengan *cross-validation score*-nya seperti kedua grafik *learning curve* sebelumnya.



Gambar 4. 5 *Learning Curve* setelah *Hyperparameter Tuning*

Table 4. 3 *Performance Metrics* setelah *Hyperparameter Tuning*

Class	Precision	Recall	F1-Score	Support
0	0.97	1.0	0.98	37
1	0.98	0.98	0.99	302
2	0.93	0.97	0.95	46
Accuracy				0.98

Berdasarkan Gambar 4.5, evaluasi grafik menunjukkan peningkatan performa model dinilai dari peningkatan *cross-validation score* yang sepadan dengan nilai *training score*. Grafik tersebut memiliki *positive trend*, yaitu kemampuan model dalam mengeneralisasikan data meningkat. Grafik memiliki fluktuasi yang teratur dan tidak terlalu mengikuti (*mirroring*), yang artinya model menangkap struktur mendasar dalam data dan tidak hanya menghafal. Namun, perlu diketahui bahwa pada *training example* <600, *gap* antara *training* dengan *cross-validation score* cukup jauh, hal ini dikarenakan oleh keterbatasan dataset yang digunakan untuk melatih model hanya 1099. Adanya *gap* yang jauh ini menandakan bahwa pada saat awal pelatihan terjadi sedikit *overfitting* terutama pada saat grafik menunjukan *training examples* sekitar 550 sampai sekitar 900 namun, seiring banyaknya data yang dipelajari oleh RF model, *overfitting* tersebut teratasi.

Selain grafiknya yang menunjukkan peningkatan performa, nilai metriks RF model ini juga jauh lebih baik. Selain grafik Gambar 4.4, metriks yang terlihat pada Tabel 4.3 mendukung adanya peningkatan performa, di mana nilai pada tabel ini lebih stabil dibandingkan tabel-tabel metriks sebelumnya. Nilai metriks performa ini harus sesuai dengan kestabilan grafik dan fluktuasi pada *learning curve*. Akurasi yang didapatkan oleh *training score* dan *cross-validation score* pada Gambar 4.4 sangat dekat dan hampir menutup *gap* antara keduanya, sejalan dengan *precision* dan *recall* serta akurasi yang ada pada Tabel 4.3.

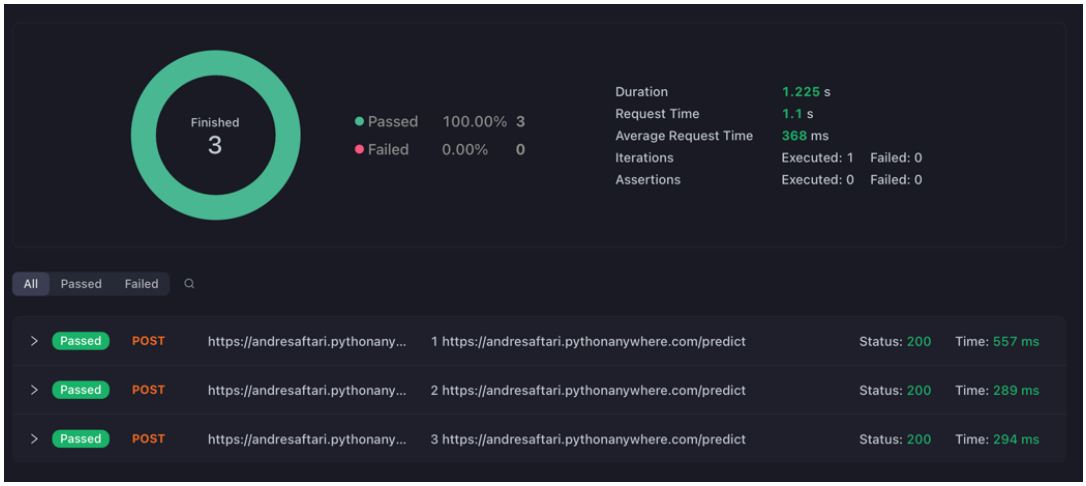
4.5 Evaluasi Performa *Hosted Model* API

Setelah lolos pengujian performa, maka model diekspor ke API *hosting* di *Cloud server service* milik *PythonAnywhere by Anaconda*. Performa API diuji untuk memberikan hasil pengalaman pengguna terbaik pada saat menggunakan aplikasi *monitoring*. Pengujian performa API ini sedikit berbeda dengan pengujian model yang dapat dilakukan tanpa ada pertimbangan lainnya. Performa API ditentukan bukan hanya dari cara kerja dan proses transaksi data yang dilakukan oleh API tersebut, tetapi juga disebabkan oleh *bandwidth* dan kecepatan unggah / unduh pengujinya. Maka dari itu, evaluasi ini dilakukan menggunakan *performance* API *tools* yaitu Apidog, sebuah platform pengujian performa API yang didesain dengan tampilan yang ramah pengguna, ramah pemula [21]. Apidog menyediakan fitur API *test* yang bersifat automatis, artinya penguji cukup menuliskan API *endpoint* lalu menyusun skenario pengujian, misalnya seperti pada penelitian ini dilakukan menguji 3 data sekaligus secara berurutan.

Pengujian API ini menghasilkan metriks berupa durasi respon, *request time,* dan *average request time*. Durasi respon sendiri merupakan lamanya API *endpoint* diakses oleh platform, mulai dari menembak API *endpoint*, lamanya API menangkap *request*, dan lamanya API memberikan *response*. Kemudian ada *request time*, yaitu lamanya API menangkap *request* sejak data diberikan kepada API *endpoint* dan terkadang disediakan juga *average request time* untuk melihat rata-rata *request time* dari seluruh skenario yang diuji oleh platform. Setelah melakukan pengujian, performa API yang didapatkan adalah sebagai Tabel 4.4, yang kemudian dirangkum ke dalam Gambar 4.6 berikut.

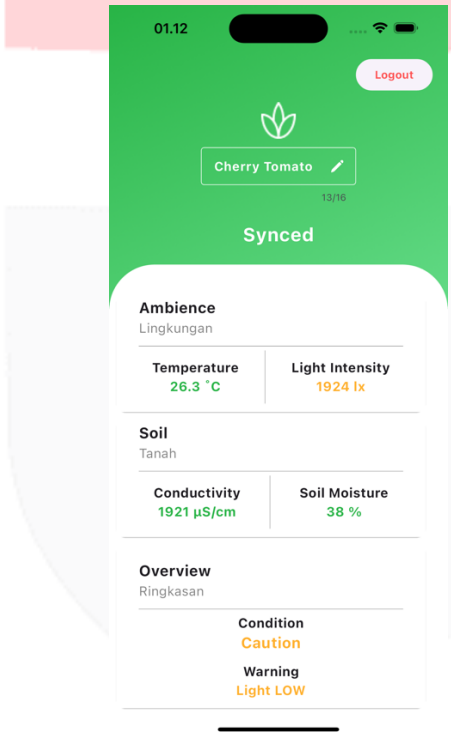
Table 4. 4 *Performance Metrics* API

Duration	**Request**		**Response**	
	Request Time	***Avg. Request Time***	***Response Time***	***Status***
1225ms	1104ms	368ms	1.1s	3 Passed



Gambar 4. 6 API Performance Details

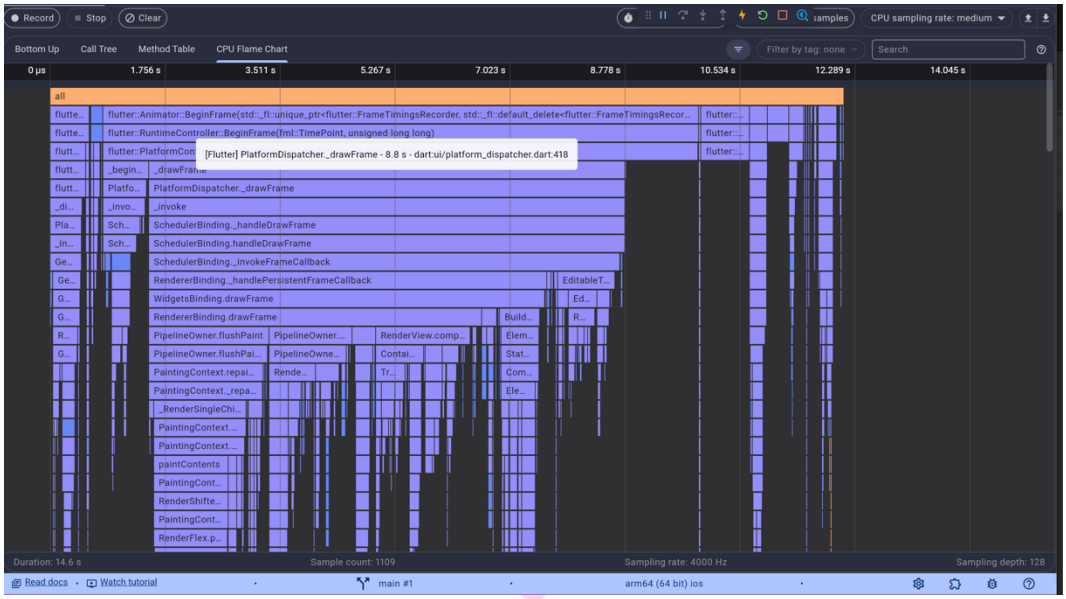
4.6 Evaluasi Performa Aplikasi *Multi-Platform Mobile App*



Gambar 4. 7 Tampilan Aplikasi *Monitoring*

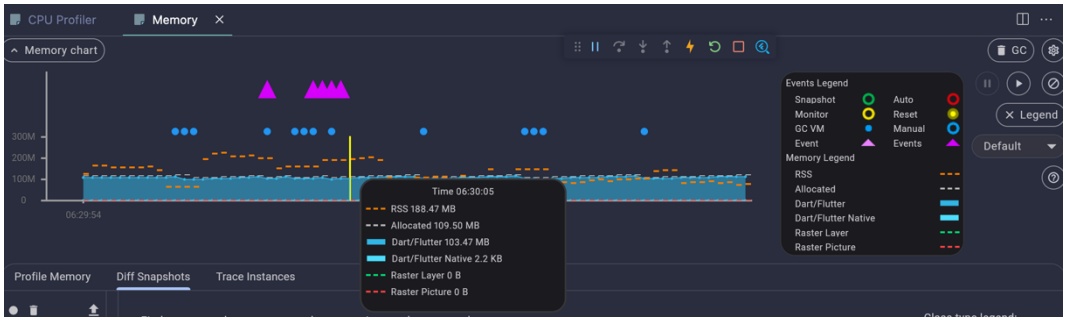
Setelah API berhasil diuji dan dievaluasi, selanjutnya performa aplikasi juga diuji. Aplikasi *monitoring* diuji menggunakan *emulator* dan ponsel fisik untuk mendapatkan hasil yang detail dan konkrit, seperti Gambar 4.7. Aplikasi ini dikembangkan menggunakan Flutter, *framework* pengembangan aplikasi *multi-platform mobile app* yang diciptakan oleh Google menggunakan bahasa Dart [22]. Maka dari itu, aplikasi diuji menggunakan Flutter *DevTools*, fitur yang dipaketkan ke dalam *framework* secara langsung untuk melakukan pengujian performa dan dampak tampilan ke aplikasi. Dalam penelitian ini, pengujian

performa aplikasi dilakukan untuk mengetahui dampak tampilan dan *request* ke API pada saat *real-time monitoring* berlangsung.



Gambar 4. 8 *Flame Chart* pengujian CPU pada Ponsel

Gambar 4.8 merupakan *Flame Chart* yang menggambarkan proses kerja CPU pada ponsel. Pengujian ini dilakukan selama 14 detik, dimulai dari menjalankan aplikasi, melakukan *sign-in* menggunakan akun Google, lalu menunggu aplikasi menampilkan data terakhir dan prediksi model secara *real-time*. Pada *Flame Chart*, semakin banyak proses, maka panjang chart secara vertikalnya semakin panjang menuju ke bawah. Berdasarkan Gambar 4.8, dapat disimpulkan bahwa chart memiliki *total CPU process* sebanyak 1109 samples dan kedalaman proses 128 *depths*. Selama 14 detik, proses menampilkan halaman utama adalah pada saat detik 7.023s sampai 10,534s. Selama 3 detik ini, aplikasi menampilkan UI utama sebelum data diproses, yang mulai pada detik 7,023s sampai 8,778s atau artinya hanya memakan waktu 1,7 detik. Kemudian apabila diteliti kembali pada Gambar 4.8, dimulai dari detik 8,778s ke detik 10,534s kedalaman proses hanya terdapat 3 tumpukan saja, ini adalah proses menampilkan data ke dalam tampilan UI yang sesuai dengan harapan di mana pemrosesan data disarankan untuk tidak lebih dari 3 detik, sebab tentunya akan mempengaruhi pengalaman pengguna.



Gambar 4. 9 *Memory Profiler* yang mengukur Penggunaan RAM pada Ponsel

Gambar 4.9 adalah grafik *Memory Profiler* yang memberikan gambaran visual mengenai penggunaan memori selama pengujian aplikasi. Grafik ini memberikan informasi tentang alokasi dan dealokasi memori seiring berjalannya waktu. Tujuan utama dari *memory profiler* ini adalah identifikasi area ataupun kejadian pada aplikasi yang mungkin tergolong penggunaan memori yang tidak efisien. Selain itu, pada pengembangan aplikasi *mobile* maupun *multi-platform mobile app* dikenali adanya masalah berupa kebocoran memori (*memory leak*) yang pada umumnya memiliki gejala aplikasi mengalami *crash*, *forced close* ataupun *not responding* (*hanged*). Kebocoran memori ini dapat diidentifikasi menggunakan grafik *memory profiler*.

Adapun cara membaca grafik pada Gambar 4.9 adalah sebagai berikut. Sumbu X merepresentasikan waktu eksekusi, sedangkan sumbu Y menunjukkan ukuran memori yang digunakan. Garis-garis maupun simbol lingkaran dan segitiga berwarna mewakili jenis memori yang berbeda, seperti memori yang digunakan oleh objek *Dart*, *RSS*, dan *raster layer* (sistem). Pada grafik tersebut, dapat disimpulkan dari total 14 detik pengujian, pada detik 6,30s tercatat aktivitas memori yang cukup besar disebabkan oleh proses *sign-in* pengguna dan *rendering* saat menuju halaman utama, dengan penggunaan RSS (memori fisik yang memuat aplikasi) sebesar 188,47 MB, Dart/Flutter sebesar 103,47 MB dan *Allocated Memory* sebesar 109,50 MB. Sama seperti evaluasi CPU, aktivitas dan penggunaan memori secara total sesuai dengan yang diharapkan karena memiliki total penggunaan di bawah 400 MB yang dapat menyebabkan aplikasi menjadi berat.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Tugas akhir ini mengembangkan sistem pemantauan kondisi tanaman menggunakan *Flower Care Sensor* dan metode *Fuzzy-Random Forest*, yang dirancang untuk mengukur parameter penting berupa *intensitas cahaya*, kelembaban tanah, suhu, dan *electrical conductivity*. Hasil penelitian menunjukkan bahwa model *Fuzzy-Random Forest* mampu mengolah data sensor dengan akurasi tinggi, memberikan informasi *real-time* yang dapat diandalkan. Dengan sistem ini, petani maupun peneliti hidroponik menjadi lebih proaktif dalam mengelola tanaman, meningkatkan hasil pertanian, dan mengurangi risiko kerugian akibat kondisi lingkungan yang ekstrem. Sistem ini dibangun sejalan dengan tujuan *Sustainable Development Goals* (SDG) untuk mencapai ketahanan pangan, serta sesuai dengan tujuan yaitu membantu pertanian pintar dan berkelanjutan dengan menciptakan sistem *early warning* untuk keadaan lingkungan yang dapat menyebabkan tanaman terancam.

5.2. Saran

Hasil dari tugas akhir ini sudah sangat baik dan sesuai dengan yang diperkirakan, namun dataset dapat ditarik lebih banyak lagi dan beragam penelitian berikutnya dilakukan. Penyesuaian *hyperparameter* dan evaluasi *metrics* juga merupakan faktor penting yang perlu diperhatikan untuk meningkatkan performa model agar lebih optimal. Namun, metode ini dapat diterapkan dan dikembangkan dalam konteks yang lebih luas, seperti memprediksi kondisi tanaman di berbagai jenis lahan pertanian maupun hidroponik.

DAFTAR PUSTAKA

[1] M. Brown, "Smart Farming Technologies for Sustainable Agriculture: A Comprehensive Review.," *Sustainability,* vol. 10, no. 8, pp. 1-12, 2018.

[2] Y. Kim and S. Lee, "IoT-based Monitoring Systems for Extreme Environmental Conditions in Agriculture," *Sensors,* vol. 21, no. 6, pp. 5-18, 2021.

[3] Jan-Willem, "Kruse Smart Home | Article," 13 Mar 2023. [Online]. Available: https://smarthome.familykruse.eu/the-flower-care-plant-sensor-from-xiaomi/. [Accessed 1 Dec 2023].

[4] A. Smith and B. Jones, "Advanced Agricultural Sensors: A Review of Current Trends," *Journal of Agricultural Technology,* vol. 14, no. 2, pp. 46-48, 2019.

[5] S. Mujab, "IMPLEMENTASI FUZZY INFRENCE SYSTEM METODE MAMDANI MOM (MEAN OF MAXIMUM METHOD) UNTUK KLASIFIKASI KELOMPOK BELAJAR SISWA BARU," 8 Nov 2018. [Online]. Available: http://eprints.umg.ac.id/626/. [Accessed 1 Dec 2023].

[6] "Ensiklopedia Dunia," [Online]. Available: https://p2k.stekom.ac.id/ensiklopedia/Random_forest. [Accessed 5 December 2023].

[7] P. Bonissone, J. Cadenas, C. Garrido and R. Andres, "A Fuzzy Random Forest," *International Journal of Approximate Reasoning,* vol. 51, no. 7, 2010.

[8] N. N. P. C. Binaraesa, S. M. Sutan and A. M. Ahmad, "NILAI EC (ELECTRO CONDUCTIVITY) BERDASARKAN UMUR TANAMAN SELADA DAUN HIJAU (Lactuca sativa L.) DENGAN SISTEM HIDROPONIK NFT (NUTRIENT FILM TECHNIQUE)," *Jurnal Keteknikan Pertanian Tropis dan Biosistem,* vol. 4, no. 1, pp. 65-74, February 2016.

[9] "Ensiklopedia Dunia," [Online]. Available: https://p2k.stekom.ac.id/ensiklopedia/PH. [Accessed 5 December 2023].

[10] "en.m.nu," [Online]. Available: https://en.m.nu/other-7/mi-flora-plant-sensor. [Accessed 1 December 2023].

[11] Raspberry Pi , "Raspberry Pi 4 Model B Tech Specs," [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. [Accessed 30 November 2023].

[12] "Bluetooth," [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/. [Accessed 1 December 2023].

[13] Gomez, Oller, Paradells and Baldini, "Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology," *Sensors,* vol. 12, no. 9, 2012.

[14] "JSON Documentation," [Online]. Available: https://stleary.github.io/JSON-java/index.html. [Accessed 5 Juli 2024].

[15] "Pandas Dataframe Documentation," 2024. [Online]. Available: https://pandas.pydata.org/docs/reference/frame.html. [Accessed 2024 Juli 5].

[16] A. A. Awan, "KDnuggets," 29 November 2022. [Online]. Available: https://www.kdnuggets.com/2022/11/introduction-smote.html. [Accessed 12 June 2024].

[17] A. Kharwal, "The Clever Programmer - StandardScaler in Machine Learning," 22 September 2020. [Online]. Available: https://thecleverprogrammer.com/2020/09/22/standardscaler-in-machine-learning/#google_vignette. [Accessed 14 June 2024].

[18] "ivosights - Mengenal Apa Itu Hyperparameter Tuning dalam Machine Learning," 31 Januari 2023. [Online]. Available: https://ivosights.com/read/artikel/machine-learning-mengenal-apa-itu-hyperparameter-tuning-dalam. [Accessed 20 Juli 2024].

[19] "Revoupedia - Apa itu Cross Validation?," Revoupedia, [Online]. Available: https://revou.co/kosakata/cross-validation. [Accessed 20 Juli 2024].

[20] "DQLab - Tipe Machine Learning dengan K-Fold Cross Validation," DQLab, 5 Oktober 2023. [Online]. Available: https://dqlab.id/tipe-machine-learning-dengan-k-fold-cross-validation. [Accessed 20 Juli 2024].

[21] "Apidog," [Online]. Available: https://apidog.com. [Accessed 2024 Juli 20].

[22] "Flutter Docs," [Online]. Available: https://flutter.dev. [Accessed 2024 Juli 20].

[23] "Firebase Docs | Firebase with Python Admin-SDK Setup," [Online]. Available: https://firebase.google.com/docs/database/admin/start?hl=en&authuser=1#authenticate-with-admin-privileges. [Accessed 25 Juli 2024].

[24] P. R. M. S. Pratiwi and E. Mustari, "PENGARUH TINGKAT EC (ELECTRICAL CONDUCTIVITY) TERHADAP PERTUMBUHAN TANAMAN SAWI (Brassica juncea L.) PADA SISTEM INSTALASI AEROPONIK VERTIKAL," *Jurnal Agro,* vol. II, no. 1, p. 51, 2015.

LAMPIRAN

Lampiran 1: Kode Program

Lampiran 1 adalah kode program yang digunakan pada pengembangan tugas akhir ini, di mana kode program tersebut dibagi menjadi 3 proses, yaitu:

1. Data Gathering

Proses *data gathering* menggunakan program berbahasa Python untuk menghubungkan sensor dengan Raspberry Pi 4 Model B, mengambil data setiap 5 menit dan mengirimnya ke *Firebase Realtime Database*.
https://github.com/andresaftari/MoniFlora-Skripsi-CLI

2. Model Engineering

Proses *model engineering* menggunakan program berbahasa Python untuk melakukan *data preprocessing*, melatih *Fuzzy–Random Forest* model mengambil data, serta mengevaluasi model.
https://github.com/andresaftari/MoniFlora-ModelTraining

3. Multi-Platform Mobile Monitoring App

Proses *app development* menggunakan program dengan Flutter (*framework* pengembangan *multi-platform* berbahasa Dart) ditujukan untuk *platform* Android dan iOS.
https://github.com/andresaftari/MoniFlora-Mobile

Lampiran 2: Dataset Pengujian

Dataset pengujian yang diekspor dari *Firebase Realtime Database* memiliki format JSON yang khusus hanya untuk Firebase RTDB. Maka dari itu, untuk mencoba dataset JSON ini dapat langsung diimpor ke projek Firebase RTDB baru, diubah ke CSV ataupun menggunakan langsung JSON tersebut, namun tetap harus dirapikan karena format strukturnya, terutama *List*-nya khusus dan berbeda dengan JSON pada umumnya.
https://drive.google.com/file/d/1KIBmHYDoeW1ZyN90ukG6qMIFbfcyLFzK/view

Dataset tersebut dapat digunakan oleh Python langsung, namun perlu dihubungkan Firebase RTDB admin-sdk, yang sesuai dengan dokumentasi Firebase RTDB pada Python [23].
https://drive.google.com/file/d/1Pw5mcL-vvsgRvnbu0_Lif5NvuBBGL6eA/view

Lampiran 3: *Hosted Model*

Model sudah di-*hosting* ke *Cloud server* milik *PythonAnywhere by Anaconda*. Model ini dapat dicoba secara langsung menggunakan bahasa pemrograman apapun, Python *request*, Javascript, metode CURL, ataupun menggunakan platform API testing seperti Postman dan Apidog. Format untuk menguji API ini adalah sebagai berikut.

Method	POST
Endpoint	https://andresaftari.pythonanywhere.com/predict
Body type	Raw JSON
Body	{"temperature": 26.2, "light": 4321.0, "conductivity": 2123.0, "moisture": 36.0}
Content-Type	Application/Json
Success Code	200

Response	{
	"prediction": "Optimal",
	"prediction_index": 0,
	"probability": [
	0.9992288689909189,
	0.0007711310090810949,
	0.0
	]
	}

Response Schema	{
	"prediction": "string",
	"prediction_index": “integer”,
	"probability": [
	float,
	float,
	float
	]
	}