LAMPIRAN

Lampiran 1. Surat Kesepakatan Bimbingan Skripsi

SURAT KESEPAKATAN BIMBINGAN TUGAS AKHIR

Kami yang bertanda tangan di bawah ini :

Pihak Pertama

Nama	: Nur Rizqi Maulana
NIM	: 21090068
Program Studi	: Sarjana Terapan Teknik Informatika

Pihak Kedua

Nama	: Hepatika Zidny Ilmadina, S.Pd., M.Kom.
Status	: Dosen
NIDN	: 0618119101
Jabatan Fungsional	: Asisten Ahli
Pangkat/Golongan	: Penata Muda Tingkat I/III-b

Pada hari ini Kamis tanggal 06 Maret 2025 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I/II Tugas Akhir Pihak Pertama dengan syarat Pihak Pertama wajib melakukan bimbingan Tugas Akhir minimal 8 kali kepada Pihak Kedua. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

Demikian kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas Akhir

Tegal, 06 Maret 2025

Pihak Pertama	Pihak Kedua
Nur Rizqi Maulana	Hepatika Zidny Ilmadina, S.Pd., M.Kom. NIPY: 08.017.340

Mengetahui
Ketua Program Studi Sarjana Terapan Teknik Informatika

Dyah Apriliani, S.T., M.Kom.
NIPY. 09.015.225

SURAT KESEPAKATAN BIMBINGAN TUGAS AKHIR

Kami yang bertanda tangan di bawah ini :

Pihak Pertama

Nama	: Nur Rizqi Maulana
NIM	: 21090068
Program Studi	: Sarjana Terapan Teknik Informatika

Pihak Kedua

Nama	: Mirza Alim Mutasodirin, S.Kom., M.Kom.
Status	: Dosen
NIDN	: 0605049004
Jabatan Fungsional	: Asisten Ahli
Pangkat/Golongan	: III/c

Pada hari ini Kamis tanggal 06 Maret 2025 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I/II Tugas Akhir Pihak Pertama dengan syarat Pihak Pertama wajib melakukan bimbingan Tugas Akhir minimal 8 kali kepada Pihak Kedua. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

Demikian kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas Akhir

Tegal, 06 Maret 2025

Pihak Pertama	Pihak Kedua
Nur Rizqi Maulana	Mirza Alim Mutasodirin, S.Kom., M.Kom. NIPY: 03.023.534

Mengetahui
Ketua Program Studi Sarjana Terapan Teknik Informatika

Dyah Aprilliani, S.T., M.Kom.
NIPY: 09.015.225

Lampiran 2. Surat Pernyataan Pengajuan HKI

SURAT PERNYATAAN

Yang bertanda tangan di bawah ini, pemegang hak cipta:

1. Nama : Nur Rizqi Maulana
   Kewarganegaraan : Indonesia
   Alamat : Jalan Sunan Amangkurat 1 No. 11. Desa Lemahduwur Rt.011/02, Kecamatan Adiwerna, Kabupaten Tegal, Provinsi Jawa Tengah, 52194

2. Nama : Hepatika Zidny Ilmadina, S.Pd., M.Kom.
   Kewarganegaraan : Indonesia
   Alamat : Jalan Kenanga Gang I Nomor 9, Kelurahan Mangkukusuman, Kecamatan Tegal Timur, Kota Tegal, Provinsi Jawa Tengah, 5212

3. Nama : Mirza Alim Mutasodirin, S.Kom., M.Kom.
   Kewarganegaraan : Indonesia
   Alamat : Griya Santika blok J no. 11, Desa Pengabean, Kec. Dukuhturi, Kab. Tegal, Jawa Tengah, Indonesia Kode Pos 52192

Dengan ini menyatakan bahwa:

1. Karya Cipta yang saya mohonkan:
   Berupa : Program Komputer
   Berjudul : HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0
   - Tidak meniru dan tidak sama secara esensial dengan Karya Cipta milik pihak lain atau obyek kekayaan intelektual lainnya sebagaimana dimaksud dalam Pasal 68 ayat (2);
   - Bukan merupakan Ekspresi Budaya Tradisional sebagaimana dimaksud dalam Pasal 38;
   - Bukan merupakan Ciptaan yang tidak diketahui penciptanya sebagaimana dimaksud dalam Pasal 39;
   - Bukan merupakan hasil karya yang tidak dilindungi Hak Cipta sebagaimana dimaksud dalam Pasal 41 dan 42;
   - Bukan merupakan Ciptaan seni lukis yang berupa logo atau tanda pembeda yang digunakan sebagai merek dalam perdagangan barang/jasa atau digunakan sebagai lambang organisasi, badan usaha, atau badan hukum sebagaimana dimaksud dalam Pasal 65 dan;
   - Bukan merupakan Ciptaan yang melanggar norma agama, norma susila, ketertiban umum, pertahanan dan keamanan negara atau melanggar peraturan perundang-undangan sebagaimana dimaksud dalam Pasal 74 ayat (1) huruf d Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

2. Sebagai pemohon mempunyai kewajiban untuk menyimpan asli contoh ciptaan yang dimohonkan dan harus memberikan apabila dibutuhkan untuk kepentingan penyelesaian sengketa perdata maupun pidana sesuai dengan ketentuan perundang-undangan.

3. Karya Cipta yang saya mohonkan pada Angka 1 tersebut di atas tidak pernah dan tidak sedang dalam sengketa pidana dan/atau perdata di Pengadilan.

4. Dalam hal ketentuan sebagaimana dimaksud dalam Angka 1 dan Angka 3 tersebut di atas saya / kami langgar, maka saya / kami bersedia secara sukarela bahwa:
   a. permohonan karya cipta yang saya ajukan dianggap ditarik kembali; atau
   b. Karya Cipta yang telah terdaftar dalam Daftar Umum Ciptaan Direktorat Hak Cipta, Direktorat Jenderal Hak Kekayaan Intelektual, Kementerian Hukum Dan Hak Asasi Manusia R.I dihapuskan sesuai dengan ketentuan perundang-undangan yang berlaku.
   c. Dalam hal kepemilikan Hak Cipta yang dimohonkan secara elektronik sedang dalam berperkara dan/atau sedang dalam gugatan di Pengadilan maka status kepemilikan surat pencatatan elektronik tersebut ditangguhkan menunggu putusan Pengadilan yang berkekuatan hukum tetap.

Demikian Surat pernyataan ini saya/kami buat dengan sebenarnya dan untuk dipergunakan sebagimana mestinya.

Tegal, 23 Juni 2025

Nur Rizqi Maulana
Pemegang Hak Cipta*

Hepatika Zidny Ilmadina, S.Pd., M.Kom.
Pemegang Hak Cipta*

Mirza Alim Mutasodirin, S.Kom., M.Kom.
Pemegang Hak Cipta*

* Semua pemegang hak cipta agar menandatangani di atas materai.

Lampiran 3. Surat Pengalihan HKI

SURAT PENGALIHAN HAK CIPTA

Yang bertanda tangan di bawah ini :

1. Nama : Nur Rizqi Maulana
   Kewarganegaraan : Indonesia
   Alamat : Jalan Sunan Amangkurat 1 No. 11. Desa Lemahduwur Rt.011/02, Kecamatan Adiwerna, Kabupaten Tegal, Provinsi Jawa Tengah, 52194

2. Nama : Hepatika Zidny Ilmadina, S.Pd., M.Kom.
   Kewarganegaraan : Indonesia
   Alamat : Jalan Kenanga Gang I Nomor 9, Kelurahan Mangkukusuman, Kecamatan Tegal Timur, Kota Tegal, Provinsi Jawa Tengah, 5212

3. Nama : Mirza Alim Mutasodirin, S.Kom., M.Kom.
   Kewarganegaraan : Indonesia
   Alamat : Griya Santika blok J No. 11, Desa Pengabean, Kec. Dukuhturi, Kab. Tegal, Jawa Tengah, Indonesia Kode Pos 52192

Adalah **Pihak I** selaku pencipta, dengan ini menyerahkan karya ciptaan saya kepada :

Nama : Pusat Penelitian dan Pengabdian Masyarakat (P3M)
Politeknik Harapan Bersama

Alamat : Jl. Mataram No.9, Pesurungan Lor, Kec. Margadana,
Kota Tegal, Provinsi Jawa Tengah, 52147

Adalah **Pihak II** selaku Pemegang Hak Cipta berupa Program Komputer dengan judul :**"HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0"** untuk didaftarkan di Direktorat Hak Cipta dan Desain Industri, Direktorat Jenderal Kekayaan Intelektual, Kementerian Hukum dan Hak Asasi Manusia Republik Indonesia.

Demikianlah surat pengalihan hak ini kami buat, agar dapat dipergunakan sebagaimana mestinya.

Tegal, 23 Juni 2025

Pemegang Hak Cipta
Kepala P3M

(Muhammad Fikri Hidayattullah, S.T., M.Kom)

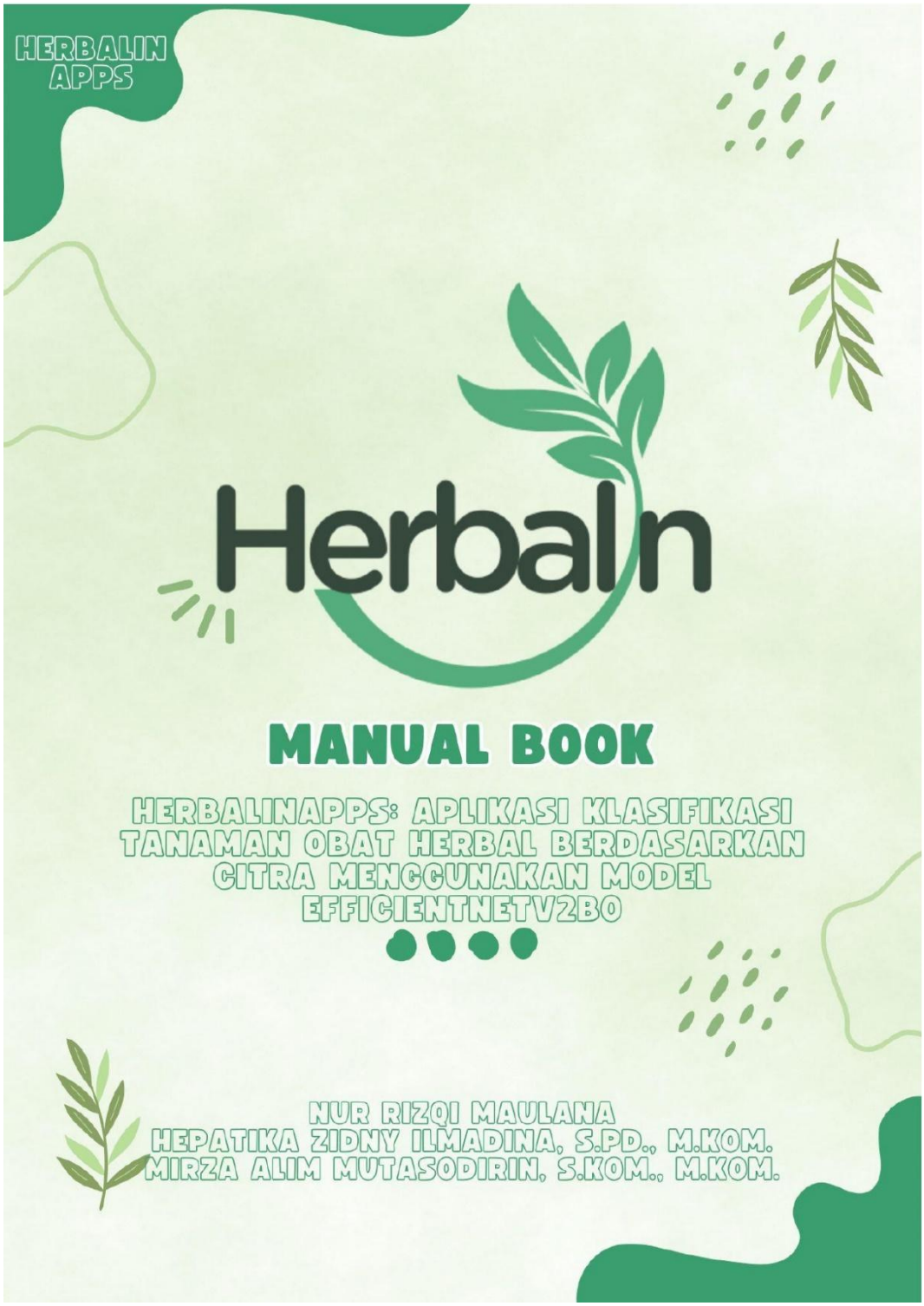
Pencipta

(Nur Rizqi Maulana)

(Hepatika Zidny Ilmadina, S.Pd., M.Kom.)

(Mirza Alim Mutasodirin, S.Kom., M.Kom.)

Lampiran 4. Manual Book dan Dokumen Teknikal



Daftar Isi

Pendahuluan

Dokumen manual ini untuk memberikan panduan teknis dan fungsional kepada pengguna dalam mengoperasikan "HerbalinApps:Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0". Panduan ini bertujuan membantu pengguna memahami proses instalasi aplikasi, navigasi antar muka, serta memaksimalkan fitur fitur yang tersedia. Selain itu, dokumen ini berfungsi sebagai dokumentasi resmi yang menjelaskan alur penggunaan sistem mekanisme kerja aplikasi, sehingga pengguna dapat menjalankan aplikasi dengan efisien dan sesuai dengan tujuan pengembangannya.

Aplikasi Klasifikasi Tanaman Obat Herbal adalah aplikasi berbasis mobile yang memanfaatkan teknologi Convolutional Neural Network (CNN) dengan arsitektur EfficientNetV2B0 untuk mengklasifikasikan tanaman obat herbal berdasarkan citra. Tujuan utama aplikasi ini adalah untuk memudahkan masyarakat dalam mengidentifikasi berbagai jenis tanaman obat herbal hanya dengan mengambil gambar. Selain itu, aplikasi ini dilengkapi dengan informasi mengenai manfaat dan cara pengolahan tanaman, sehingga berfungsi sebagai edukasi yang praktis dan informatif. Aplikasi ini dibangun menggunakan teknologi pemrograman kotlin dan tensorflow yang mendukung pengolahan citra.

Dokumen ini menyajikan informasi lengkap mengenai petunjuk teknis penggunaan aplikasi, mulai dari proses instalasi, tampilan awal, navigasi antarmuka, hingga cara kerja fitur yang tersedia untuk pengguna, serta langkah langkah penggunaan aplikasi dari awal hingga akhir. Dengan adanya manual book ini, diharapkan pengguna dapat memahami dan memanfaatkan aplikasi secara optimal sesuai dengan fungsinya. Dokumen ini berisikan informasi sebagai berikut:

- Informasi umum yang merupakan bagian pendahuluan, yang meliputi tujuan pembuatan dokumen, deskripsi umum sistem serta deskripsi dokumen
- Berisi perangkat yang dibutuhkan untuk sistem " HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0" meliputi perangkat lunak dan perangkat keras.
- Berisi pengguna manual sistem "HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0" untuk pengguna

Spesifikasi Perangkat

Persyaratan Sistem Operasi

Komponen	Spesifikasi
Minimum Android	Android 7.0 (Nougat) – API Level **24**
Target Android	Android 14 – API Level **34**
Compile SDK	Android SDK 34

Spesifikasi Minimum Perangkat

Prosesor (CPU)	Quad-core ARM Cortex-A53 atau setara
RAM	**2 GB RAM** atau lebih
Penyimpanan	Tersedia minimal **100 MB** ruang kosong
Resolusi Layar	Minimum **720 x 1280 piksel (HD)**
Fitur Tambahan	Kamera belakang (jika diperlukan oleh fitur deteksi)

Izin Aplikasi (Permissions)

- Akses Kamera (jika digunakan untuk klasifikasi citra tanaman)
- Akses Penyimpanan (untuk menyimpan atau membaca gambar)
- Akses Lokasi (jika digunakan untuk mencatat lokasi tanaman)
- Akses Internet (untuk menghubungkan ke server)

Catatan Tambahan

- Aplikasi **tidak dapat diinstal pada perangkat dengan versi Android di bawah 7.0 (API 24)**
- Performa optimal diperoleh pada perangkat dengan Android 10 ke atas

Fitur dan Cara Penggunaan

Struktur Fitur

Struktur halaman pada sistem " HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0" adalah sebagai berikut :

1. Fitur Register User
2. Fitur Login User
3. Fitur Home
4. Fitur Klasifikasi Tanaman Obat Herbal
5. Fitur Daftar Penyakit Ringan
6. Fitur My Plant
7. Fitur Lokasi Tanaman (gmaps)
8. Fitur History klasifikasi Tanaman
9. Fitur Profil Pengguna

Cara Penggunaan Aplikasi

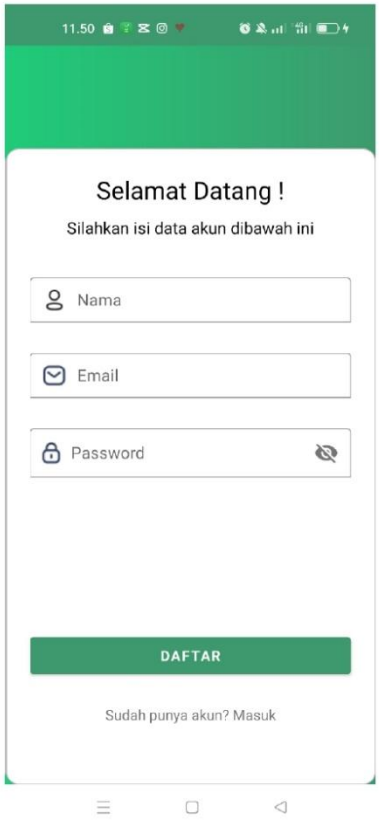
1. Register User

a. Buka aplikasi Herbalin Apps.

b. Pilih "Daftar" untuk membuat akun baru.

c. Isi formulir pendaftaran dengan informasi yang diminta, seperti, nama, email dan password.

d. Klik "Daftar" untuk menyelesaikan proses pendaftaran.



Gambar Register

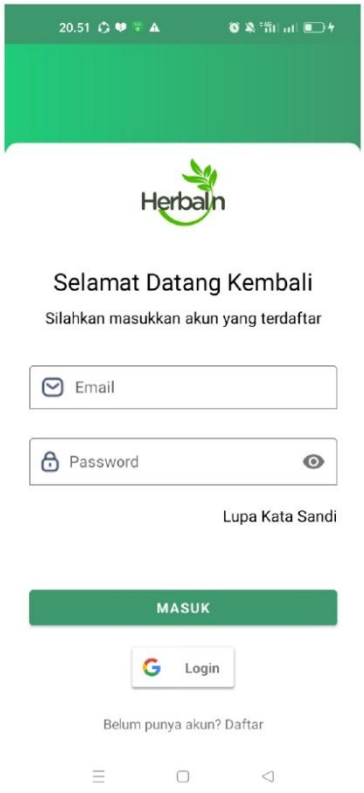
2. Login User

a. Buka aplikasi Herbalin Apps.

b. Masukkan email dan kata sandi yang telah Anda daftarkan.

c. Klik "Masuk" untuk masuk ke aplikasi Menggunakan akun Anda.

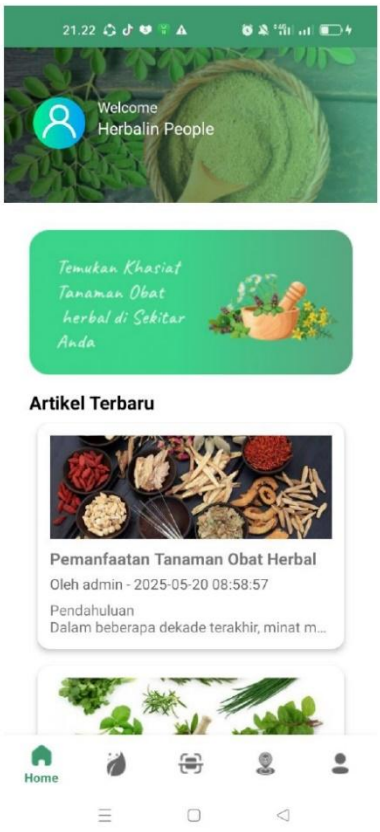
d. Pengguna juga dapat login Menggunakan Googe Account.



Gambar Login

3. Dashboard

Setelah berhasil login, Anda akan diarahkan ke dashboard. Di sini Anda dapat mengakses berbagai fitur utama aplikasi, termasuk klasifikasi tanaman obat herbal, posting tanaman herbal, melihat lokasi tanaman herbal, profile. Yang bisa di akses melalui navigation button

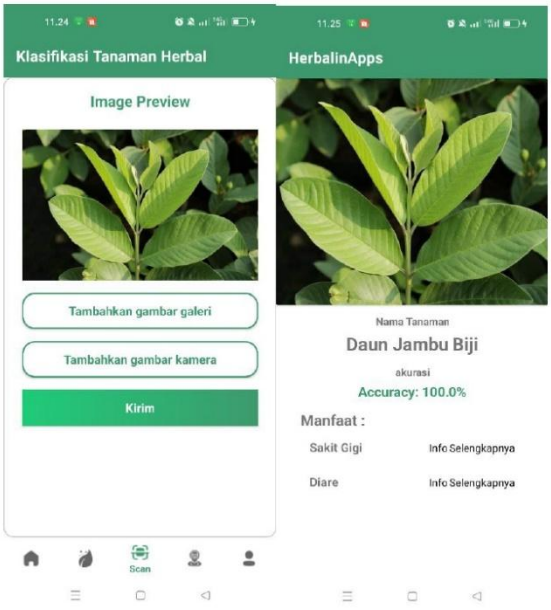


Gambar Dashboard Home

4. Klasifikasi Tanaman Obat Herbal

Aplikasi ini menyediakan fitur klasifikasi tanaman herbal yang dirancang untuk membantu Anda mengidentifikasi jenis tanaman herbal beserta Manfaat dan cara pengolahannya. Berikut cara mengaksesnya:

a. Pilih Menu "Scan" pada navigation button, Maka akan diarahkan ke halaman klasifikasi.
b. Tambahkan gambar dari galeri atau kamera.
c. Klik "kirim" untuk melakukan proses identifikasi tanaman.
d. Setelah selesai, Anda akan mendapatkan hasil identifikasi tanaman beserta manfaatnya. Anda dapat mengklik "info selengkapnya " untuk detail cara pengolahan tanaman tersebut.

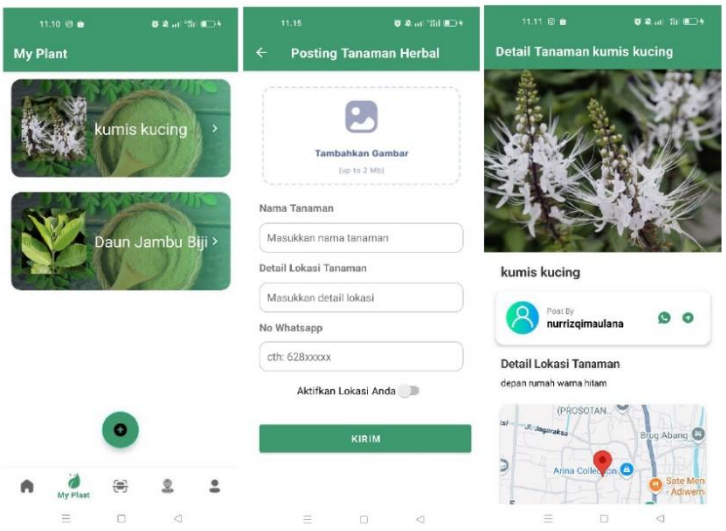


Gambar Klasifikasi Tanaman Obat Herbal

5. Fitur My Plant

fitur My plant yang dirancang untuk membantu melakukan posting dan berbagi tanaman yang ada disekitar lokasi pengguna. Berikut cara mengaksesnya:

a. Pilih Menu “My Plant” pada navigation button, Maka akan diarahkan ke halaman daftar My Plant.
b. Klik tombol (+) untuk diarahkan ke halaman form posting tanaman
c. Masukkan informasi pada form yang telah disediakan
d. Aktfikan lokasi anda
e. Klik “Kirim” untuk melakukan proses posting tanaman tersebut.
f. Setelah selesai, Anda dapat melihat daftar posting tanaman herbal pada halaman awal My Plant.

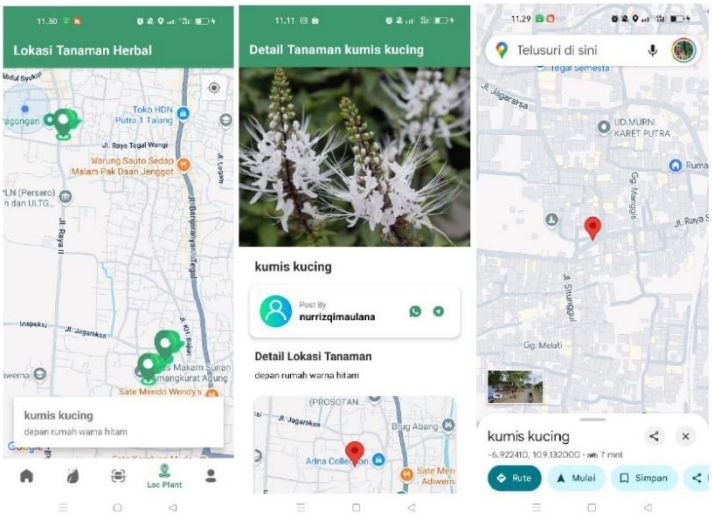


Gambar My Plant

6. Fitur Loc Plant

Fitur Loc plant yang dirancang untuk membantu mencari tanaman yang ada disekitar lokasi pengguna dengan menampilkan titik titik tanaman pada gMaps. Berikut cara mengaksesnya:

a. Pilih Menu “Loc Plant” pada navigation button, Maka akan diarahkan ke halaman lokasi Loc Plant.
b. Setelah itu akan menampilkan titik titik lokasi tanaman herbal.
c. Pengguna dapat mengklik pin lokasi tanaman tersebut dan akan ditampilkan detail lokasi tanaman tersebut

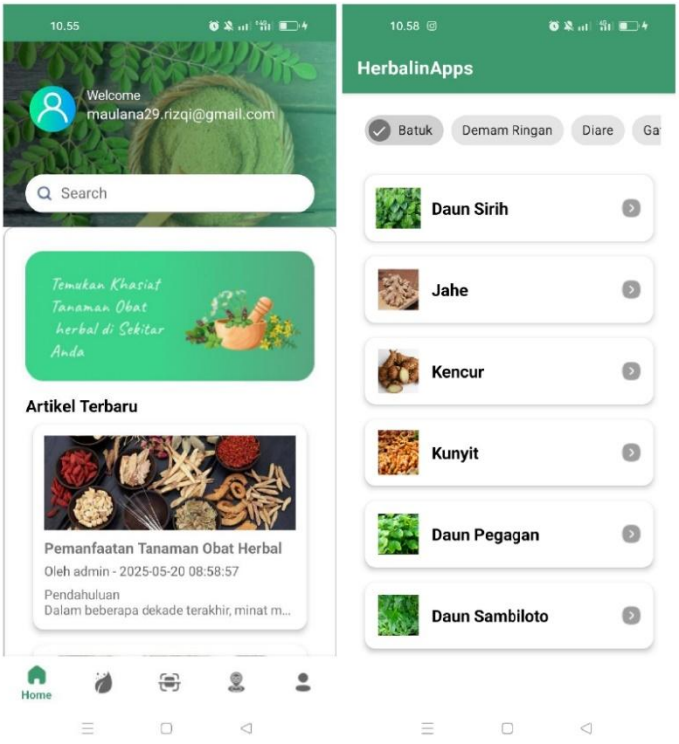


Gambar Loc Plant

7. Fitur Daftar Penyakit Ringan

Fitur ini dirancang untuk membantu pengguna melakukan filter tanaman obat herbal yang dapat meredakan penyakit ringan. Berikut cara mengaksesnya:

a. Klik Banner "Temukan Khasiat Tanaman Obat Herbal" pada halaman dashboard
b. Setelah itu pilih salah satu daftar penyakit ringan yang ada di list.
c. Lalu akan muncul daftar tanaman herbal berdasarkan jenis penyakit ringan yang dipilih.

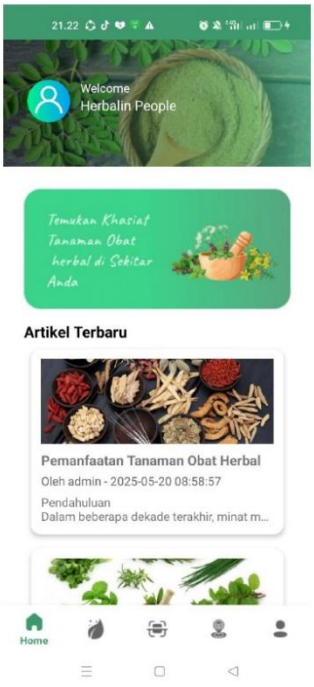


Gambar List Penyakit Ringan

8. Fitur Artikel Tentang Tanaman Obat Herbal

Aplikasi ini menyediakan berbagai artikel yang berfokus pada tanaman obat herbal:

a. Pilih "Artikel" pada halaman dashboard.



b. Telusuri dan baca artikel yang tersedia.

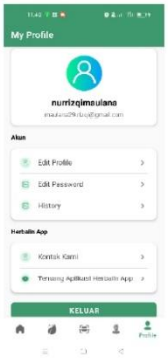
c. Artikel ini mencakup topik-topik yang berhubungan dengan tanaman obat herbal.

Gambar Artikel

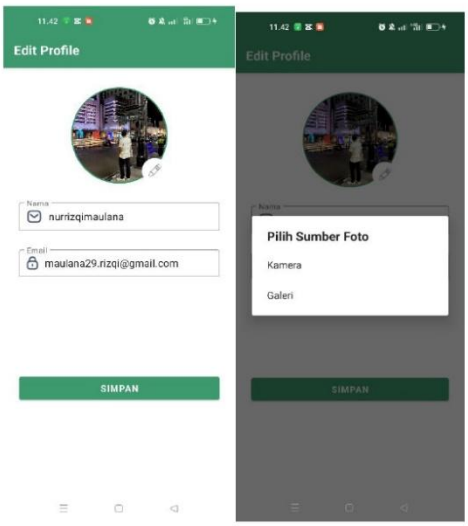
9. Fitur Edit Profile dan Password

Anda dapat mengakses dan mengatur informasi akun Anda di menu profil:

a. Pilih Menu “Profile” pada navigation button



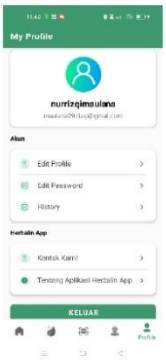
b. Klik “edit profile” pada halaman my profile
c. Edit informasi profile seperti nama, email, dan foro profile(opsional).
d. Simpan perubahan untuk memperbarui profil Anda.



Gambar Edit Profile

Anda dapat mengakses dan mengubah password akun Anda di menu profil:

e. Pilih Menu “Profile” pada navigation button



f. Klik “edit password” pada halaman my profile

g. Masukkan password lama dan password yang baru

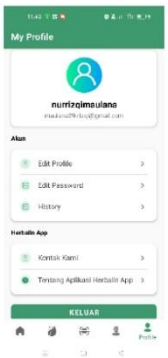
h. Simpan perubahan untuk memperbarui password Anda.

Gambar Edit Password

10. History Klasifikasi

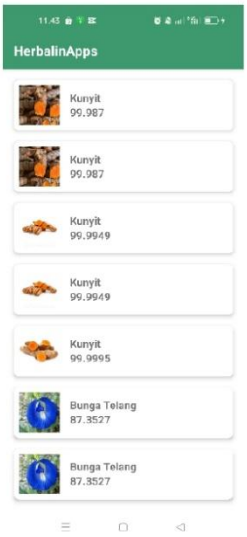
Fitur ini dirancang untuk melihat history/riwayat klasifikasi tanaman obat herbal yang pernah dilakukan pengguna. Berikut cara mengaksesnya

a. Pilih Menu “Profile” pada navigation button



b. Klik “History” pada halaman my profile.

c. Lalu pengguna akan diarahkan ke halaman daftar history klasifikasi



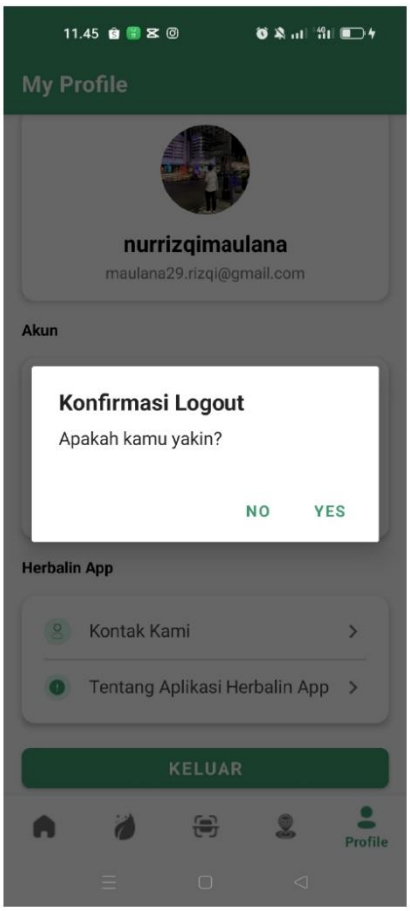
Gambar History Klasifikasi

11. Logout Aplikasi

Untuk keluar dari aplikasi:

a. Pilih menu "Logout" di halaman my profile.

b. Konfirmasi untuk keluar dari akun Anda.



Gambar Logout Aplikasi

Pemecahan Masalah

1. Tidak Bisa Login
    - Periksa Koneksi Internet: Pastikan perangkat Anda terhubung ke internet.
    - Cek Email dan Kata Sandi: Pastikan Anda memasukkan email dan kata sandi yang benar.
2. Aplikasi Tidak Merespon / Crash

    Tutup dan Buka Kembali Aplikasi: Coba tutup aplikasi dan buka kembali.
3. Tidak Bisa Ambil Gambar dan kirim Gambar Saat Klasifikasi
    - Pastikan anda sudah mengizinkan akses kamera diperangkat anda
    - Periksa Koneksi Internet: Pastikan perangkat Anda terhubung ke internet.


HERBALIN
APPS
HerbalIn
DOKUMEN TEKNIS
HERBALINAPPS: APLIKASI KLASIFIKASI TANAMAN OBAT HERBAL BERDASARKAN CITRA MENGGUNAKAN MODEL EFFICIENTNETV2B0
NUR RIZQI MAULANA
HEPATIKA ZIDNY ILMADINA, S.PD., M.KOM.
MIRZA ALIM MUTASODIRIN, S.KOM., M.KOM.

Daftar Isi

Pendahuluan

Indonesia merupakan salah satu negara di dunia yang terkenal memiliki keanekaragaman hayati yang besar dimana memiliki sekitar 30.000 jenis tanaman yang merupakan 75% dari jumlah tanaman didunia sehingga Indonesia dikenal dengan mega-center keanekaragaman hayati dunia. Berdasarkan data riset dari Badan Litbang Kementrian Kesehatan (Riset Tumbuhan Obat dan Jamu/RISTOJA), telah ditemukan sebanyak 10.047 ramuan tradisional yang telah digunakan oleh masyarakat Indonesia untuk pengobatan 74 indikasi penyakit.[1] Tanaman obat yang ada di sekitar lingkungan hidup memiliki nilai yang sangat penting. Keberadaan tanaman obat menjadi solusi dalam memberikan perawatan kesehatan alternatif untuk meredakan gejala penyakit umum, mengobati luka ringan, atau menjaga kesehatan secara umum. Beberapa tanaman obat yang berkhasiat untuk mengurangi nyeri yaitu jahe, tyme (mint), teh hijau, lidah buaya, temulawak, kunyit. Tanaman- tanaman tersebut mempunyai khasiat sebagai anti inflamasi atau anti radang.[2] Salah satu contoh tanaman obat dari daun yang sering digunakan adalah daun sirih untuk menghentikan mimisan.

Tanaman obat herbal dalam lingkungan masyarakat umum sering ditemui dengan bentuk-bentuk yang berbeda-beda. Namun, saat ini banyak orang yang menghadapi kesulitan dalam membedakan jenis-jenis tanaman obat tersebut. kemiripan bentuk dan tekstur antara tanaman obat herbal sering kali menimbulkan kesulitan dalam proses identifikasinya, terutama bagi masyarakat awam. Banyak tanaman yang secara visual menyerupai tanaman obat herbal, padahal sebenarnya bukan, sehingga berpotensi menimbulkan risiko kesehatan apabila dikonsumsi tanpa pengetahuan yang memadai. Beberapa di antaranya bahkan mengandung zat beracun yang dapat membahayakan kesehatan. Selain itu, pengamatan secara langsung memerlukan waktu yang lama dan tidak efisien, terutama bagi masyarakat yang tidak memiliki latar belakang pengetahuan tentang tanaman obat herbal. Untuk mengatasi masalah tersebut, pengembangan sistem yang dapat mengenali dan membedakan jenis tumbuhan obat berdasarkan ciri-ciri daun dapat menjadi solusi yang lebih efektif.[3]

Perkembangan teknologi, khususnya di bidang *AI (Artificial Intelligence)* telah membantu manusia memecahkan masalah yang lebih sulit untuk diselesaikan oleh program konvensional. CNN secara efektif digunakan dalam banyak aplikasi pengenalan pola dan gambar, seperti pengenalan gerakan, pengenalan wajah, klasifikasi objek, dan deskripsi gambar. Selain itu, CNN adalah salah satu metode *deep learning* karena menghasilkan

model yang lebih baik untuk pengenalan, segmentasi, deteksi, dan pengambilan gambar yang lebih baik. Selain itu, sejumlah penelitian sebelumnya juga telah dilakukan dalam upaya mengembangkan sistem identifikasi dan klasifikasi tanaman obat berbasis citra daun dengan memanfaatkan metode *Convolutional Neural Network* (CNN). Penelitian oleh Adiningsi dan Saputra [3] menggunakan arsitektur CNN *VGG16* untuk mengidentifikasi 10 jenis daun tanaman obat. Model ini mampu mencapai akurasi pengujian hingga 92%, menunjukkan bahwa CNN dapat secara efektif membedakan jenis-jenis daun tanaman obat berdasarkan fitur visualnya. Penelitian lain oleh Setiyono et al.[4] memanfaatkan CNN untuk mengklasifikasikan tanaman obat Indonesia dengan hasil akurasi rata-rata sebesar 98%, yang diperoleh dari dataset citra daun yang dikumpulkan dari Mendeley Data dan lingkungan masyarakat sekitar.

Meskipun hasil tersebut menunjukkan tingkat akurasi yang cukup tinggi, masih terdapat beberapa tantangan yang perlu diselesaikan. Tantangan tersebut meliputi keterbatasan jumlah dataset lokal yang representatif, perbedaan kualitas citra dari berbagai sumber, serta kebutuhan akan arsitektur model yang lebih efisien namun tetap memiliki akurasi tinggi. Oleh karena itu, dalam penelitian ini dipilih arsitektur *EfficientNetV2B0*, yang dikenal memiliki keunggulan dalam hal efisiensi parameter dan performa akurasi, sebagai solusi dalam pengembangan sistem klasifikasi tanaman obat berbasis citra. *EfficientNetV2B0* menunjukkan performa yang sangat stabil dan konsisten, dengan akurasi *validasi* dan pengujian tinggi pada berbagai konfigurasi *learning rate*. Kombinasi terbaik diperoleh pada *learning rate* 5E-04, dengan *val accuracy* sebesar 98.61% dan *test accuracy* sebesar 98.33%. Selain itu, arsitektur *EfficientNetV2B0* juga memiliki ukuran parameter yang lebih kecil dibanding *DenseNet121*, dengan waktu pelatihan yang lebih cepat, berkat penggunaan *fused-MBConv block* dan *progressive learning strategies* yang menjadi keunggulan utama dari seri *EfficientNetV2*.[5] Dengan mempertimbangkan hasil eksperimen yang menunjukkan akurasi tinggi, stabilitas *across hyperparameter settings*, serta efisiensi arsitektural yang sesuai untuk *deployment* pada perangkat *mobile*, *EfficientNetV2B0* dipilih sebagai model dasar dalam penelitian ini.

Dengan demikian, penelitian ini bertujuan untuk mengembangkan aplikasi *Mobile* yang dapat mengklasifikasikan tanaman obat herbal. Sehingga hasil penelitian ini akan memudahkan masyarakat awam dalam mengidentifikasi tanaman obat herbal yang berada disekitar mereka, sehingga masyarakat dapat mengetahui jenis-jenis tanaman herbal, beserta manfaat dan cara pengolahannya.

Spesifikasi Teknis

Spesifikasi teknis meliputi:

1. Modul Pengguna

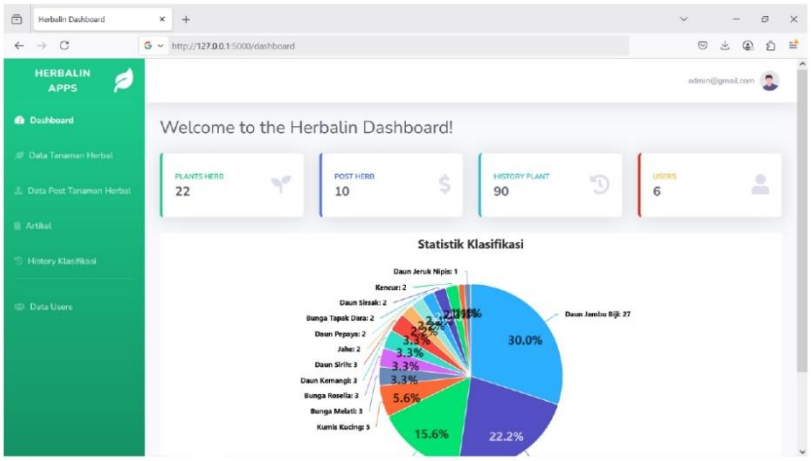
2. Source Code

Berikut uraian spesifikasi untuk pembangunan aplikasi:

1. Windows 10 Ram 8gb
2. Visual Studio Code
3. Android Studio
4. Postman
5. Xampp
6. Web Browser
7. Smartphone

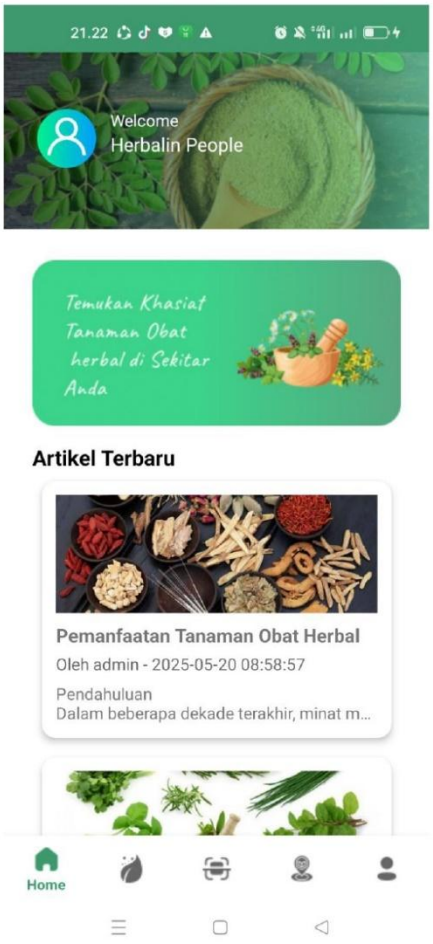
Berikut uraian spesifikasi modul:

1. **Modul untuk developer**
   Bagian ini hanya dapat diakses oleh developer, bertujuan untuk memudahkan melakukan perbaharuan isi konten data-data dari aplikasi berupa data daftar tanaman herbal, data artikel dan data user yang ada didalamnya.



2. **Modul untuk pengguna**

Bagian ini adalah aplikasi Herbalin Apps, bagian yang akan ditampilkan untuk pengguna berserta beragam fitur yang disediakan oleh aplikasi berupa Klasifikasi Tanaman Obat Herbal, Daftar Penyakit Ringan yang dapat diredakan dengan Tanaman Obat Herbal, Posting/Upload Tanaman Obat Herbal, Mencari Lokasi Titik-titik Tanaman, Artikel dan Profile.



Source Code

Source code back-end dari aplikasi Herbalin Apps mengguna framework Flask-Api yang digunakan sebagai route untuk melakukan login, register, edit pengguna, edit password dan menambahkan serta menampilkan artikel, back-end ini juga dijadikan sebagai website untuk menampilkan landing page pada Herbalin Apps serta halaman developer.

1. **Import Library**

```
from tensorflow import keras
import numpy as np
from datetime import datetime
import uuid
from flask_login import UserMixin
from flask_sqlalchemy import SQLAlchemy
import os
import secrets
from flask_login import LoginManager,
current_user,login_required, logout_user,login_user
from flask_mail import Mail, Message
from flask import Flask, flash, json, jsonify,
redirect, render_template, request, url_for
from flask_migrate import Migrate
from werkzeug.utils import secure_filename
from detections import predict_bunga, predict_label,
predict_rimpang
from model import Article, Category, History, Manfaat,
TanamanHerbal, db, Users, Herbal,Postherbal
from config import Config
from datetime import datetime, timedelta
from flask_jwt_extended import JWTManager,
create_access_token, jwt_required, get_jwt_identity
from argon2 import PasswordHasher
from PIL import Image
```

Script tersebut adalah sebuah program Python yang menggunakan berbagai pustaka dan kerangka kerja seperti Flask, Tensorflow, Flask-SQLAlchemy, dan Flask-Mail untuk membuat RestFull API dan Aplikasi Website. Berikut adalah penjelasan singkat mengenai bagian-bagian dari script tersebut:

1. from tensorflow import keras : Digunakan untuk mengakses modul keras dari TensorFlow yang berfungsi untuk memuat model deep learning
2. import numpy as np : Digunakan untuk mengimpor NumPy, pustaka Python untuk operasi matematika dan manipulasi array, sering digunakan untuk menyiapkan data pelatihan.
3. from datetime import datetime : Digunakan untuk mengakses waktu dan tanggal saat ini, misalnya untuk mencatat waktu pembuatan data.
4. import uuid: Digunakan untuk menghasilkan UUID (Unique Universal Identifier), biasanya untuk ID unik
5. from flask_login import UserMixin : Digunakan untuk menambahkan fungsionalitas autentikasi user ke model pengguna dalam Flask, seperti login/logout.
6. from flask_sqlalchemy import SQLAlchemy : Digunakan untuk menghubungkan dan mengelola database menggunakan SQLAlchemy di dalam aplikasi Flask.
7. import os : Untuk mengakses fungsi sistem operasi, seperti membaca path atau file.
8. import secrets : Untuk menghasilkan token acak yang aman, misalnya untuk reset password atau token verifikasi.
9. from flask_login import LoginManager, current_user,login_required, logout_user,login_user : Digunakan untuk mengelola autentikasi pengguna (login/logout) dalam aplikasi Flask.
10. from flask_mail import Mail, Message : Untuk mengirim email melalui aplikasi Flask.
11. from flask import Flask …. : Fungsi-fungsi utama dari Flask untuk membangun dan mengelola web (routing, form, tampilan, dll).
12. from flask_migrate import Migrate : Untuk mengelola migrasi database secara otomatis.
13. from werkzeug.utils import secure_filename : Untuk mengamankan nama file saat upload agar tidak mengandung karakter berbahaya.
14. from detections import predict_bunga, predict_label, predict_rimpang : Mengimpor fungsi prediksi dari modul detections, mungkin untuk klasifikasi gambar bunga, label, dan rimpang.
15. from model import Article, Category, History, Manfaat, TanamanHerbal, db, Users, Herbal,Postherbal : Mengimpor model database dan objek penting dari file model.py.
16. from config import Config : Mengimpor konfigurasi aplikasi (biasanya berisi pengaturan database, JWT, email, dll). :
17. from flask_jwt_extended import ….. : Untuk mengelola autentikasi menggunakan JWT (JSON Web Token).
18. from argon2 import PasswordHasher : Untuk mengenkripsi dan memverifikasi password menggunakan algoritma Argon2 (aman dan modern).
19. from PIL import Image : Untuk memproses dan memanipulasi gambar (resize, convert, dsb.).

Setelah mengimpor semua modul yang diperlukan, script kemudian mendefinisikan dan mengkonfigurasi aplikasi Flask beserta berbagai fitur seperti SQLAlchemy untuk koneksi database, Flask-Mail untuk pengiriman email, serta integrasi fungsi klasifikasi citra menggunakan Flask. Script ini kemungkinan besar juga akan berisi definisi rute-rute (routes) yang menangani permintaan dari klien, termasuk operasi seperti otentikasi, manipulasi data dalam database, pengiriman email, dan pemrosesan klasifikasi gambar yang dikirimkan oleh pengguna.

2. Konfigurasi Database

```
SQLALCHEMY_DATABASE_URI =
'mysql+mysqlconnector://root:@localhost/db_herbalin'
SQLALCHEMY_TRACK_MODIFICATIONS = False
SECRET_KEY = 'your_secret_key'
```

1. Mengatur URI koneksi ke database MySQL bernama db_herbalin dan Menggunakan user root tanpa password.
2. Menonaktifkan fitur pelacakan perubahan objek oleh SQLAlchemy (menghemat memori dan performa).
3. Kunci rahasia untuk menjaga keamanan aplikasi Flask (misalnya untuk session atau CSRF protection).

```
MAIL_SERVER ='smtp.gmail.com'
MAIL_PORT = 587
MAIL_USE_TLS = True
MAIL_USERNAME = "maulana29.rizqi@gmail.com"
MAIL_PASSWORD = "wtblqfxrlnjijnhr"
MAIL_DEFAULT_SENDER = ('Herbalin Apps',
'emailkamu@gmail.com')
```

1. Mengatur server email Gmail agar dapat mengirim email dari aplikasi.
2. Menggunakan TLS (Transport Layer Security).
3. Menentukan email dan password pengirim (harus pakai App Password dari Gmail jika 2FA aktif).
4. Menentukan nama dan alamat pengirim default.

```
app.config.from_object(Config)
db.init_app(app)
migrate = Migrate(app, db)
mail = Mail(app)
```

1. Memasukkan semua konfigurasi dari class Config ke dalam objek app

2. Menginisialisasi koneksi database SQLAlchemy dengan aplikasi Flask
3. Mengaktifkan fitur migrasi database dengan Flask-Migrate
4. Menginisialisasi Flask-Mail untuk mengirim email dari aplikasi

Setelah konfigurasi, objek Flask-Mail (mail) juga dibuat dengan menggunakan konfigurasi aplikasi (app), dan objek SQLAlchemy (db) juga dibuat dengan menggunakan konfigurasi aplikasi (app). Dengan konfigurasi ini, aplikasi Flask telah siap digunakan dengan koneksi database MySQL dan pengiriman email yang terkonfigurasi.

3. Load Model .keras

```
from tensorflow import keras
import numpy as np

# Load model
model =
keras.models.load_model("model/FinalModelEfficiennetv2b
0.keras")


label = ['Bunga Kitolod', 'Bunga Melati', 'Bunga
Rosella', 'Bunga Tapak Dara', 'Bunga Telang', 'Daun
Jambu Biji', 'Daun Jeruk Nipis', 'Daun Kari', 'Daun
Kelor', 'Daun Kemangi', 'Daun Kunyit', 'Daun Mint',
'Daun Pandan', 'Daun Pepaya', 'Daun Sirih', 'Daun
Sirsak', 'Jahe', 'Kencur', 'Kumis Kucing', 'Kunyit',
'Lengkuas', 'Lidah Buaya', 'Temu Kunci', 'Temulawak']
```

1. Mengimpor modul keras dari TensorFlow dan numpy untuk manipulasi data numerik.
2. Memuat tiga model deep learning berformat .keras untuk klasifikasi citra.
3. Membuat daftar nama kelas (label) yang berisi nama-nama jenis tanaman herbal. Daftar ini digunakan untuk mengonversi output numerik model (berupa indeks atau array prediksi) menjadi informasi yang bermakna bagi user berupa nama kelas prediksi.

```
def predict_label(img):
i = np.asarray(img) / 255.0
i = i.reshape(1, 384, 384, 3)
pred = model.predict(i)
result_label = label[np.argmax(pred)]
result_accuracy = np.max(pred) * 100  # Convert to
percentage
```

```
return result_label, result_accuracy

def predict_bunga(img):
i = np.asarray(img) / 255.0
i = i.reshape(1, 384, 384, 3)
pred = model_bunga.predict(i)
result_label = labelbunga[np.argmax(pred)]
result_accuracy = np.max(pred) * 100  # Convert to
percentage
return result_label, result_accuracy

def predict_rimpang(img):
i = np.asarray(img) / 255.0
i = i.reshape(1, 384, 384, 3)
pred = model_rimpang.predict(i)
result_label = labelrimpang[np.argmax(pred)]
result_accuracy = np.max(pred) * 100  # Convert to
percentage
return result_label, result_accuracy
```

Fungsi ini digunakan untuk memprediksi label (daun, bunga dan rimpang) dari sebuah gambar.

Penjelasan Scripts tersebut:

1. img diubah menjadi array dan dinormalisasi dengan membagi 255 agar berada dalam rentang 0–1.
2. Ukuran gambar diubah menjadi (1, 384, 384, 3) agar cocok sebagai input model.
3. model.predict(i) digunakan untuk memprediksi gambar tersebut.
4. np.argmax(pred) mencari indeks dengan nilai tertinggi (label yang diprediksi).
5. np.max(pred) * 100 menghitung akurasi (probabilitas tertinggi dikalikan 100).
6. Fungsi mengembalikan label hasil prediksi dan persentase akurasinya.

4. Route Register

```
@app.route('/register', methods=['POST'])
def register_user():
    data = request.form
    username = data['username']
    email = data['email']
    password = data['password']

    if not username:
```

```
        return jsonify(message='Username harus
diisi'),400
    if not email:
        return jsonify(message='Email harus diisi'),400
    if not password:
        return jsonify(message='Password harus diisi')

    if Users.query.filter_by(email=email).first():
        return jsonify({'error': True, 'message':
'email already exists'}), 400

    password_hash = ph.hash(password)
    verification_token = secrets.token_urlsafe(32)

    new_user = Users(id=str(uuid.uuid4()),
username=username, email=email,
password=password_hash,verification_token=verification_
token)

    db.session.add(new_user)
    db.session.commit()

    confirmation_url = url_for('confirm_email',
token=verification_token, _external=True)
    msg = Message(subject="Verify your email",
sender="noreply@app.com", recipients=[email])
    msg.html = render_template("verify-email.html",
confirmation_url=confirmation_url)

    mail.send(msg)

    return jsonify({'message': 'User registered
successfully'})
```

Ketika pengguna mengirim permintaan POST ke endpoint /register, fungsi register_user() akan dijalankan. Fungsi ini pertama-tama mengambil data formulir yang dikirimkan oleh pengguna, yaitu username, email, dan password. Selanjutnya, sistem melakukan validasi untuk memastikan bahwa ketiga data tersebut tidak kosong. Jika salah satu data tidak diisi, maka

sistem akan memberikan respons berupa pesan kesalahan yang menjelaskan data mana yang harus diisi, dengan status HTTP 400 (Bad Request).

Setelah itu, sistem memeriksa apakah email yang digunakan sudah terdaftar dalam database. Jika email sudah ada, maka sistem akan menghentikan proses registrasi dan mengembalikan respons dengan pesan bahwa email tersebut telah digunakan.
Jika email belum terdaftar, maka sistem akan melanjutkan proses dengan meng-hash password menggunakan metode keamanan tertentu agar tidak tersimpan dalam bentuk asli di database. Selanjutnya, sistem membuat token verifikasi secara acak menggunakan fungsi secrets.token_urlsafe(). Token ini nantinya akan digunakan untuk proses verifikasi email.
Kemudian, sistem membuat objek pengguna baru yang berisi ID unik (menggunakan UUID), username, email, password yang telah di-hash, dan token verifikasi. Objek ini ditambahkan ke dalam sesi database dan disimpan secara permanen dengan commit().
Setelah data pengguna berhasil disimpan, sistem membentuk URL konfirmasi berdasarkan token yang telah dibuat dan menggunakannya dalam email verifikasi. Email ini dikirim ke alamat email yang telah didaftarkan oleh pengguna dengan subjek "Verify your email" dan isi email berasal dari template HTML yang telah disiapkan.
Terakhir, jika seluruh proses berjalan dengan sukses, sistem akan mengembalikan respons dalam format JSON yang menyatakan bahwa pengguna telah berhasil terdaftar.

5. **Route Konfirmasi Email**

```
@app.route('/confirm_email/<token>')
def confirm_email(token):
    user =
Users.query.filter_by(verification_token=token).first_o
r_404()
    if user.is_verified:
        return jsonify(message="Account already
verified."), 200
    user.is_verified = True
    user.verification_token = None
    user.updated_at = datetime.utcnow()
    db.session.commit()
    return render_template('verify-success.html',
message="Email berhasil diverifikasi"), 200

def send_email(to, subject, body):
```

```
    msg = Message(subject, recipients=[to], body=body)
    mail.send(msg)
```

Fungsi confirm_email(token) dijalankan ketika pengguna mengakses tautan verifikasi email yang dikirim oleh sistem, dengan menyertakan token unik sebagai bagian dari URL. Token ini digunakan untuk mencari pengguna yang sesuai di dalam database berdasarkan nilai verification_token. Jika token tersebut tidak ditemukan, maka sistem akan secara otomatis menampilkan halaman error 404.

Setelah pengguna ditemukan, sistem akan memeriksa apakah akun pengguna tersebut sudah diverifikasi sebelumnya. Jika akun sudah terverifikasi, maka sistem akan memberikan respons berupa pesan bahwa akun telah diverifikasi, dan permintaan dianggap berhasil dengan status kode 200.

Namun, jika akun belum diverifikasi, maka sistem akan mengubah nilai is_verified menjadi True, menghapus token verifikasi agar tidak bisa digunakan ulang, serta memperbarui waktu pembaruan data dengan waktu saat ini menggunakan datetime.utcnow(). Seluruh perubahan ini kemudian disimpan ke dalam database dengan perintah db.session.commit(). Setelah itu, sistem akan menampilkan halaman HTML verify-success.html yang berisi pesan bahwa verifikasi email berhasil dilakukan.

Sedangkan fungsi send_email(to, subject, body) berfungsi untuk mengirimkan email ke alamat yang ditentukan. Fungsi ini akan membuat objek pesan email berdasarkan alamat tujuan (to), subjek (subject), dan isi pesan (body) yang diberikan. Selanjutnya, email tersebut akan dikirim melalui sistem pengiriman email yang telah dikonfigurasi dengan memanggil mail.send(msg).

6. **Route Login**

```
@app.route('/login', methods=['POST'])
def login_android():
    data = request.form
    email = data['email']
    password = data['password']

    user = Users.query.filter_by(email=email).first()

    # Jika pengguna tidak ditemukan
    if not user:
        return jsonify({'error': True, 'message': 'Akun
belum terdaftar. Silakan lakukan pendaftaran terlebih
dahulu!'}), 404
```

```
    if user:
        if not user.is_verified:
            return jsonify({'error': True, 'message':
'Akun belum diverifikasi. Silakan cek email Anda!'}),
403

        try:
            if ph.verify(user.password, password):
                access_token =
create_access_token(identity=user.id)
                return jsonify({'token':
access_token}), 200
        except:
            return jsonify({'error': True, 'message':
'Kredensial salah'}), 401

    return jsonify({'error': True, 'message': 'Invalid
credentials'}), 401
```

Fungsi login_android() digunakan untuk menangani proses login pengguna melalui metode POST. Ketika pengguna mengirimkan permintaan login, sistem akan menerima data berupa email dan password melalui request.form.

Pertama, sistem akan mencari pengguna dalam database berdasarkan alamat email yang dikirimkan. Jika pengguna dengan email tersebut tidak ditemukan, maka sistem akan memberikan respons berupa pesan bahwa akun belum terdaftar dan menyarankan pengguna untuk melakukan pendaftaran terlebih dahulu. Respons ini dikembalikan dengan status kode HTTP 404.

Jika pengguna ditemukan, sistem akan memeriksa apakah akun tersebut sudah diverifikasi. Jika akun belum diverifikasi, maka sistem akan memberikan respons dengan status 403 dan pesan yang menyarankan pengguna untuk memeriksa email mereka terlebih dahulu untuk melakukan verifikasi.

Jika akun telah diverifikasi, sistem akan melanjutkan proses dengan memverifikasi kecocokan antara password yang dikirimkan oleh pengguna dan password yang tersimpan di database. Verifikasi dilakukan menggunakan fungsi ph.verify() untuk mencocokkan password yang telah di-hash. Jika password cocok, maka sistem akan membuat token akses (access token) menggunakan create_access_token() dengan identitas berupa ID pengguna. Token ini

kemudian dikembalikan kepada pengguna sebagai tanda berhasil login, dengan status HTTP 200.

Namun, jika terjadi kesalahan dalam proses verifikasi password—misalnya karena password salah—maka sistem akan memberikan respons dengan pesan bahwa kredensial salah dan status HTTP 401 (unauthorized). Jika semua kondisi gagal dipenuhi, sistem akan mengembalikan respons umum bahwa kredensial tidak valid.
Dengan demikian, fungsi ini memastikan bahwa hanya pengguna yang sudah terdaftar dan telah melakukan verifikasi email yang dapat login ke dalam sistem, serta menjaga keamanan dengan menggunakan verifikasi password terenkripsi dan token autentikasi.

7. **Route My Profile**

```
@app.route('/my_profile', methods=['GET'])
@jwt_required()
def my_profile():
    user_id = get_jwt_identity()
    user = Users.query.get(user_id)

    if user:
        return jsonify({
            'id': user.id,
            'username': user.username,
            'email': user.email,
            'avatar': user.avatar
        }), 200
    else:
        return jsonify({'error': True, 'message': 'User
not found'}), 404
```

Fungsi my_profile() digunakan untuk mengambil data profil pengguna yang sedang login. Fungsi ini hanya bisa diakses oleh pengguna yang telah terautentikasi karena dilindungi oleh dekorator @jwt_required(), yang berarti pengguna harus menyertakan token JWT yang valid saat mengakses endpoint ini.
Ketika fungsi dipanggil, sistem terlebih dahulu mengekstrak user_id dari token JWT menggunakan get_jwt_identity(). Nilai user_id ini digunakan untuk mencari data pengguna pada tabel Users melalui Users.query.get(user_id).

Jika data pengguna berhasil ditemukan, maka sistem akan mengembalikan informasi profil pengguna dalam bentuk JSON, yang mencakup id, username, email, dan avatar. Data ini dikirim bersama status kode HTTP 200 yang menandakan bahwa permintaan berhasil.

Namun, jika pengguna tidak ditemukan di dalam database—misalnya karena ID tidak valid atau akun sudah dihapus—maka sistem akan memberikan respons berupa pesan "User not found" dan status HTTP 404.

8. **Route Edit Profile**

```
@app.route('/edit_profile', methods=['PUT'])
@jwt_required()
def edit_profile():
    current_user_id = get_jwt_identity()

    user = db.session.get(Users, current_user_id)

    if not user:
        return jsonify({"message": "User not found"}),
404

    data = request.form if request.form else
request.json

    if 'username' in data:
        user.username = data['username']

    # if 'email' in data:
    #     user.email = data['email']

    if 'file' in request.files:
        file = request.files['file']
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file_path =
os.path.join(app.config['UPLOAD_FOLDER'], filename)
            file.save(file_path)
            user.avatar = url_for('static',
filename='uploads/' + filename, _external=True)
```

```
    # user.updated_at = datetime.now()
    db.session.commit()
    return jsonify({
        "message": "Profile updated successfully",
    }), 200
```

Fungsi edit_profile() bertugas untuk mengedit atau memperbarui data profil pengguna yang sedang login. Akses ke fungsi ini dilindungi oleh dekorator @jwt_required(), yang memastikan hanya pengguna dengan token JWT yang valid yang dapat menggunakan fitur ini.

Saat fungsi dijalankan, sistem akan mengambil ID pengguna yang sedang login dari token JWT menggunakan get_jwt_identity(). Kemudian, sistem mencoba mengambil data pengguna dari database berdasarkan ID tersebut menggunakan db.session.get().

Jika pengguna tidak ditemukan di dalam database, maka sistem akan membalas dengan pesan "User not found" dan status HTTP 404.

Jika pengguna ditemukan, sistem akan membaca data yang dikirimkan oleh pengguna. Data ini bisa dikirim dalam bentuk form (jika ada file yang diunggah) atau dalam format JSON. Sistem kemudian memeriksa apakah terdapat data username baru, dan jika ada, maka nilai username pada objek pengguna akan diperbarui.

Untuk pengunggahan gambar avatar, sistem memeriksa apakah ada file yang dikirim dengan nama field 'file'. Jika ada file dan ekstensi file tersebut diperbolehkan (sesuai fungsi allowed_file()), maka file tersebut akan disimpan ke direktori yang telah ditentukan di konfigurasi aplikasi. Nama file akan diamankan menggunakan secure_filename() untuk mencegah injeksi nama file yang berbahaya. Setelah file disimpan, URL dari file tersebut akan disimpan ke properti avatar milik pengguna, menggunakan fungsi url_for() agar dapat diakses secara eksternal.

Setelah semua perubahan selesai, sistem akan menyimpan perubahan tersebut ke database menggunakan db.session.commit(). Akhirnya, sistem akan mengirimkan respons bahwa profil berhasil diperbarui, dengan status kode HTTP 200.

Dengan demikian, fungsi ini memberikan kemampuan bagi pengguna untuk memperbarui nama pengguna dan foto profil mereka secara aman dan efisien, dengan validasi login melalui JWT.

9. **Route Edit Password**

```
@app.route('/change_password', methods=['PUT'])
@jwt_required()
def change_password():
```

```
    user_id = get_jwt_identity()
    data = request.form

    old_password = data.get('old_password')
    new_password = data.get('new_password')

    if not old_password or not new_password:
        return jsonify({'error': True, 'message':
'Password lama dan baru harus diisi'}), 400

    user = Users.query.filter_by(id=user_id).first()

    if not user:
        return jsonify({'error': True, 'message':
'Pengguna tidak ditemukan'}), 404

    try:
        # Verifikasi password lama
        if not ph.verify(user.password, old_password):
            return jsonify({'error': True, 'message':
'Password lama salah'}), 401
    except:
        return jsonify({'error': True, 'message':
'Password lama salah'}), 401

    # Hash password baru dan update
    user.password = ph.hash(new_password)
    user.updated_at = datetime.utcnow()
    db.session.commit()

    return jsonify({'message': 'Password berhasil
diubah'}), 200
```

Fungsi change_password() digunakan untuk mengubah kata sandi pengguna yang sedang login. Akses ke fungsi ini dibatasi menggunakan dekorator @jwt_required(), yang memastikan hanya pengguna dengan token autentikasi JWT yang valid yang dapat menjalankan fungsi ini.

Ketika fungsi dipanggil, sistem akan mengambil ID pengguna dari token JWT melalui get_jwt_identity(), kemudian data permintaan diambil dari request.form. Dari data tersebut,

sistem mengekstrak dua input penting, yaitu old_password (kata sandi lama) dan new_password (kata sandi baru).

Jika salah satu dari input tersebut tidak disediakan oleh pengguna, sistem akan langsung membalas dengan pesan bahwa kedua kolom harus diisi, dan mengembalikan kode status HTTP 400.

Selanjutnya, sistem mencoba mengambil data pengguna dari database berdasarkan ID yang telah diperoleh. Jika data pengguna tidak ditemukan, maka respons berupa pesan "Pengguna tidak ditemukan" dikembalikan dengan kode status 404.

Jika pengguna ditemukan, sistem akan melakukan verifikasi terhadap kata sandi lama menggunakan ph.verify() (fungsi dari library hashing password seperti Argon2 atau bcrypt). Jika verifikasi gagal, artinya kata sandi lama yang dimasukkan salah, maka sistem akan membalas dengan pesan kesalahan dan status HTTP 401.

Namun jika verifikasi berhasil, maka sistem akan mengenkripsi (hash) kata sandi baru menggunakan metode ph.hash() dan memperbarui nilai kata sandi pada akun pengguna di database. Waktu pembaruan juga dicatat dengan datetime.utcnow(). Setelah itu, sistem menyimpan perubahan menggunakan db.session.commit().

Terakhir, sistem akan memberikan respons bahwa kata sandi berhasil diubah dengan kode status HTTP 200.

10. Route Klasifikasi Citra

```
@app.route("/predict", methods=["POST"])
@jwt_required()  # Memastikan pengguna terautentikasi
sebelum klasifikasi
def klasifikasi_daun():
# Ambil ID user dari token JWT
    user_id = get_jwt_identity()

    # Ambil file gambar dari form
    file = request.files.get('file')
    if file is None or file.filename == "":
        return jsonify({"error": "No file provided"}),
400

    # Validasi format file
    if not allowed_file(file.filename):
```

```
        return jsonify({"error": "Format file tidak
diizinkan"}), 400

    try:
        # Simpan gambar ke direktori uploads
        filename = secure_filename(file.filename)  #
Pastikan nama file aman
        file_path =
os.path.join(app.config['UPLOAD_FOLDER'], filename)  #
Path file
        file.save(file_path)  # Simpan file ke
direktori

        # Generate URL gambar menggunakan url_for
        image_url = url_for('static',
filename=f'uploads/{filename}', _external=True)

        # Buka gambar dengan Pillow untuk prediksi
        img = Image.open(file_path)
        img = img.resize((384, 384), Image.NEAREST)

        # Lakukan prediksi menggunakan model
        pred_label, pred_accuracy = predict_label(img)

        # Jika akurasi kurang dari 75%, berikan respons
error dan hentikan proses
        if pred_accuracy < 80:
            return jsonify({
                "message": "Maaf bukan daun, Silahkan
masukkan gambar lain",
            }), 400

        # Ambil tanaman berdasarkan nama (pred_label)
dari database
        tanaman =
TanamanHerbal.query.filter_by(nama=pred_label).first()

        if tanaman is None:
```

```
            return jsonify({"error": "Tanaman tidak
ditemukan dalam database"}), 404

        # Ambil manfaat dan cara pengolahan dari
tanaman yang ditemukan
        manfaat_list = [{"nama_manfaat":
m.nama_manfaat, "cara_pengolahan": m.cara_pengolahan}
for m in tanaman.manfaat]

        # Simpan hasil klasifikasi ke dalam database
(History)
        new_history = History(
            user_id=user_id,
            pred_label=pred_label,
            pred_accuracy=float(pred_accuracy),
            image_url=image_url  # Simpan URL gambar ke
database
        )
        db.session.add(new_history)
        db.session.commit()

        # Kembalikan hasil klasifikasi
        return jsonify({
            "label": pred_label,
            "accuracy": f"{pred_accuracy:.2f}%",
            "image_url": image_url,
            "manfaat": manfaat_list
        }), 200

    except Exception as e:
        print(f"Error: {e}")
        return jsonify({"error": "Terjadi kesalahan
saat klasifikasi"}), 500
```

Kode di atas merupakan bagian dari backend aplikasi Flask yang bertanggung jawab untuk melakukan klasifikasi gambar daun menggunakan metode deep learning, serta menyimpan hasil klasifikasinya ke dalam database sebagai riwayat (history). Endpoint ini hanya bisa

diakses oleh pengguna yang sudah login, karena terdapat dekorator @jwt_required() yang memastikan token JWT valid dan pengguna terautentikasi.

Ketika endpoint /predict menerima request POST, langkah pertama yang dilakukan adalah mengambil ID pengguna dari token JWT dengan fungsi get_jwt_identity(). Kemudian, backend mencoba mengambil file gambar yang dikirimkan melalui form. Jika file tidak ada atau nama file kosong, maka akan dikembalikan response error 400.

Selanjutnya, file gambar akan divalidasi formatnya melalui fungsi allowed_file(). Jika format tidak sesuai (misalnya bukan jpg atau png), maka response error akan dikirim kembali.

Jika validasi berhasil, gambar akan disimpan di direktori upload menggunakan secure_filename() untuk memastikan nama file aman, dan path lengkapnya akan disusun. File kemudian disimpan ke path tersebut, dan URL gambar dibuat menggunakan url_for() agar bisa diakses secara publik.

Setelah gambar disimpan, gambar dibuka menggunakan library Pillow (Image.open) dan diubah ukurannya menjadi 384x384 piksel untuk menyesuaikan dengan input model prediksi. Kemudian, fungsi predict_label() dipanggil untuk melakukan prediksi label daun dan mendapatkan nilai akurasinya.

Jika akurasi hasil prediksi kurang dari 80%, sistem akan menolak hasil klasifikasi dan mengembalikan pesan bahwa gambar tersebut kemungkinan bukan daun yang valid. Namun, jika akurasi cukup tinggi, sistem akan mencoba mencari tanaman herbal yang sesuai di database berdasarkan pred_label. Jika tidak ditemukan, maka response error 404 akan dikembalikan.

Jika tanaman ditemukan, sistem mengambil semua data manfaat dan cara pengolahan dari tanaman tersebut dan menyusunnya dalam bentuk list of dictionary. Hasil klasifikasi ini kemudian dicatat ke dalam tabel History yang berisi informasi pengguna, label hasil prediksi, akurasi, dan URL gambar.

Terakhir, data hasil klasifikasi (label, akurasi, URL gambar, dan daftar manfaat serta cara pengolahan) dikembalikan ke pengguna dalam bentuk JSON dengan status sukses 200. Jika terjadi error dalam proses apa pun, sistem akan menangkap exception dan mengembalikan response error 500

11. Route Posting Tanaman

```
@app.route('/post_herbal', methods=['POST'])
@jwt_required()
def post_herbal():
    user_id = get_jwt_identity()

    # Ambil data dari form
    name = request.form.get('name')
```

```
    description = request.form.get('description')
    lat = request.form.get('lat')
    lon = request.form.get('lon')
    no_wa = request.form.get('no_wa')
    avatar = request.files.get('avatar')  # Ambil file
avatar

    # Validasi input
    if not name or not description:
        return jsonify({'error': True, 'message':
'Semua field harus diisi'}), 400

    # Validasi file gambar avatar
    if avatar and allowed_file(avatar.filename):
        filename = secure_filename(avatar.filename)
        file_path =
os.path.join(app.config['UPLOAD_FOLDER'], filename)  #
Path lengkap file
        avatar.save(file_path)  # Simpan file gambar

        # Simpan path URL gambar relatif di database
        avatar_url = url_for('static',
filename=f'uploads/{filename}', _external=True)
    else:
        avatar_url = None  # Jika avatar tidak ada atau
tidak valid

    # Buat entri postherbal baru
    new_herbal = Postherbal(
        user_id=user_id,
        name=name,
        description=description,
        no_wa=no_wa,
        lat=lat,
        lon=lon,
        avatar=avatar_url  # Simpan URL avatar
    )
```

```
    # Simpan ke database
    db.session.add(new_herbal)
    db.session.commit()

    return jsonify({
        'message': 'Herbal berhasil diposting',
        'herbal': new_herbal.serialize()
    }), 201
```

Kode tersebut merupakan bagian dari API backend yang digunakan untuk memposting data tanaman herbal ke sistem. Endpoint /post_herbal hanya bisa diakses oleh pengguna yang sudah login atau memiliki token JWT yang valid, karena dilindungi oleh dekorator @jwt_required(). Ketika pengguna mengirimkan permintaan POST ke endpoint ini, sistem pertama-tama mengambil ID pengguna dari token JWT yang aktif menggunakan fungsi get_jwt_identity(). Kemudian, data tanaman herbal dikumpulkan dari form yang dikirim oleh pengguna, termasuk name (nama tanaman), description (deskripsi), lat dan lon (koordinat lokasi), no_wa (nomor WhatsApp), serta file gambar avatar (jika ada).

Selanjutnya dilakukan validasi sederhana. Jika name atau description kosong, maka sistem akan langsung merespons dengan pesan error dan status 400 yang menunjukkan bahwa semua field wajib diisi.

Jika pengguna juga mengunggah gambar sebagai avatar dan format filenya valid (dicek menggunakan fungsi allowed_file()), maka gambar tersebut akan disimpan ke direktori upload yang telah ditentukan. Nama file akan diamankan dengan secure_filename() untuk mencegah nama file yang tidak aman. Setelah disimpan, URL gambar akan dibuat menggunakan fungsi url_for() agar nantinya bisa diakses dari luar.

Jika tidak ada file gambar atau file tidak valid, maka URL avatar akan diset ke None.

Setelah semua data siap, sistem membuat objek Postherbal baru yang berisi semua data termasuk ID pengguna, nama, deskripsi, nomor WhatsApp, koordinat lokasi, dan URL gambar (jika ada). Objek ini kemudian ditambahkan ke sesi database dan disimpan secara permanen menggunakan db.session.commit().

Terakhir, sistem merespons dengan status 201 Created dan mengembalikan pesan sukses serta data tanaman herbal yang baru diposting dalam bentuk JSON melalui pemanggilan fungsi serialize() pada objek new_herbal. Fungsi serialize() bertugas untuk mengubah data model ke format dictionary agar bisa dikirim dalam respons JSON.

12. Route Artikel

```
@app.route('/api/articles', methods=['GET'])
def get_articles():
    articles = Article.query.all()
    result = []
    for article in articles:
        result.append({
            'id': article.id,
            'judul': article.title,
            'konten': article.content,
            'image_url': article.image_url,
            'tanggal_post':
article.created_at.strftime('%Y-%m-%d %H:%M:%S'),
            'author': article.author.username
        })
    return jsonify({'data': result}), 200
```

Kode tersebut merupakan endpoint API yang berfungsi untuk mengambil semua artikel yang tersedia dalam database dan mengembalikannya dalam format JSON. Endpoint ini dapat diakses melalui URL /api/articles dengan metode GET, tanpa memerlukan autentikasi, sehingga bisa diakses secara publik.

Pertama, kode melakukan query ke database untuk mengambil semua data artikel menggunakan Article.query.all(), yang akan mengembalikan list berisi semua objek artikel dari tabel Article. Selanjutnya, sistem menyiapkan list kosong bernama result untuk menyimpan data artikel yang akan dikembalikan dalam format yang lebih mudah dibaca (terstruktur).

Setelah semua artikel diproses, data dalam result dikembalikan ke pengguna dalam bentuk JSON dengan kunci data, dan status HTTP 200 OK sebagai tanda bahwa permintaan berhasil. Endpoint ini sangat berguna, misalnya, untuk menampilkan daftar artikel pada aplikasi frontend atau mobile.

13. Route Tanaman Herbal

```
@app.route("/api/tanaman", methods=["GET"])
def api_tanaman():
    tanaman_list = TanamanHerbal.query.all()
    result = []
    for tanaman in tanaman_list:
        manfaat_list = [
            {
```

```
                "id": m.id,
                "nama_manfaat": m.nama_manfaat,
                "cara_pengolahan": m.cara_pengolahan
            }
            for m in tanaman.manfaat
        ]
        result.append({
            "id": tanaman.id,
            "nama": tanaman.nama,
            "manfaat": manfaat_list
        })
    return jsonify(result)
```

Kode ini merupakan endpoint API yang berfungsi untuk mengambil seluruh data tanaman herbal beserta manfaat dan cara pengolahannya dari database. Endpoint ini dapat diakses melalui URL /api/tanaman dengan metode GET, dan bisa digunakan tanpa autentikasi (terbuka untuk umum).

Saat endpoint diakses, sistem pertama-tama melakukan query ke database untuk mengambil seluruh data tanaman herbal melalui TanamanHerbal.query.all(). Hasilnya disimpan dalam variabel tanaman_list berupa list objek TanamanHerbal.

Selanjutnya, disiapkan list kosong result yang akan digunakan untuk menampung data yang akan dikembalikan ke klien.

Untuk setiap objek tanaman dalam tanaman_list, sistem akan melakukan iterasi dan mengakses relasi ke tabel manfaat, yaitu daftar manfaat dari tanaman tersebut. Data setiap manfaat akan diubah menjadi dictionary yang berisi:

- id: ID unik manfaat.
- nama_manfaat: Nama manfaat.
- cara_pengolahan: Penjelasan tentang bagaimana cara mengolah tanaman untuk memperoleh manfaat tersebut.

Kumpulan dictionary manfaat tersebut kemudian dimasukkan ke dalam list manfaat_list.

Setelah itu, setiap tanaman akan diubah menjadi dictionary dengan struktur:

- id: ID tanaman.
- nama: Nama tanaman.
- manfaat: List manfaat yang sudah diproses sebelumnya.

Data tanaman yang sudah diproses kemudian dimasukkan ke dalam list result.

Terakhir, sistem akan mengembalikan result dalam format JSON sebagai respons dari API. Output ini sangat berguna untuk frontend atau aplikasi mobile yang ingin menampilkan daftar tanaman herbal lengkap beserta manfaat dan cara pengolahannya.

14. Route History Klasifikasi

```
@app.route('/history', methods=['GET'])
@jwt_required()
def history():
    # Ambil user_id dari token JWT
    current_user_id = get_jwt_identity()

    herbals =
History.query.filter_by(user_id=current_user_id).all()
    if not herbals:
        return jsonify([]), 200
    # Format data untuk JSON
    herbal_list = [{
        "id": herbal.id,
        "user_id": herbal.user_id,
        "pred_label": herbal.pred_label,
        "pred_accuracy": herbal.pred_accuracy,
        "image_url": herbal.image_url,
        "created_at": herbal.created_at
    } for herbal in herbals]

    return jsonify(herbal_list), 200
```

Kode ini mendefinisikan sebuah endpoint API dengan URL /history dan metode GET, yang digunakan untuk mengambil riwayat klasifikasi daun milik pengguna yang sedang login. Endpoint ini dilindungi dengan autentikasi JWT (@jwt_required()), artinya hanya pengguna yang telah login dan memiliki token JWT yang valid yang bisa mengaksesnya.

Saat endpoint diakses, sistem pertama-tama mengambil user_id dari token JWT yang sedang aktif menggunakan get_jwt_identity(). ID ini digunakan untuk menyaring data riwayat klasifikasi yang hanya milik pengguna tersebut.

Kemudian dilakukan query ke tabel History menggunakan filter_by(user_id=current_user_id), agar hanya mengambil data yang sesuai dengan user_id tersebut. Hasilnya disimpan dalam variabel herbals.

Jika tidak ditemukan data riwayat apapun (artinya herbals kosong), sistem akan langsung mengembalikan array kosong [] dalam format JSON dan status 200 OK, yang menandakan permintaan berhasil meskipun tidak ada data.

Namun, jika ada data yang ditemukan, maka sistem akan memformat setiap entri herbal ke dalam bentuk dictionary yang berisi:

- id: ID riwayat klasifikasi.
- user_id: ID pengguna yang melakukan klasifikasi.
- pred_label: Hasil label klasifikasi tanaman.
- pred_accuracy: Akurasi dari hasil klasifikasi tersebut.
- image_url: URL gambar daun yang diklasifikasikan.
- created_at: Waktu ketika klasifikasi tersebut dilakukan.

Semua data riwayat tersebut dimasukkan ke dalam list herbal_list, dan kemudian dikembalikan ke klien dalam bentuk JSON dengan status HTTP 200 OK.

15. Route My Profile

```
@app.route('/my_profile', methods=['GET'])
@jwt_required()
def my_profile():
    user_id = get_jwt_identity()
    user = Users.query.get(user_id)
 if user:
        return jsonify({
            'id': user.id,
            'username': user.username,
            'email': user.email,
            'avatar': user.avatar
        }), 200
    else:
        return jsonify({'error': True, 'message': 'User
not found'}), 404
```

Fungsi my_profile() merupakan sebuah endpoint API yang merespons permintaan HTTP GET ke URL /my_profile. Endpoint ini dilindungi oleh dekorator @jwt_required(), yang artinya hanya pengguna yang telah login dan memiliki token JWT yang valid yang dapat mengaksesnya.

Ketika endpoint ini dipanggil, sistem akan mengambil user_id dari token JWT yang sedang aktif menggunakan fungsi get_jwt_identity(). ID ini merepresentasikan identitas pengguna yang sedang login.

Setelah mendapatkan user_id, sistem mencari data pengguna dari database menggunakan Users.query.get(user_id). Jika data pengguna ditemukan, maka sistem akan mengembalikan data pengguna dalam format JSON yang terdiri dari id, username, email, dan avatar, dengan kode status HTTP 200 OK.

Namun, jika data pengguna tidak ditemukan (misalnya karena ID tidak valid atau pengguna sudah dihapus), maka sistem akan mengembalikan respons JSON berisi pesan kesalahan "User not found" dan error: true dengan kode status HTTP 404 Not Found.

16. Api Service Android Kotlin to Flask-API

Konfigurasi Api Config

```
fun getApiService(): ApiService {
    val loggingInterceptor =
HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Level.BODY)

    // Tambahkan timeout dan retry mekanisme
    val client = OkHttpClient.Builder()
        .connectTimeout(30, TimeUnit.SECONDS) // Timeout koneksi 30
detik
        .readTimeout(30, TimeUnit.SECONDS)    // Timeout membaca
respons 30 detik
        .writeTimeout(30, TimeUnit.SECONDS)   // Timeout mengirim
data 30 detik
        .addInterceptor(loggingInterceptor)
        .addInterceptor { chain ->
            var response: Response
            var tryCount = 0
            val maxTries = 3 // Coba ulang hingga 3 kali jika gagal

            do {
                try {
                    response = chain.proceed(chain.request())
                } catch (e: IOException) {
                    tryCount++
                    if (tryCount >= maxTries) throw e // Jika sudah
mencoba 3 kali, lempar error
                    continue
                }
                break
            } while (true)

            response
        }
        .build()

    val retrofit = Retrofit.Builder()
        .baseUrl("https://sunfish-caring-readily.ngrok-free.app/")
        .addConverterFactory(GsonConverterFactory.create())
        .client(client)
```

```
        .build()

    return retrofit.create(ApiService::class.java)
}
```

Fungsi getApiService() memulai dengan membuat sebuah HttpLoggingInterceptor, yaitu alat untuk mencatat semua aktivitas HTTP yang dilakukan oleh aplikasi. Dalam hal ini, tingkat pencatatan ditetapkan ke BODY, artinya seluruh isi permintaan dan respons akan dicatat, berguna untuk debugging.

Selanjutnya, dibuatlah sebuah objek OkHttpClient yang dikonfigurasi dengan:

- **Timeout koneksi**, **timeout membaca**, dan **timeout menulis** masing-masing selama 30 detik. Ini bertujuan untuk mencegah aplikasi menunggu terlalu lama ketika tidak ada respons dari server.
- Penambahan loggingInterceptor agar setiap request dan response dapat dicatat.
- Penambahan interceptor khusus yang berfungsi untuk **mekanisme retry** atau **percobaan ulang**. Interceptor ini akan mencoba permintaan HTTP maksimal 3 kali jika terjadi IOException (misalnya kegagalan jaringan). Jika setelah tiga kali percobaan masih gagal, maka error akan dilemparkan.

Setelah konfigurasi client selesai, dibuatlah instance Retrofit dengan:

- Base URL yang ditetapkan ke "https://sunfish-caring-readily.ngrok-free.app/" sebagai alamat utama dari API yang akan diakses.
- Konverter JSON menggunakan GsonConverterFactory agar respons JSON dari server bisa diubah menjadi objek Kotlin secara otomatis.
- client yang sudah dikustomisasi tadi ditambahkan ke konfigurasi Retrofit.

Terakhir, Retrofit digunakan untuk membuat implementasi dari ApiService, yaitu interface yang berisi definisi endpoint-endpoint yang akan dipanggil. Hasil dari getApiService() adalah instance siap pakai dari ApiService yang sudah lengkap dengan logging, timeout, dan retry mechanism.

Konfigurasi API Service

```
interface ApiService {

    @FormUrlEncoded
    @POST("login")
    suspend fun login(
        @Field("email") email: String,
        @Field("password") password: String
    ): LoginResponse

    @FormUrlEncoded
    @POST("register")
    suspend fun register(
        @Field("username") name: String,
        @Field("email") email: String,
        @Field("password") password: String
```

```
    ): RegisterResponse

    @Multipart
    @POST("predict")
    suspend fun uploadImageDaun(
        @Header("Authorization") token: String,
        @Part file: MultipartBody.Part,
    ): DaunNewResponse

    @Multipart
    @POST("predict_bunga")
    suspend fun uploadImageBunga(
        @Header("Authorization") token: String,
        @Part file: MultipartBody.Part,
    ): DaunNewResponse

    @Multipart
    @POST("predict_rimpang")
    suspend fun uploadImageRimpang(
        @Header("Authorization") token: String,
        @Part file: MultipartBody.Part,
    ): DaunNewResponse

    @GET("my_profile")
    suspend fun getMyProfile(
        @Header("Authorization") token: String
    ): ProfileResponse

    @GET("history")
    suspend fun getHistory(
        @Header("Authorization") token: String
    ): List<HistoryResponseItem>

    @Multipart
    @PUT("edit_profile")
    suspend fun editProfile(
        @Header("Authorization") token: String,
        @Part avatar: MultipartBody.Part,
        @Part("username") name: RequestBody?,
        @Part("email") email: RequestBody?,
    ): RegisterResponse

    @Multipart
    @PUT("change_password")
    suspend fun editPassword(
        @Header("Authorization") token: String,
        @Part("old_password") oldPassword: RequestBody?,
        @Part("new_password") newPassword: RequestBody?,
    ): RegisterResponse

    @Multipart
    @POST("api/send_reset_token")
    suspend fun resetPassword(
        @Part("email") oldPassword: RequestBody?,
    ): RegisterResponse

    @Multipart
    @POST("api/verify_reset_token")
    suspend fun verifyResetPassword(
        @Part("email") email: RequestBody?,
        @Part("token") token: RequestBody?,
```

```
        @Part("new_password") newPassword: RequestBody?,
    ): RegisterResponse




//    @FormUrlEncoded
//    @POST("chat")
//    suspend fun chatWithTheBit(
//        @Field("chatInput") chatInput : String
//    ): ChatResponse
//
    @GET("herbal")
    suspend fun getSpices(
    ): HerbalResponse

    @GET("api/articles")
    suspend fun getArticle(
    ): ArticleResponse

    @GET("get_all_herbals")
    suspend fun getPostHerbal(
    @Header("Authorization") token: String,
    ): LocHerbalResponse

    @GET("get_my_herbals")
    suspend fun getMyHerbal(
        @Header("Authorization") token: String,
    ): LocHerbalResponse
//
    @Multipart
    @POST("post_herbal")
    suspend fun uploadProductSpice(
    @Header("Authorization") token: String,
    @Part avatar: MultipartBody.Part,
    @Part("name") name: RequestBody?,
    @Part("description") description: RequestBody?,
    @Part("no_wa") noWa:RequestBody?,
    @Part("lat") lat: RequestBody?,
    @Part("lon") lan: RequestBody?

    ): PostHerbalResponse
//
    @GET("get_all_herbals")
    suspend fun getHerbalWithLocation(
    @Header("Authorization") token: String,
    @Query("location") location : Int = 1,
    ): LocHerbalResponse


}
```

Interface ApiService adalah kumpulan definisi endpoint (URL dan metode HTTP) yang digunakan aplikasi untuk **berkomunikasi dengan backend (server)** melalui HTTP. Masing-masing fungsi merepresentasikan satu permintaan ke server dan mengembalikan hasilnya secara suspend (asinkron), cocok digunakan dengan **Coroutine**.

Autentikasi dan Registrasi:

1. login(...)

   Mengirim permintaan POST ke endpoint login dengan data email dan password. Responsnya berupa LoginResponse.

2. register(...)

   Mengirim data pendaftaran (username, email, password) ke endpoint register dan menerima RegisterResponse.

Upload Gambar untuk Prediksi (Model AI):

3. uploadImageDaun(...)
4. uploadImageBunga(...)
5. uploadImageRimpang(...)

   Ketiga fungsi ini mengirim gambar (daun, bunga, atau rimpang) ke server menggunakan POST bertipe Multipart. Semua memerlukan token Authorization dan mengembalikan DaunNewResponse.

Profil dan Password:

6. getMyProfile(...)

   Mengambil data profil pengguna menggunakan token autentikasi. Mengembalikan ProfileResponse.

7. editProfile(...)

   Mengubah data profil (gambar/avatar, username, email) dengan metode PUT. Token dibutuhkan. Format data dikirim menggunakan Multipart.

8. editPassword(...)

   Mengubah password lama ke password baru. Butuh token dan dikirim dalam bentuk Multipart.

Reset Password:

9. resetPassword(...)

   Mengirim email pengguna untuk mendapatkan token reset password.

10. verifyResetPassword(...)

    Memverifikasi token dan mengganti password lama dengan yang baru.

Data Herbal:

11. getArticle()

    Mengambil daftar artikel dari API endpoint api/articles.

12. getPostHerbal(...)

    Mengambil seluruh data tanaman herbal dari pengguna lain (semua lokasi). Butuh token.

13. getMyHerbal(...)

    Mengambil data tanaman herbal yang diunggah oleh pengguna itu sendiri. Butuh token.

14. uploadProductSpice(...)

    Mengunggah data tanaman herbal milik pengguna, termasuk gambar, nama, deskripsi, nomor WhatsApp, dan koordinat lokasi (lat, lon). Token dibutuhkan.

15. getHerbalWithLocation(...)

    Mengambil data tanaman herbal dengan parameter lokasi tertentu. Default location = 1.

Lampiran 5. Sertifikat HKI

REPUBLIK INDONESIA
KEMENTERIAN HUKUM

SURAT PENCATATAN CIPTAAN

Dalam rangka pelindungan ciptaan di bidang ilmu pengetahuan, seni dan sastra berdasarkan Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta, dengan ini menerangkan:

Nomor dan tanggal permohonan	:	EC002025076246, 26 Juni 2025
Pencipta		
Nama	:	**Nur Rizqi Maulana, Hepatika Zidny Ilmadina, S.Pd., M.Kom. dkk**
Alamat	:	Jalan Sunan Amangkurat 1 No.11 Rt.11 Rw.02 Desa Lemahduwur, Adiwerna, Kab. Tegal, Jawa Tengah, 52194
Kewarganegaraan	:	Indonesia
Pemegang Hak Cipta		
Nama	:	**Pusat Penelitian dan Pengabdian Masyarakat (P3M) Politeknik Harapan Bersama**
Alamat	:	Jalan Mataram No. 9, Pesurungan Lor, Kecamatan Margadana 52142, Margadana, Kota Tegal, Jawa Tengah, 52142
Kewarganegaraan	:	Indonesia
Jenis Ciptaan	:	**Program Komputer**
Judul Ciptaan	:	**HerbalinApps: Aplikasi Klasifikasi Tanaman Obat Herbal Berdasarkan Citra Menggunakan Model EfficientNetV2B0**
Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia	:	26 Juni 2025, di Kota Tegal
Jangka waktu pelindungan	:	Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.
Nomor Pencatatan	:	000916507

adalah benar berdasarkan keterangan yang diberikan oleh Pemohon.
Surat Pencatatan Hak Cipta atau produk Hak terkait ini sesuai dengan Pasal 72 Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

a.n. MENTERI HUKUM
DIREKTUR JENDERAL KEKAYAAN INTELEKTUAL
u.b
Direktur Hak Cipta dan Desain Industri

Agung Damarsasongko,SH.,MH.
NIP. 196912261994031001

Balai Besar Sertifikasi Elektronik

Disclaimer:
1. Dalam hal pemohon memberikan keterangan tidak sesuai dengan surat pernyataan, Menteri berwenang untuk mencabut surat pencatatan permohonan.
2. Surat Pencatatan ini telah disegel secara elektronik menggunakan segel elektronik yang ditertibkan oleh Balai Besar Sertifikasi Elektronik, Badan Siber dan Sandi Negara.
3. Surat Pencatatan ini dapat dibuktikan keasliannya dengan memindai kode QR pada dokumen ini dan informasi akan ditampilkan dalam browser.

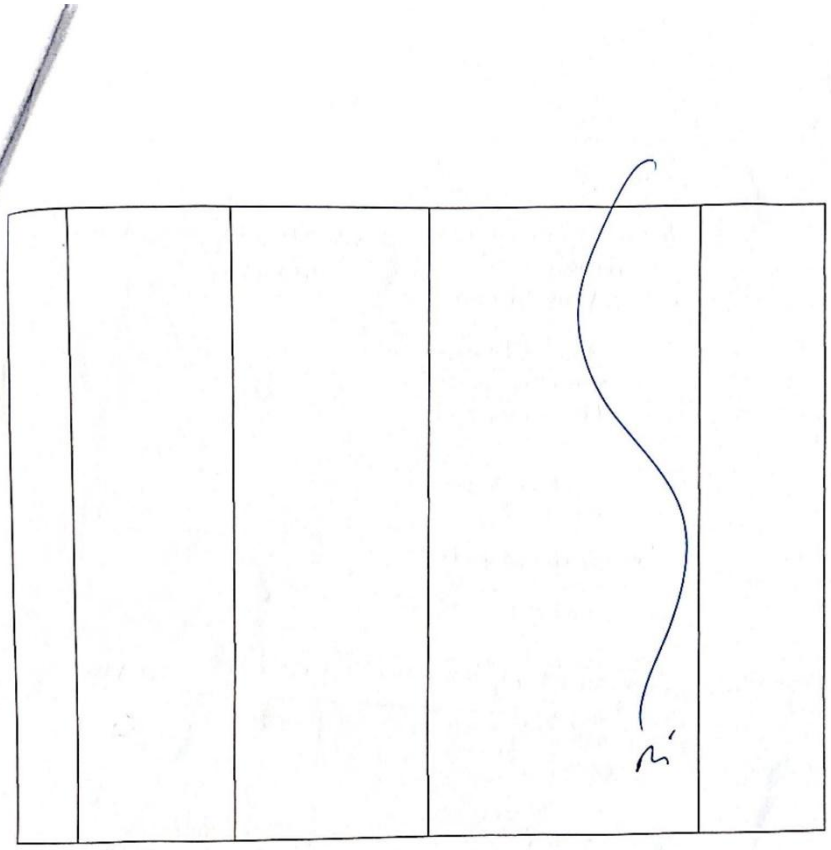
Lampiran 6. Lembar Bimbingan

SARJANA TERAPAN TEKNIK INFORMATIKA
POLITEKNIK HARAPAN BERSAMA

Nama	: Nur Rizqi Maulana
NIM	: 21090068
No.Ponsel	: 08816559397
Judul TA	: Mobile Klasifikasi Tanaman Obat Herbal berdasarkan Citra menggunakan Model Efficientnetv2s
Dosen Pembimbing I	: Hepatika Zidny Ilmadina, S.Pd., M.Kom.

No	Tanggal	Pemeriksaan	Perbaikkan yang perlu di lakukan	Paraf
1.	19/03/2025	*aplikasi	* fitur → daftar perawatan keluarga yang dapat ditangani / diredakan dg herbal ✓ ⇒ muncul pengolahan yg spesifik ke sakitnya.	
2.	23/04/2025	*aplikasi	* fitur : pada saat setelah deteksi ada pengolahan yang spesifik ; tapi tidak full artikel ; bentuknya tautan ✓	
		*model	* coba studi literatur cari penelitian serupa, ujikan modelnya, bandingkan dg efficientNet, bagus mana ? ✓	

3.	19/05/2025	* produk dan model	* produk untuk khasiat sudah sesuai per masing" dam/rimpang * model : sudah membandingkan beberapa model ⇓ Implementasi best modelnya *lanjutkan ke manual book	
4.	2/06/2025	* model	*manual book terkirim di email * dokumen teknik lengkapi * berkas HKI lengkapi * best model : selesaikan sesuai yg paling baik dan ringan di aplikasi hepatika.tidng@polteksgl.ac.id	
5.	26/06/2025	* laporan HKI	* pengecekan laporan * revisi laporan	
6.	27/06/2025	* laporan	* acc	



Tegal, Juli 2025
Dosen Pembimbing I

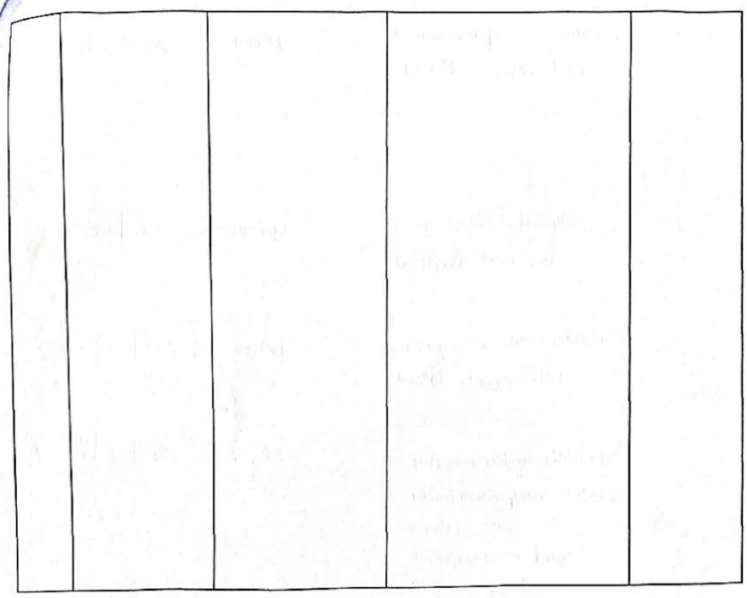
Hepatika Zidny Ilmadina, S.Pd., M.Kom.
NIPY: 08.017.340

SARJANA TERAPAN TEKNIK INFORMATIKA
POLITEKNIK HARAPAN BERSAMA

Nama : Nur Rizqi Maulana
NIM : 21090068
No.Ponsel : 08816559397
Judul TA : Mobile Klasifikasi Tanaman Obat Herbal berdasarkan Citra menggunakan Model Efficientnetv2s
Dosen Pembimbing II : Mirza Alim Mutasodirin, S.Kom., M.Kom.

No	Tanggal	Pemeriksaan	Perbaikkan yang perlu di lakukan	Paraf
1.	12/03/2025		Perbaikan problem statement	
2.	21/03/2025		Perbaikan Tujuan dan manfaat	
3.	11/04/2025		Perbaikan data	
4.	25/04/2025		Literatur Review	
5.	03/05/2025	Pemeriksaan Aplikasi	Progres fitur - login - register - clanfication	

6.	14/05/2025	model	membandingkan performa model dengan HPO	
7.	16/05/2025	Aplikasi	fitur artikel : Integrasi backend ke mobile	
8.	30/05/2025	model	melanjutkan perbandingan model dengan HPO	
9.	4/06/2025	Model	- menambahkan dataset - membandingkan performa model dan mengevaluasi hasil akurasi model	
10.		Aplikasi	membuat Test case aplikasi	



Tegal, Juni 2025
Dosen Pembimbing II

Mirza Alim Mutasodirin, S.Kom., M.Kom.
NIPY: 03.023.534