LAMPIRAN

Lampiran 1 Surat Kesediaan Bimbingan Skripsi

SURAT KESEPAKATAN BIMBINGAN SKRIPSI

Kami yang bertanda tangan di bawah ini :
Pihak Pertama

Nama	: Zulfatun Nisa
NIM	: 20090143
Program Studi	: D IV Teknik Informatika

Pihak Kedua

Nama	: Ir. Ginanjar Wiro Sasmito, M.Kom.
Status	: Dosen
NIDN	: 0613028601
Jabatan Fungsional	: Lektor Kepala
Pangkat/Golongan	: III/D (Penata Tingkat I)

Pada hari ini Kamis tanggal 14 Maret 2024 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I Skripsi Pihak Pertama dengan syarat Pihak Pertama wajib melakukan bimbingan Skripsi minimal 8 kali kepada Pihak Kedua. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.
Demikian kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Skripsi.

Tegal, 14 Maret 2024

Pihak Pertama	Pihak Kedua
Zulfatun Nisa	Ir. Ginanjar Wiro Sasmito, M.Kom.

Mengetahui
Ketua Program Studi Sarjana Terapan Teknik Informatika

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

SURAT KESEPAKATAN BIMBINGAN SKRIPSI

Kami yang bertanda tangan di bawah ini :

Pihak Pertama

Nama	: Zulfatun Nisa
NIM	: 20090143
Program Studi	: D IV Teknik Informatika

Pihak Kedua

Nama	: Ardi Susanto, S.Kom., M.Cs.
Status	: Dosen
NIDN	: 0629109301
Jabatan Fungsional	: Asisten Ahli
Pangkat/Golongan	: Penata Muda TK.I-III/b

Pada hari ini Kamis tanggal 14 Maret 2024 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing II Skripsi Pihak Pertama dengan syarat Pihak Pertama wajib melakukan bimbingan Skripsi minimal 8 kali kepada Pihak Kedua. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

Demikian kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Skripsi.

Tegal, 14 Maret 2024

Pihak Pertama	Pihak Kedua
Zulfatun Nisa	Ardi Susanto, S.Kom., M.Cs.

Mengetahui
Ketua Program Studi Sarjana Terapan Teknik Informatika

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

Lampiran 2 Surat Observasi Pada dokter mata

POLITEKNIK HARAPAN BERSAMA
The True Vocational Campus

Sarjana Terapan Teknik Informatika

Nomor : 41.03/TI.PHB/V/2024
Lampiran : -
Hal : Permohonan Izin Observasi
Kepada Yth. : **dr. Rima Octaviani,Sp.M**
di RS Harapan sehat Slawi

Dengan hormat, mahasiswa dengan identitas berikut ini:

nama : Zulfatun Nisa
NIM : 20090143
prodi : Sarjana Terapan Teknik Informatika

Bermaksud melakukan penelitian untuk keperluan Skripsi dengan judul **"Sistem Diagnosa Penyakit Mata Pada Anak Menggunakan Yolov8"**. Kami memohon Bapak/Ibu memberikan izin kepada mahasiswa yang bersangkutan agar memperoleh data, keterangan, dan bahan yang diperlukan.

Demikian permohonan ini disampaikan, atas perhatian kami ucapkan terima kasih.

Tegal, 27 Mei 2024
Ka. Prodi S.Tr. Teknik Informatika,

Dyah Aprilliani, S.T., M.Kom
NIPY. 09.015.226

Jl. Mataram No. 9 Kota Tegal 52143, Jawa Tengah, Indonesia.
(0283)352000
informatika@poltektegal.ac.id
poltektegal.ac.id

PT. ATI SRI SUBEKTI HARAPAN SEHAT
RUMAH SAKIT HARAPAN SEHAT SLAWI
Jl. Raya Gatot Subroto Slawi Tegal Jawa Tengah
No. Telp. (0283) 4563611, E-mail: rshs.slawi@gmail.com
TERAKREDITASI PARIPURNA KARS

Nomor : 0411/RSHS-SLW/SB/V/2024 Slawi, 28 Mei 2024

Hal : Surat Balasan

Kepada Yth,

Kaprodi Sarjana Terapan Teknik Informatika Politeknik Harapan Sehat Slawi

di Tempat

Dengan hormat,

Menindaklanjuti surat permohonan Ijin Obsevasi Kegiatan Penelitian Mahasiswa Program Studi Sarjana Terapan Informatika. Dengan ini kami informasikan bahwa kami menerima mahasiswa atas nama Zulfatun Nisa dari Politeknik Harapan Sehat Slawi untuk melakukan Observasi Kegiatan Penelitian di RS Harapan Sehat Slawi.

Demikian surat balasan ini kami sampaikan. Atas perhatian dan kerjasamanya kami ucapkan terimakasih.

Cp. 085713379037 (Laela Eprihatni)

RS Harapan Sehat Slawi
Direktur

dr. M. Dipa Daulatala

Tembusan :

1. Arsip

PT. ATI SRI SUBEKTI HARAPAN SEHAT
RUMAH SAKIT HARAPAN SEHAT SLAWI
Jl. Raya Gatot Subroto Slawi Tegal Jawa Tengah
No. Telp. (0283) 4563611, E-mail: rshs.slawi@gmail.com
TERAKREDITASI PARIPURNA KARS

SURAT KETERANGAN

Nomor : 0445/RSHS-SLW/KET/VI/2024

Kepada Yth,

Kaprodi Sarjana Terapam Teknik Informatika Politeknik Harapan Bersama Tegal

di Tempat

Dengan Hormat,

Sehubungan dengan permohonan Ijin Observasi kegiatan Penelitian Mahasiswa, dengan ini dokter Rima Octaviani telah melihat data penelitian berupa gambar penyakit mata pada anak yang akan digunakan dalam penelitian tugas akhir dengan judul "Sistem Diagnosa Penyakit Mata Pada Anak Menggunakan algoritma Yolo V8" oleh peneliti:

Nama	: Zulfatun Nisa
NIM	: 20090143
Program Studi	: Sarjana Terapan Teknik Informatika
Perguruan Tinggi	: Politeknik Harapan Bersama Tegal

Setelah memperlihatkan surat yang telah dibuat, maka data tersebut sudah bisa digunakan untuk penelitian.

Demikian surat keterangan ini dibuat agar dapat digunakan dalam penelitian Tugas Akhir.

RS Harapan Sehat Slawi
Direktur

dr. Yudia Mahardika
2405.2.0445

Tembusan :

1. Arsip

Lampiran 3 Surat Pernyataan HKI

SURAT PERNYATAAN

Yang bertanda tangan di bawah ini:

1. Nama : Zulfatun Nisa
   Kewarganegaraan : Indonesia
   Alamat : Ds. Tembok Luwung RT 038/ RW 008 Kec. Adiwerna, Kab. Tegal
2. Nama : Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng.
   Kewarganegaraan : Indonesia
   Alamat : Desa Kluwut RT/RW 003/002, Kecamatan Bulakamba, Kabupaten Brebes, Provinsi Jawa Tengah
2. Nama : Ardi Susanto,S.Kom,.M.Cs.
   Kewarganegaraan : Indonesia
   Alamat : Margadana,Tegal.

Dengan ini menyatakan bahwa:

1. Karya Cipta yang saya mohonkan :
   Berupa : Program Komputer
   Berjudul : Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8*.
   - 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 kami buat dengan sebenarnya dan untuk dipergunakan sebagimana mestinya.

Tegal, Juli 2024

(Zulfatun Nisa)

(Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng)

(Ardi Susanto,S.Kom,.M.Cs)

Lampiran 4 Surat Pengalihan HKI

SURAT PENGALIHAN HAK CIPTA

Yang bertanda tangan di bawah ini:

1. Nama : Zulfatun Nisa
   Kewarganegaraan : Indonesia
   Alamat : Ds. Tembok Luwung RT 038/ RW 008 Kec. Adiwerna, Kab. Tegal
2. Nama : Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng.
   Kewarganegaraan : Indonesia
   Alamat : Desa Kluwut RT/RW 003/002, Kecamatan Bulakamba, Kabupaten Brebes, Provinsi Jawa Tengah
2. Nama : Ardi Susanto,S.Kom,.M.Cs.
   Kewarganegaraan : Indonesia
   Alamat : Margadana, Tegal.

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

Nama : Pusat Penelitian dan Pengabdian Masyarakat (P3M)
Alamat : Jl. Mataram No. 9 Pesurungan Lor Kota Tegal

Adalah **Pihak II** selaku Pemegang Hak Cipta berupa Program Komputer dengan judul "Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8*". untuk didaftarkan di Direktorat Hak Cipta dan Desain Industri, Direktorat Jenderal Kekayaan Intelektual, Kementerian Hukum dan Hak Asasi Manusia Republik Indonesia.

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

Tegal, Juli 2024

Pemegang Hak Cipta
Ketua P3M

(Dr. Aldi Budi Riyanta, S.Si., M.T)

Pencipta

(Zulfatun Nisa)

(Ir. Ginanjar Wiro Sasmito, M.Kom., IPM.,ASEAN Eng)

(Ardi Susanto,S.Kom,.M.Cs.)

Lampiran 5 Manual Book dan Dokumen Teknikal

MANUAL BOOK

SISTEM DIAGNOSIS

PENYAKIT MATA PADA ANAK

MENGGUNAKAN ALGORITMA *YOLO V8*

Disusun Oleh :

Zulfatun Nisa

Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng.

Ardi Susanto ,S.Kom.,M.Cs.

1. PENDAHULUAN

1.1. Tujuan Pembuatan Dokumen

Dokumen *manual book* Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8* dibuat dengan tujuan menggambarkan, menjelaskan, panduan penggunaan bagaimana pengunaan sistem untuk digunakan pengguna.

1.2. Deskripsi Umum Sistem

Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8* yang dapat mendiagnosis penyakit mata pada anak dengan menggunakan inputan berupa foto (penyakit mata pada anak) Tujuan dibuatnya sistem ini adalah untuk menghasilkan suatu sistem yang dapat membantu masyarakat, khususnya orang tua yang anaknya mengalami gangguan penyakit mata, dalam mendeteksi dini penyakit yang dialami. Sistem ini juga memudahkan mereka untuk mendapatkan informasi kesehatan mata secara lebih mudah. Selain itu, sistem ini bertujuan untuk memberikan informasi tambahan kepada orang tua, sehingga mereka bisa mendapatkan pemahaman lebih lanjut mengenai perlunya datang ke klinik atau apotik terdekat untuk membeli obat yang sesuai dengan kondisi mata yang terdeteksi.

1.3. Deskripsi Dokumen

Dokumen ini dibuat untuk memberikan panduan penggunaan "Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8*". Dokumen ini berisikan informasi sebagai berikut:

1. **BAB I.**
   Berisi informasi umum yang merupakan bagian pendahuluan, yang meliputi tujuan pembuatan dokumen, deskripsi umum sistem serta deskripsi dokumen.
2. **BAB II.**
   Berisi perangkat yang dibutuhkan untuk "Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8*" meliputi perangkat lunak dan perangkat keras.
3. **BAB III.**
   Berisi *pengguna* manual Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8.*

2. PERANGKAT YANG DIBUTUHKAN

2.1 Perangkat Lunak

Perangkat lunak yang digunakan adalah sebagai berikut.

1. Sistem operasi seperti *windows*
2. *Web browser (Google Chrome, Microsoft Edge, dll)*

2.2 Perangkat Keras

Perangkat keras (*Hardware*) yang digunakan sebagai berikut:

1. Laptop maupun komputer pc.
2. *Mouse*.
3. *Keyboard*.

3. FITUR DAN CARA PENGGUNAAN

3.1 Struktur Fitur

Adapun struktur halaman pada Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8* adalah sebagai berikut.

1. **Halaman Utama Beranda**

Pada halaman beranda akan muncul tampilan awal *navbar* pada saat mengakses sistem,serta terdapat halaman tips tips menjaga kesehatan mata pada anak.

2. Halaman Konsultasi

Pada halaman utama pengguna ada beberapa fitur yang tersedia adalah sebagai berikut.

- Deteksi Penyakit menggunakan unggah gambar
- Deteksi Penyakit menggunakan kamera untuk ambil gambar
- Fitur hasil Diagnosis dan tips mengatasi penyakit mata anak
- Serta fitur untuk klinik dan apotek terdekat.

3. Halaman Utama Admin

Pada halaman utama admin ada beberapa fitur yang tersedia adalah sebagai berikut.

1. Fitur history
2. Fitur penyakit mata terbanyak

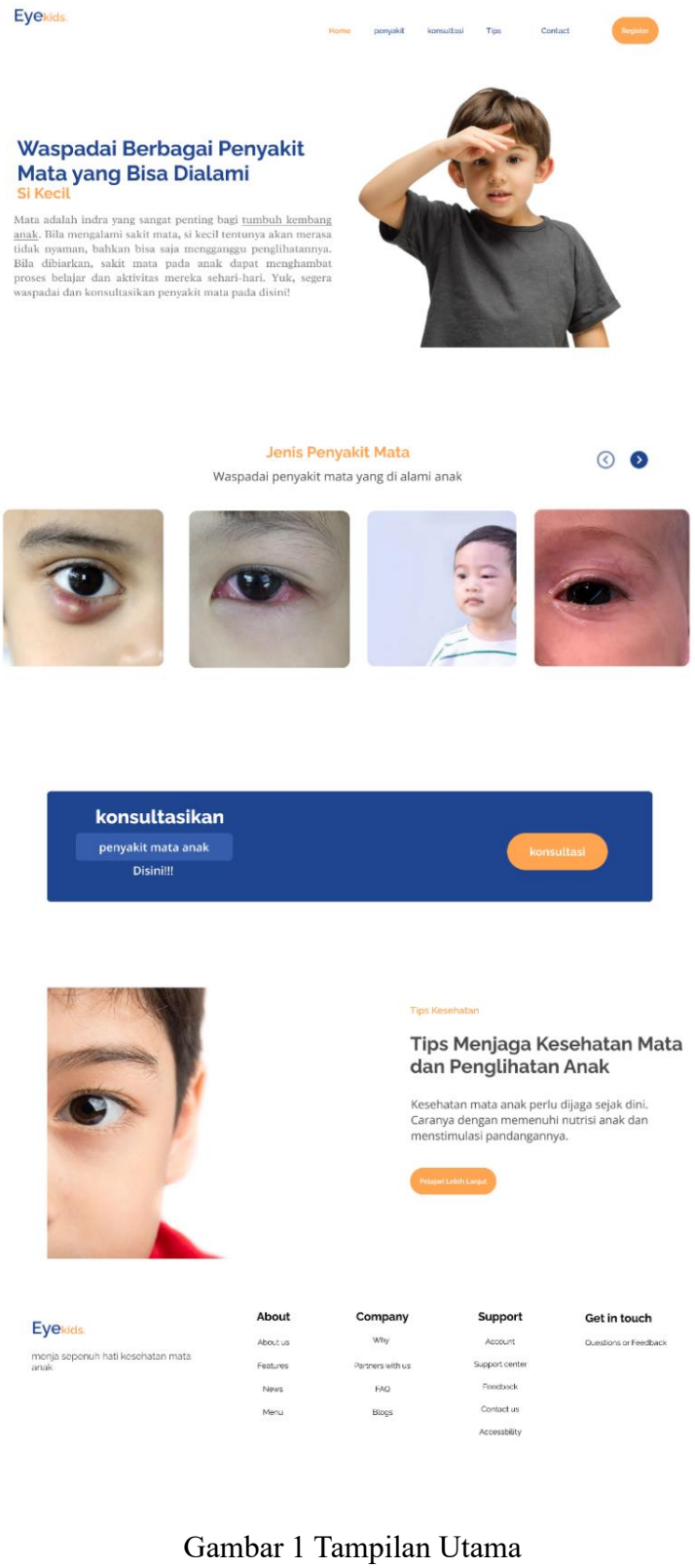
3.2 Pengguna

Pada bagian ini akan dijelaskan mengenai fitur-fitur yang tersedia disistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8*

1. Cara Membuka Sistem

Untuk menjalankan website Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8* sebagai berikut:

1. Bukalah *website* sistem melalui *web browser* yang tersedia seperti (*Google Chrome*, *Microsoft Edge*,dll).
2. Tekan tombol enter pada *keyboard*.
3. Setelah itu, akan muncul tampilan utama *website* sistem . Pada halaman terdapat fitur login,register dan *navbar.*Adapun tampilan utama dapat dilihat pada gambar 1.



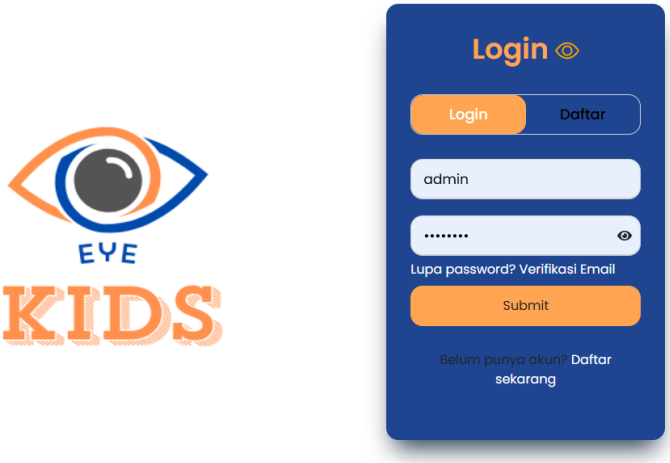
Gambar 1 Tampilan Utama

2. Halaman Pengguna

Pada sistem terdapat halaman pengguna yang terdiri dari berbagai fitur sebagai berikut:

1. Fitur *Login*

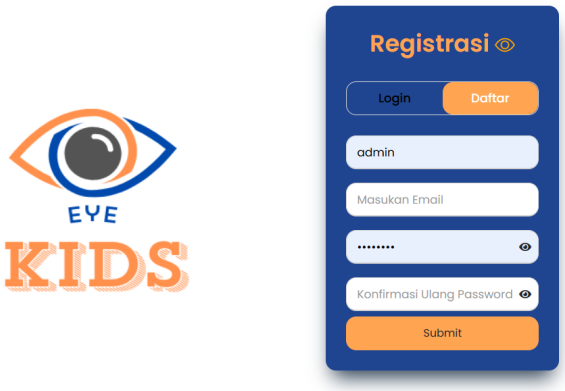
   Fitur *login* berisi form terdapat *username* dan *password* yangdigunakan untuk masuk pengguna. Adapun tampilan halaman *login* dapat dilihat pada gambar 2.



Gambar 2 Halaman *Login*

2. Fitur *Register*

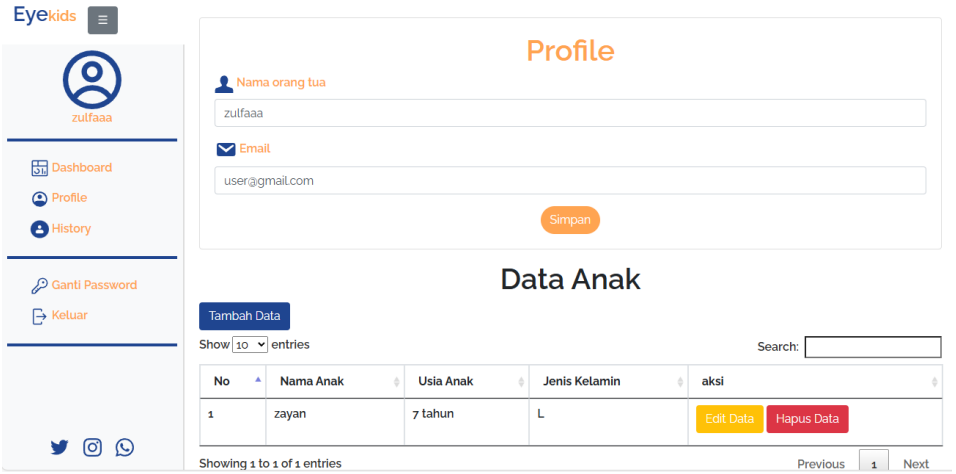
   Fitur *Register* berisi form terdapat masukan nama,masukan *email,*masukan *password* dan ulangi *password* yang digunakan untuk masuk pengguna. Adapun tampilan *register* dapat dilihat pada gambar 3



Gambar 3 Halaman *Register*

3. Fitur profil

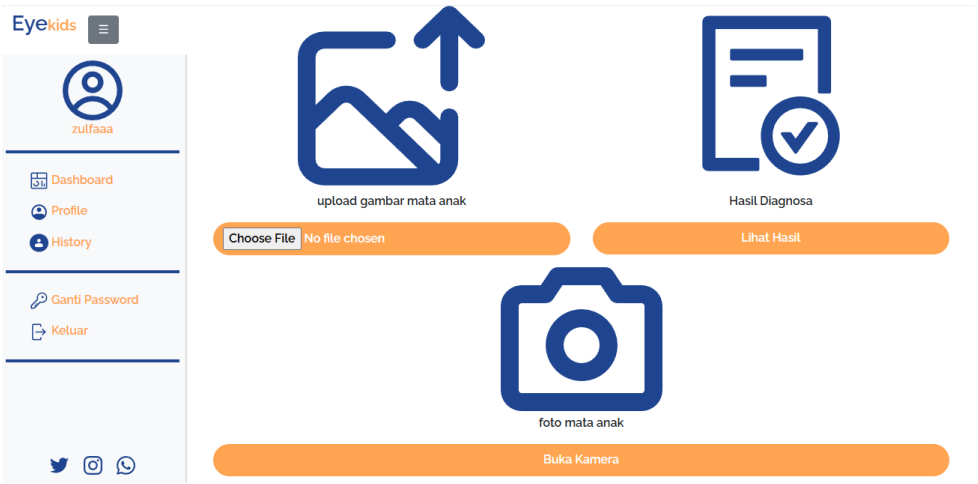
Setelah berhasil *login* maka akan langsung ke halaman profil,di dalam fitur profil pengguna diminta menginputkan nama,*email* dan data anak agar ketika konsultasi sudah tertera nama anak yang terkena penyakit mata.Adapun tampilan halaman profil dapat dilihat pada gambar 4.



Gambar 4 Halaman profil

4. Fitur konsultasi

fitur konsultasi yang dapat digunakan untuk mengetahui informasi terkena penyakit mata apakah anak. Cara pengunaannya dengan mengunggah foto atau mengambil potret dari kamera penyakit mata yang sedang di alami,kemudian sistem akan mendeteksi penyakit mata yang sedang di alami serta pengobatan dan rekomendasi klinik dan apotik terdekat.Adapun tampilan konsultasi dapat dilihat pada gambar 5.



Gambar 5 HalamanKonsultasi

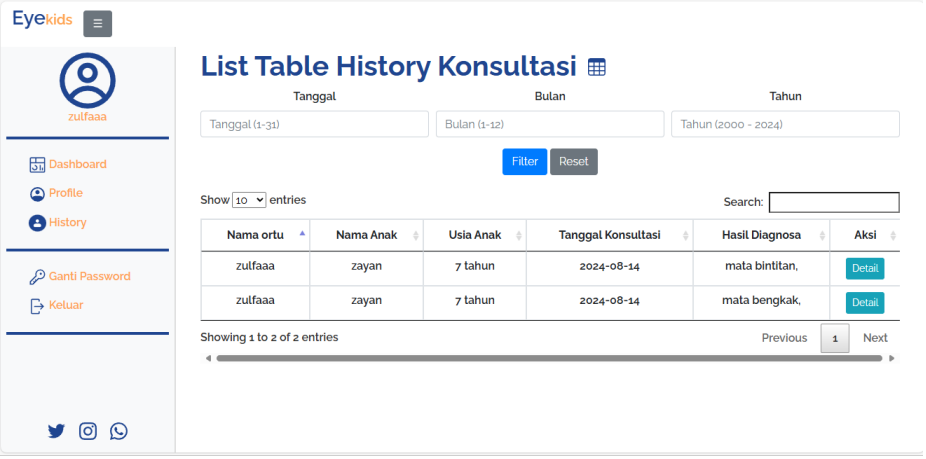
5. Fitur hasil Diagnosis

Pada fitur ini pengguna langsung bisa melihat penyakit mata apa yang sedang di alami anak beserta keterangan dan cara mengatasinya. Adapun tampilan hasil Diagnosis dapat dilihat pada gambar 6.

Gambar 6 Halaman hasil Diagnosis

6. Fitur History User

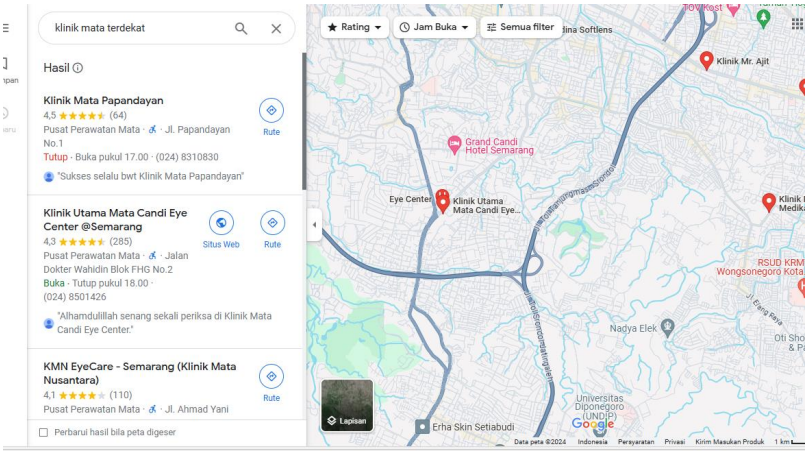
Pada Halaman *history* berisi fitur untuk melihat dan mengelola riwayat konsultasi kesehatan yang telah dilakukan. Informasi ini mencakup nama orang tua, nama anak, usia anak, tanggal konsultasi, dan tindakan yang telah diambil.



7. Fitur rekomendasi apotek dan klinik terdekat

Fitur apotek dan klinik terdekat bisa diakses dengan klik tombol apotek terdekat /klinik terdekat setelah muncul hasil diagnosis penyakit mata.setelah di klik akan mengarah ke *gmaps* dan mencari apotek

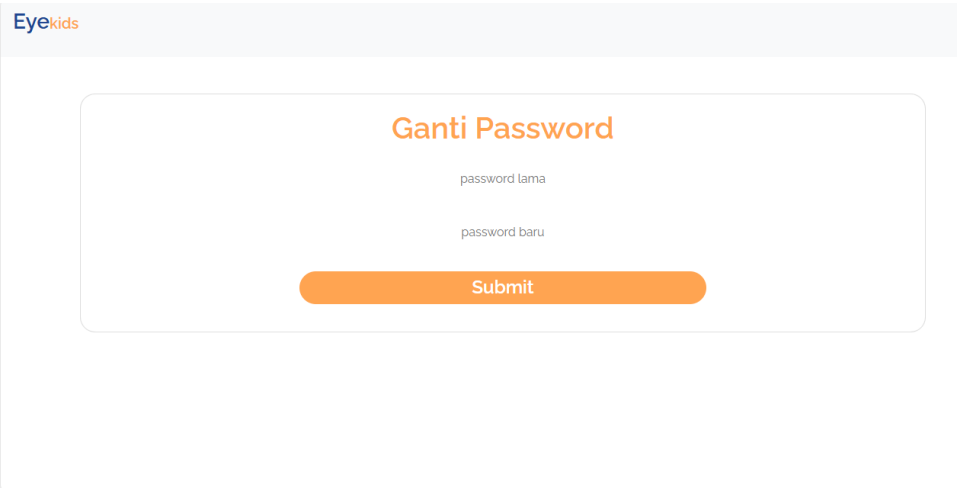
terdekat berdasarkan gps. tampilan apotek terdekat dapat dilihat pada gambar 7.



Gambar 7 Halaman rekomendasi klinik terdekat

8. Fitur ganti *password*

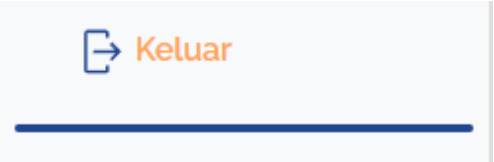
Fitur ini untuk mengubah *password* akun pengguna demi menjaga keamanan dan privasi. Pada halaman ganti *password*, pengguna akan diminta untuk memasukkan *password* lama dan *Password* baru kedalam *form* kemudian tekan tombol submit,maka password otomatis akan terganti.Gambar ganti *password* bisa dilihat pada gambar 8.



Gambar 8 Halaman ganti *password*

9. Fitur *Logout*

Fitur *logout* digunakan untuk keluar dari halaman pengguna dan, ketika diklik, akan membawa pengguna kembali ke halaman utama. Tampilan halaman logout dapat dilihat pada gambar 9.



Gambar 9 Halaman logout

10. Fitur Tips Menjaga kesehatan mata anak

Fitur ini berisi tips yang bermanfaat untuk menjaga kesehatan mata anak . Fitur tips menjaga kesehatan mata anak bisa diakses dengan klik tombol tips di fitur utama. Tampilan tips bisa dilihat pada gambar 10.

Gambar 10 Halaman Tips

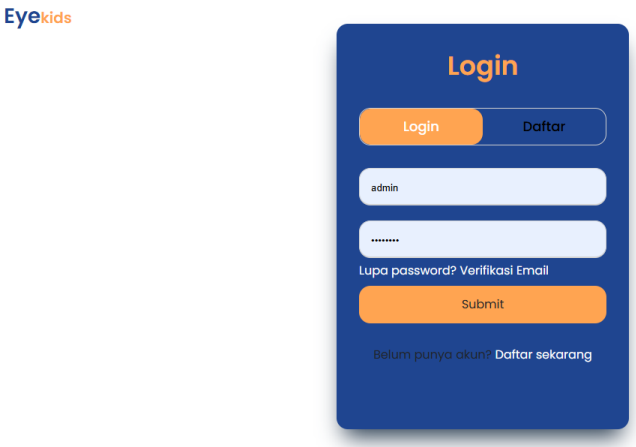
3. Halaman Admin

Pada sistem Sistem Diagnosis Penyakit Mata Pada Anak Menggunakan Algoritma *Yolo V8* terdapat halaman admin yang terdiri dari berbagai fitur sebagai berikut:

1. Fitur *login* admin

Fitur *login* admin berisi form terdapat *penggunaname* dan *password* yang digunakan untuk masuk admin. Adapun tampilan halaman *login* admin dapat dilihat pada gambar 11.

.



Gambar 11 Halaman login admin

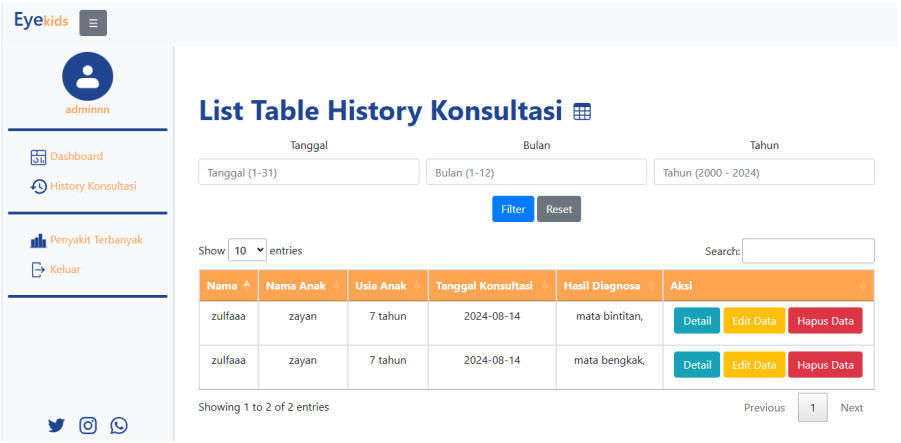
2. Fitur *dashboard* admin

Fitur *dashboard* admin ini berisi fitur *history* konsultasi penyakit secara lengkap dan penyakit yang paling banyak ditemukan.Tampilan halaman *dashboard* admin bisa dilihat pada gambar 12.

Gambar 12 Halaman *dashboard* admin.

3. Fitur *history* admin

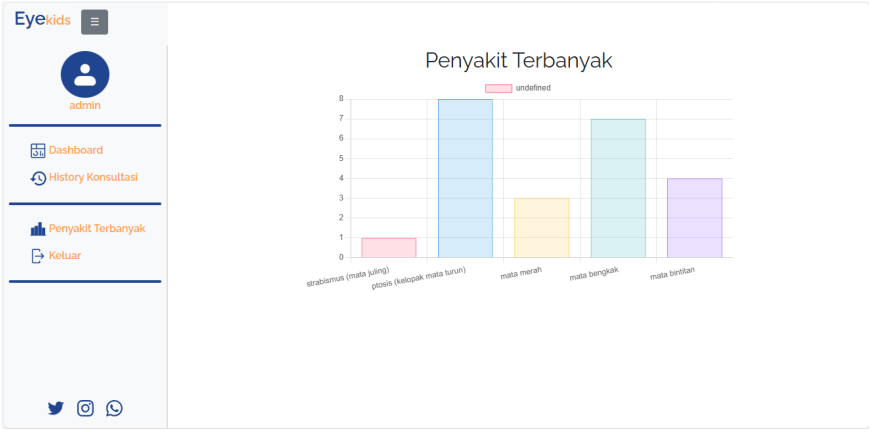
Pada Fitur *history* berisi fitur untuk melihat dan mengelola riwayat konsultasi kesehatan yang telah dilakukan. Informasi ini mencakup nama orang tua, nama anak, usia anak, tanggal konsultasi, dan tindakan yang telah diambil. Tampilan halaman *history* admin bisa dilihat pada gambar 13



Gambar 13 Halaman *history* admin.

4. Fitur penyakit terbanyak

Fitur ini menyediakan informasi tentang penyakit yang paling banyak terdeteksi oleh sistem. Dengan fitur ini, admin dapat memantau tren penyakit dan mengambil tindakan yang diperlukan untuk pencegahan dan penanganan. Tampilan halaman penyakit terbanyak admin bisa dilihat pada gambar 14.



Gambar 14 fitur penyakit terbanyak admin

DOKUMEN TEKNIKAL

SISTEM DIAGNOSIS PENYAKIT MATA PADA ANAK MENGGUNAKAN ALGORITMA *YOLO V8*

Penulis :

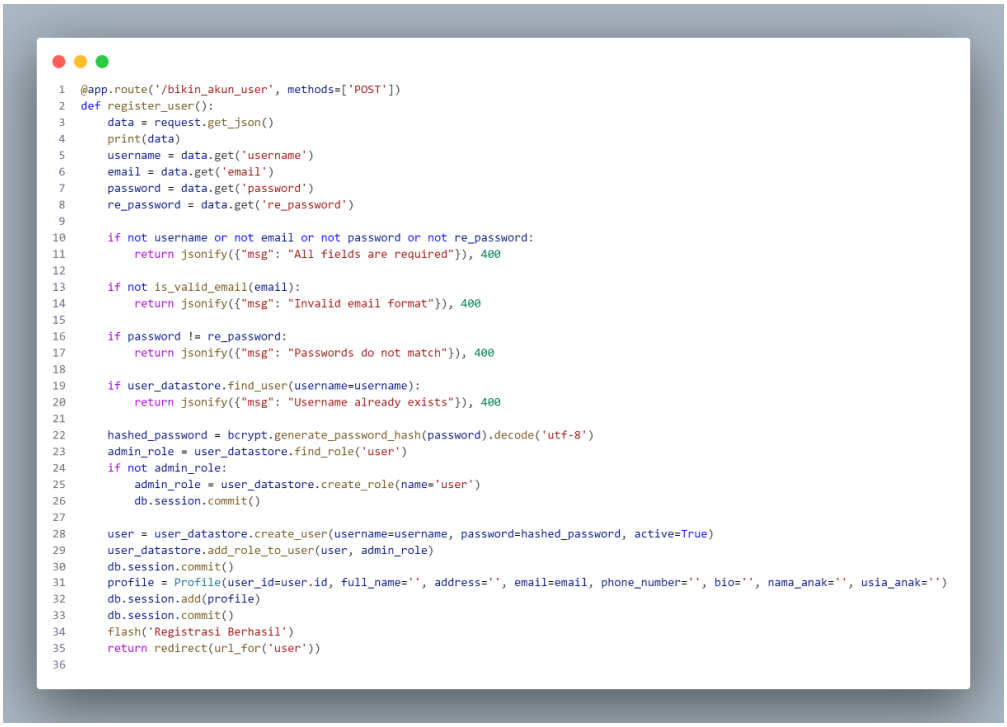
Zulfatun Nisa

Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng.

Ardi Susanto ,S.Kom.,M.Cs.

1. *Register* Pengguna

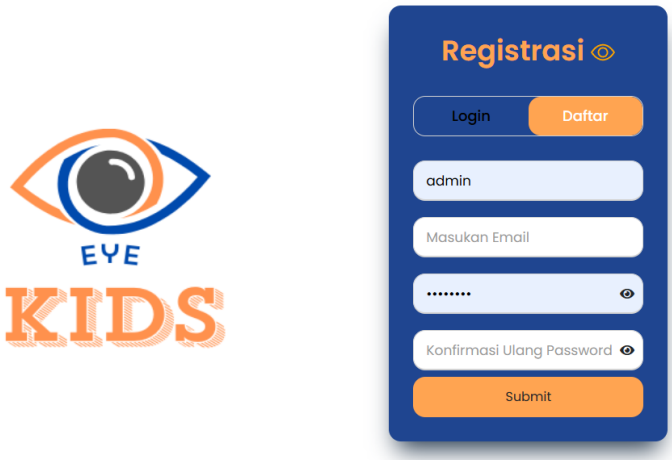
Tampilan registrasi ini menampilkan formulir yang terdiri dari 4 input yaitu *username*,email,masukkan *password*,ulangi *password.*



```python
@app.route('/bikin_akun_user', methods=['POST'])
def register_user():
    data = request.get_json()
    print(data)
    username = data.get('username')
    email = data.get('email')
    password = data.get('password')
    re_password = data.get('re_password')

    if not username or not email or not password or not re_password:
        return jsonify({"msg": "All fields are required"}), 400

    if not is_valid_email(email):
        return jsonify({"msg": "Invalid email format"}), 400

    if password != re_password:
        return jsonify({"msg": "Passwords do not match"}), 400

    if user_datastore.find_user(username=username):
        return jsonify({"msg": "Username already exists"}), 400

    hashed_password = bcrypt.generate_password_hash(password).decode('utf-8')
    admin_role = user_datastore.find_role('user')
    if not admin_role:
        admin_role = user_datastore.create_role(name='user')
        db.session.commit()

    user = user_datastore.create_user(username=username, password=hashed_password, active=True)
    user_datastore.add_role_to_user(user, admin_role)
    db.session.commit()
    profile = Profile(user_id=user.id, full_name='', address='', email=email, phone_number='', bio='', nama_anak='', usia_anak='')
    db.session.add(profile)
    db.session.commit()
    flash('Registrasi Berhasil')
    return redirect(url_for('user'))

```

a. *Endpoint* ini digunakan untuk mendaftarkan pengguna baru.
b. Memeriksa validitas data permintaan dan memastikan bahwa semua *field* yang diperlukan diberikan.
c. Memvalidasi format email dan memastikan bahwa kata sandi cocok.
d. Melakukan *hash* pada *password* yang diberikan dan membuat pengguna baru dengan peran "pengguna" di dalam pengguna_*datastore.*
e. Menambahkan profil pengguna baru ke dalam *database* dan mengirim respons bahwa pengguna berhasil didaftarkan.

Berikut tampilan *register* pengguna



2. *Login* pengguna

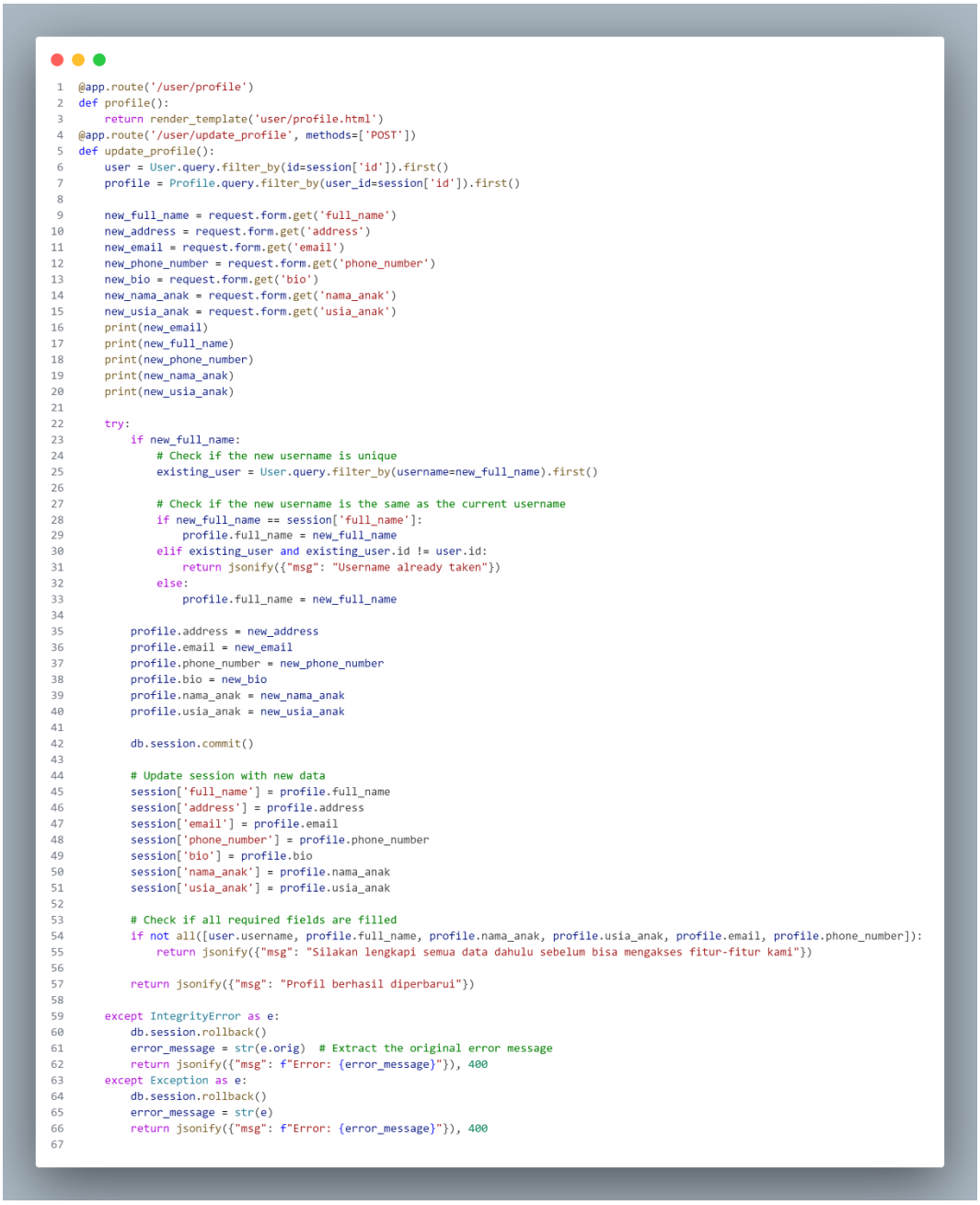
```python
# Endpoint untuk membuat token
@app.route('/login_user/proses', methods=['POST'])
def proses_user():
    data = request.get_json()
    if not data or 'username' not in data or 'password' not in data:
        return jsonify({"msg": "Invalid request"}), 400

    username = data['username']
    password = data['password']
    user = user_datastore.find_user(username=username)
    if not user:
        return jsonify({"msg": "username salah"}), 404

    if 'user' not in [role.name for role in user.roles]:
        return jsonify({"msg": "User is not a regular user"}), 403

    if not bcrypt.check_password_hash(user.password, password):
        return jsonify({"msg": "password salah"}), 401

    access_token = create_access_token(identity=username)
    session['jwt_token'] = access_token
    session['role'] = "user"
    session['full_name'] = username
    session['id'] = user.id
    profile = Profile.query.filter_by(user_id=user.id).first()
    if not profile:
        profile = Profile(user_id=user.id, full_name='', address='', email='', phone_number='', bio='', nama_anak='', usia_anak='')
        db.session.add(profile)
        db.session.commit()
    profile = Profile.query.filter_by(user_id=user.id).first()
    session['full_name'] = profile.full_name
    session['address'] = profile.address
    session['email'] = profile.email
    session['phone_number'] = profile.phone_number
    session['bio'] = profile.bio
    session['nama_anak'] = profile.nama_anak
    session['usia_anak'] = profile.usia_anak
    print(session)
    if profile.full_name == "" or profile.nama_anak == "" or profile.usia_anak == "" or profile.email == "" or profile.phone_number == "":
        return jsonify(access_token=access_token, redirect_url=url_for('profile'))
    else:
        return jsonify(access_token=access_token, redirect_url=url_for('dashboarduser'))

```

a. *Endpoint* ini memproses *login* untuk pengguna.

b. Memeriksa validitas data permintaan dan memastikan bahwa *username* dan *password* diberikan.
c. Memeriksa apakah pengguna ada di dalam pengguna_*datastore* dan apakah pengguna tersebut memiliki peran "pengguna".
d. Memvalidasi kata sandi yang dimasukkan dengan kata sandi yang tersimpan di *database.*
e. Jika valid, token *JWT* dibuat dan disimpan dalam *session,* serta data profil pengguna diisi ke dalam *session*.
f. Mengarahkan pengguna ke halaman profil jika profil tidak lengkap, atau ke *dashboard* pengguna jika profil sudah lengkap.

Berikut tampilan *login* pengguna

3. Profil pengguna



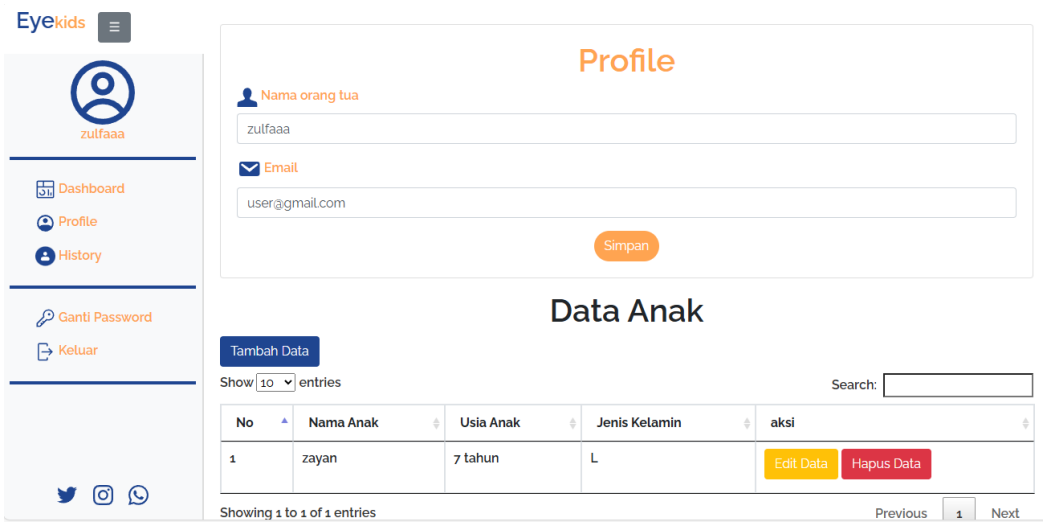
```python
@app.route('/user/profile')
def profile():
    return render_template('user/profile.html')
@app.route('/user/update_profile', methods=['POST'])
def update_profile():
    user = User.query.filter_by(id=session['id']).first()
    profile = Profile.query.filter_by(user_id=session['id']).first()

    new_full_name = request.form.get('full_name')
    new_address = request.form.get('address')
    new_email = request.form.get('email')
    new_phone_number = request.form.get('phone_number')
    new_bio = request.form.get('bio')
    new_nama_anak = request.form.get('nama_anak')
    new_usia_anak = request.form.get('usia_anak')
    print(new_email)
    print(new_full_name)
    print(new_phone_number)
    print(new_nama_anak)
    print(new_usia_anak)

    try:
        if new_full_name:
            # Check if the new username is unique
            existing_user = User.query.filter_by(username=new_full_name).first()

            # Check if the new username is the same as the current username
            if new_full_name == session['full_name']:
                profile.full_name = new_full_name
            elif existing_user and existing_user.id != user.id:
                return jsonify({"msg": "Username already taken"})
            else:
                profile.full_name = new_full_name

        profile.address = new_address
        profile.email = new_email
        profile.phone_number = new_phone_number
        profile.bio = new_bio
        profile.nama_anak = new_nama_anak
        profile.usia_anak = new_usia_anak

        db.session.commit()

        # Update session with new data
        session['full_name'] = profile.full_name
        session['address'] = profile.address
        session['email'] = profile.email
        session['phone_number'] = profile.phone_number
        session['bio'] = profile.bio
        session['nama_anak'] = profile.nama_anak
        session['usia_anak'] = profile.usia_anak

        # Check if all required fields are filled
        if not all([user.username, profile.full_name, profile.nama_anak, profile.usia_anak, profile.email, profile.phone_number]):
            return jsonify({"msg": "Silakan lengkapi semua data dahulu sebelum bisa mengakses fitur-fitur kami"})

        return jsonify({"msg": "Profil berhasil diperbarui"})

    except IntegrityError as e:
        db.session.rollback()
        error_message = str(e.orig)  # Extract the original error message
        return jsonify({"msg": f"Error: {error_message}"}), 400
    except Exception as e:
        db.session.rollback()
        error_message = str(e)
        return jsonify({"msg": f"Error: {error_message}"}), 400

```

Penjelasan

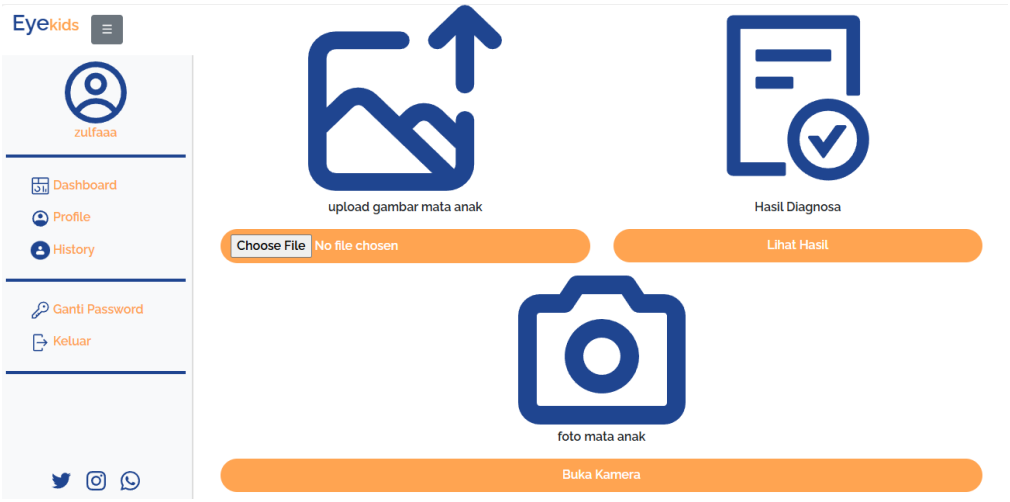
a. *Route* /pengguna/profile digunakan untuk menampilkan halaman profil pengguna.
b. Fungsi profile memanggil template pengguna/profile.html yang akan menampilkan informasi profil pengguna.
c. Mengambil data pengguna dan profil dari *database* berdasarkan *session['id']*.

d. Mengambil data yang diinputkan oleh pengguna melalui *form*: *full_name*, *address*, email, *phone_number*, nama_anak, usia_anak.,Mencetak data baru untuk *debugging*,memeriksa dan memperbarui *full_name*,Memastikan *username* baru unik dan tidak sama dengan *username* pengguna lain.
e. Jika *username* baru sama dengan *username* yang ada di session, maka diperbarui tanpa pengecekan.Memperbarui informasi profil lainnya (alamat, email, nomor telepon, bio, nama anak, usia anak),Menyimpan perubahan ke *database* dengan *db.session.commit()*.Memperbarui data *session* dengan informasi profil yang baru,memeriksa apakah semua *field* yang dibutuhkan telah diisi:
f. Jika tidak, mengirim pesan kesalahan bahwa semua data harus diisi sebelum mengakses fitur-fitur lain.
g. Jika berhasil, mengirim pesan bahwa profil berhasil diperbarui.
h. Menangani *IntegrityError* jika terjadi kesalahan integritas *database*, seperti duplikasi email.
i. Menangani *exception* lain dan mengembalikan pesan kesalahan.

Berikut tampilan profil pengguna:



4. Fitur konsultasi



Fungsi `dashboard pengguna()` berfungsi untuk mengatur akses ke halaman *dashboard* pengguna berdasarkan ketersediaan informasi dalam sesi profil pengguna . Pada saat pengguna mengakses *URL* `/pengguna/dashboard`, fungsi ini memeriksa apakah semua data penting seperti nama lengkap, nama anak, usia anak, email, dan nomor telepon dari profil pengguna sudah tersimpan dalam sesi pengguna. Jika salah satu informasi tersebut tidak lengkap, pengguna akan diarahkan kembali ke halaman profil untuk melengkapi data tersebut. Jika semua informasi telah lengkap, fungsi akan merender halaman `pengguna/dashboard.html`, memungkinkan pengguna untuk melihat konten *dashboard* mereka.

```python
#halaman dashboard user
@app.route('/user/dashboard')
def dashboarduser():
    if not all([session.get('full_name'), session.get('nama_anak'), session.get('usia_anak'), session.get('email'), session.get('phone_number')]):
        return redirect(url_for("profile"))
    else:
        return render_template('user/dashboard.html')
```

- Fitur konsultasi *upload image* dan *chapture image*

```python
@app.route('/predict', methods=['POST'])
def predict():
    file = request.files['gambar']
    if file is None or file.filename == '':
        return "error"

    img = Image.open(file).convert('RGB').resize((600, 300))
    img_io = BytesIO()
    img.save(img_io, 'JPEG', quality=70)
    img_io.seek(0)
    random_name = uuid.uuid4().hex + ".jpg"
    destination = os.path.join(app.config['UPLOAD_FOLDER'], random_name)
    img.save(destination)
    model_path = "./best.pt"
    save_path = "./app/static/detect"
    conf = 0.55
    print("loading..")
    # Perintah yang akan dijalankan
    command = [
        'yolo',
        'task=detect',
        'mode=predict',
        f'model={model_path}',
        f'conf={conf}',
        f'source={destination}',
        f'project={save_path}',
        'save=True'
    ]
    try:
        # Menjalankan perintah shell
        result = subprocess.run(command, capture_output=True, text=True, check=True)
        output = result.stdout

        # Logging untuk output proses
        print(f"Command output: {output}")

        # Mengecek kemunculan setiap nama dalam output
        names= ['strabismus (mata juling)','ptosis (kelopak mata turun)','mata merah','mata bengkak','mata bintitan']  # class names
        found_names = ""
        for name in names:
            if name in output:
                found_names += name+","
        current_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

        # Cari folder predict terbaru di dalam save_path
        subfolders = [f.path for f in os.scandir(save_path) if f.is_dir()]
        latest_folder = max(subfolders, key=os.path.getmtime)  # Folder terbaru berdasarkan waktu modifikasi
        latest_folder = latest_folder.replace("./app/static/detect\\", "")
        detected_file_path = latest_folder+"/"+random_name
        print(detected_file_path)
        # Menampilkan hasil
        if found_names !="":
            print("Found names and their counts in output:")
            print(found_names)
            history = History(
                nama_user=session['full_name'],
                nama_anak=session['nama_anak'],  # Ganti sesuai kebutuhan
                usia_anak=session['usia_anak'],  # Ganti sesuai kebutuhan
                tanggal_konsultasi=current_time,  # Ganti sesuai kebutuhan
                file_deteksi = detected_file_path,
                hasil_diagnosa= found_names
                )
            db.session.add(history)
            db.session.commit()
            new_history_id = history.id  # Access the ID of the newly added profile
            return jsonify({"msg":"SUKSES","id_hasil":new_history_id})
        else:
            history = History(
                nama_user=session['full_name'],
                nama_anak=session['nama_anak'],  # Ganti sesuai kebutuhan
                usia_anak=session['usia_anak'],  # Ganti sesuai kebutuhan
                tanggal_konsultasi=current_time,  # Ganti sesuai kebutuhan
                file_deteksi = detected_file_path,
                hasil_diagnosa="sehat"
            )
            db.session.add(history)
            db.session.commit()
            new_history_id = history.id  # Access the ID of the newly added profile
            print("None of the names were found in the output.")
            return jsonify({"msg":"SUKSES","id_hasil":new_history_id})

    except subprocess.CalledProcessError as e:
        return jsonify({'msg': e.stderr}), 500
```

Kode ini menangani pengunggahan gambar, memproses gambar tersebut menggunakan model *YOLOv8* untuk deteksi objek, menyimpan gambar yang telah terdeteksi, dan mencatat hasilnya ke dalam *database*. *Route* ini didefinisikan dengan *URL* `/predict` yang menerima permintaan *POST*. Pertama, kode ini mengambil file gambar yang diunggah dari permintaan menggunakan `request.files['gambar']`. Jika tidak ada file yang diunggah atau nama file kosong, kode akan mengembalikan pesan error.

Setelah file berhasil diunggah, gambar tersebut dibuka dan dikonversi ke format *RGB,* kemudian diubah ukurannya menjadi 600x300 piksel. Gambar yang sudah diproses kemudian disimpan ke dalam memori menggunakan objek *BytesIO* dengan format *JPEG* dan kualitas 70. Nama file acak kemudian dihasilkan menggunakan `uuid` dan gambar disimpan ke folder yang telah ditentukan dengan nama *file* tersebut.

Selanjutnya, *path* ke model *YOLOv8* dan path di mana gambar yang terdeteksi akan disimpan ditentukan, serta batas kepercayaan untuk model *YOLOv8* diatur ke 0.55. Perintah untuk menjalankan model *YOLOv8* dikonstruksi dengan menentukan tugas, mode, *path* model, ambang batas kepercayaan, sumber gambar, dan path penyimpanan. Perintah tersebut kemudian dijalankan menggunakan `subprocess.run`, dan *output* dari perintah tersebut ditangkap serta dicetak.

Output dari perintah *YOLOv8* kemudian diperiksa untuk setiap nama kondisi tertentu seperti *strabismus* (mata juling), *ptosis* (kelopak mata turun), mata merah, mata bengkak, dan mata bintitan. Jika nama kondisi ditemukan dalam output, nama tersebut ditambahkan ke dalam *string* `found_names`. Waktu saat ini juga dicatat untuk keperluan pencatatan ke *database*.

Untuk menyimpan hasil deteksi, *path* ke folder terbaru di dalam path penyimpanan ditentukan berdasarkan waktu modifikasi, dan *path* ke file yang terdeteksi dibangun. Jika ada nama kondisi yang ditemukan, objek `history` dibuat dengan hasil diagnosis dan ditambahkan ke *database*. Jika tidak ada nama kondisi yang ditemukan, objek `history` dibuat dengan diagnosis "sehat" dan ditambahkan ke *database*. ID dari catatan riwayat yang baru ditambahkan kemudian dikembalikan dalam respons JSON.

Jika terjadi kesalahan saat menjalankan perintah *YOLOv8,* pesan *error* akan dikembalikan dalam respons *JSON*. Kode ini secara keseluruhan menangani pengunggahan gambar, memproses gambar menggunakan model *YOLOv8,* menyimpan hasilnya, dan mencatat hasil deteksi ke dalam database.

5. Fitur hasil Diagnosis

```python
@app.route('/user/hasil_diagnosa/<id>')
def user_hasil_diagnosa(id):
    if 'full_name' not in session:
        abort(403)  # Forbidden, user tidak terautentikasi

    # Query data history berdasarkan id dan username dari session
    history_record = History.query.filter_by(id=id).first()
    if not history_record:
        abort(404)  # Not found, data history tidak ditemukan

    # Pastikan hasil_diagnosa adalah dictionary
    hasil_diagnosa_str = history_record.hasil_diagnosa

    hasil_diagnosa = hasil_diagnosa_str.split(",")

    if hasil_diagnosa[-1] == '':
        hasil_diagnosa.pop()
    print(hasil_diagnosa)
    # Query semua rekomendasi
    rekomendasi_records = Rekomendasi.query.all()
    print(rekomendasi_records)
    rekomendasi_list = [record.serialize() for record in rekomendasi_records]
    print(rekomendasi_list)

    # Gabungkan rekomendasi yang relevan dengan hasil diagnosa
    rekomendasi_diagnosa = {}
    for penyakit in hasil_diagnosa:
        for rekomendasi in rekomendasi_list:
            if rekomendasi['nama'] == penyakit:
                rekomendasi_diagnosa[penyakit] = rekomendasi
                print(rekomendasi['link_rekomendasi'])
                link_rekomendasi = rekomendasi['link_rekomendasi']
                link_rekomendasi = link_rekomendasi.split(",")
                rekomendasi_diagnosa[penyakit]['link_rekomendasi'] = link_rekomendasi
    print(rekomendasi_diagnosa)
    diagnosa = {
        'nama_user': history_record.nama_user,
        'nama_anak': history_record.nama_anak,
        'usia_anak': history_record.usia_anak,
        'tanggal_konsultasi': history_record.tanggal_konsultasi,
        'file_deteksi': history_record.file_deteksi,
        'hasil_diagnosa': hasil_diagnosa,
        'rekomendasi_diagnosa': rekomendasi_diagnosa,
    }
    print(diagnosa)

    return render_template('user/hasil_diagnosa.html', diagnosa=diagnosa)
```

a. Fungsi `pengguna_hasil_diagnosis` akan dipanggil ketika pengguna mengakses *URL* `/pengguna/hasil_Diagnosis/<id>`, di mana `<id>` adalah parameter dinamis yang mewakili ID dari hasil Diagnosis yang ingin diakses,kemudian mengecek apakah pengguna sudah terautentikasi dengan

memeriksa apakah *'full_name'* ada di *session*. Jika tidak, fungsi akan mengembalikan kode status *HTTP 403 (Forbidden).*

b. Mengambil data *history* berdasarkan ID dari *parameter URL*. Jika data *history* tidak ditemukan, fungsi akan mengembalikan kode *status HTTP 404 (Not Found).*kemudian,Mengambil *string* hasil diagnosis dari *record history* dan memisahkannya menjadi sebuah *list* dengan pemisah koma. Jika elemen terakhir dari *list* kosong, elemen tersebut dihapus.

c. Mengambil semua *record* rekomendasi dari database dan mengubahnya menjadi *list dictionary* yang bisa digunakan.

d. Membuat *dictionary* `rekomendasi_diagnosis` yang menggabungkan hasil diagnosis dengan rekomendasi yang relevan. Setiap penyakit dalam hasil diagnosis dicocokkan dengan rekomendasi yang ada, dan link rekomendasi dipisahkan menjadi list.

e. Membuat *dictionary* `Diagnosis` yang berisi informasi lengkap tentang pengguna, anaknya, hasil Diagnosis, dan rekomendasi yang relevan.

f. Mengembalikan halaman *HTML* yang dirender menggunakan template `pengguna/hasil_diagnosis.html` dengan data `diagnosis` yang sudah dipersiapkan.

Berikut tampilan dari hasil Diagnosis

6. Fitur rekomendasi apotek dan klinik terdekat

```
        <p class="font-semibold">{{ diagnosa.rekomendasi_diagnosa[penyakit].pengobatan|safe }}</p>
        {% for link in diagnosa.rekomendasi_diagnosa[penyakit]['link_rekomendasi'] %}
        {% if link == "klinik" %}
        <a class="btn btn-custom custom-bg-warning font-semibold" href="https://www.google.com/maps/search/klinik/">Rekomendasi Klinik</a>
        {% elif link == "apotek" %}
        <a class="btn btn-custom custom-bg-warning font-semibold" href="https://www.google.com/maps/search/apotek/">Rekomendasi Apotek</a>
        {% elif link == "rumah sakit"%}
        <a class="btn btn-custom custom-bg-warning font-semibold" href="https://www.google.com/maps/search/rumah+sakit/">Rekomendasi Apotek</a>
        {% elif penyakit == "sehat"%}
        <a class="btn btn-custom custom-bg-warning font-semibold" href="/tips">Tips Menjaga Kesehatan Mata</a>
        {% endif %}
        {% endfor %}
    {% endif %}
{% endfor %}
```

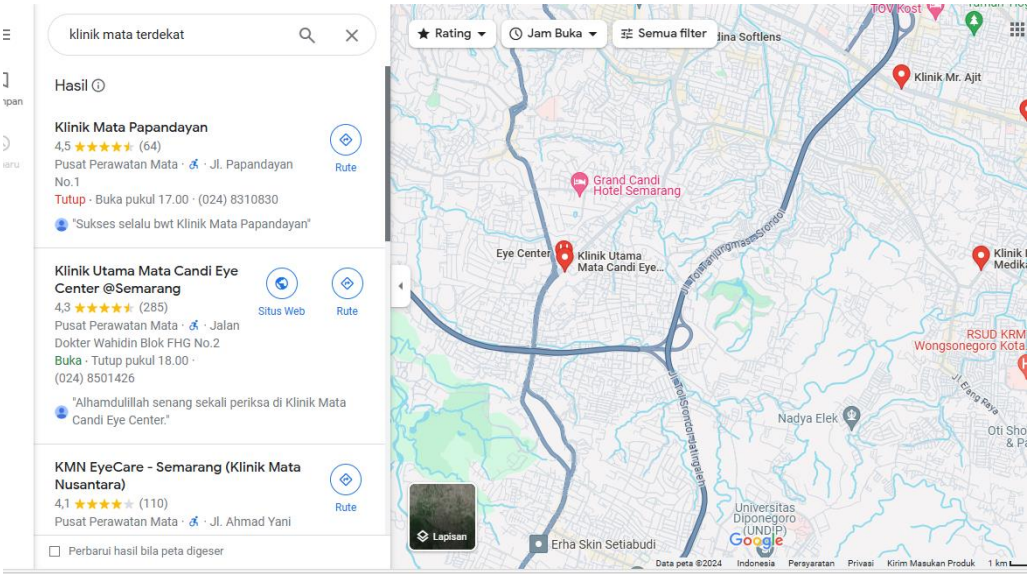
Jika nilai variabel link adalah "klinik/apotek".Maka sebuah tombol dengan kelas *CSS btn btn-custom custom-bg-warning font-semibold* dan teks "Rekomendasi klinik atau apotek".Atribut href pada elemen <a> mengarah ke *URL* https://www.google.com/maps/search/klinik /apotek/, yang akan membuka pencarian Google Maps untuk "klinik atau apotek".

https://www.google.com/maps/search/: Ini adalah *URL* dasar untuk pencarian di Google Maps.

{*keyword*}: Bagian ini diganti dengan kata kunci pencarian seperti "klinik", "apotek".

Hasil: Ketika pengguna mengklik tautan ini, Google Maps akan membuka dan menampilkan hasil pencarian untuk kata kunci yang diberikan di area geografis pengguna atau sesuai dengan lokasi yang ditentukan di pengaturan *browser*.

Berikut tampilan klinik atau apotek terdekat



7. Fitur ganti password

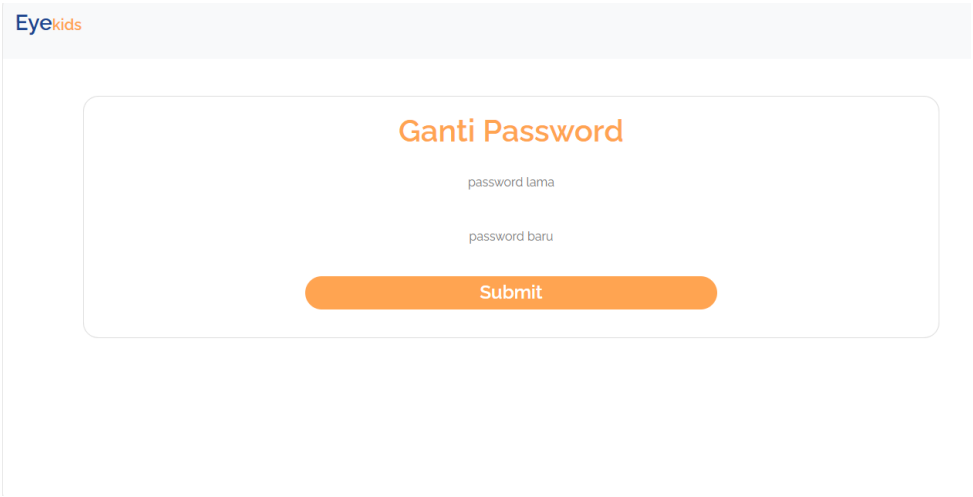
```python
@app.route('/ganti_password', methods=["POST"])
def ganti_password_post():
    # Coba ambil data dari JSON
    data = request.get_json()
    if data:
        password_lama = data.get('password_lama')
        password_baru = data.get('password_baru')
    else:
        # Jika tidak ada data JSON, ambil dari form
        password_lama = request.form.get('password_lama')
        password_baru = request.form.get('password_baru')
    user = user_datastore.find_user(username=session['full_name'])

    if user:
        if bcrypt.check_password_hash(user.password, password_lama):
            # Mengganti password lama dengan password baru
            user.password = bcrypt.generate_password_hash(password_baru).decode('utf-8')
            user_datastore.put(user)
            db.session.commit()
            return jsonify({"msg": "SUKSES"})
        else:
            return jsonify({"msg": "Password lama salah"})
    else:
        return jsonify({"msg": "user tidak ditemukan"})

```

Fungsi `ganti_password_post` akan dipanggil ketika pengguna mengirimkan permintaan *POST* ke *URL* `/ganti_password`.

1. Pertama, mencoba mengambil data dari *JSON* yang dikirim dalam permintaan. Jika data *JSON* tidak ada, mengambil data dari *form* yang dikirim melalui permintaan.
2. Mengambil data pengguna dari `pengguna_datastore` berdasarkan penggunaname yang disimpan dalam session.
3. Mengecek apakah pengguna ditemukan.
4. Mengecek apakah *password* lama yang diberikan cocok dengan *password* yang ada di *database*. Jika cocok, meng-*hash password* baru dan menyimpannya ke *database*, kemudian mengirimkan respons sukses. Jika *password* lama salah, mengirimkan *respons* yang sesuai.
5. Jika pengguna tidak ditemukan, mengirimkan *respons* yang sesuai.(pengguna tidak ditemukan

   Secara keseluruhan, fungsi ini memverifikasi *password* lama pengguna sebelum menggantinya dengan *password* baru yang di-*hash*, dan memberikan umpan balik yang sesuai kepada pengguna.

   Berikut tampilan ganti *password*



8. Halaman *login* admin

```python
@app.route('/login_admin/proses', methods=['POST'])
def proses_admin():
    username = request.json['username']
    password = request.json['password']
    user = user_datastore.find_user(username=username)
    if user:
        if 'admin' in [role.name for role in user.roles]:
            if bcrypt.check_password_hash(user.password, password):
                access_token = create_access_token(identity=username)

                session['jwt_token'] = access_token
                session['role'] = "admin"
                session['full_name'] = username
                flash('Login Berhasil')
                return jsonify(access_token=access_token)
            else:
                return jsonify({"msg": "password salah"}), 401
        else:
            return jsonify({"msg": "User is not an admin"}), 403
    else:
        return jsonify({"msg": "username salah"}), 404
```

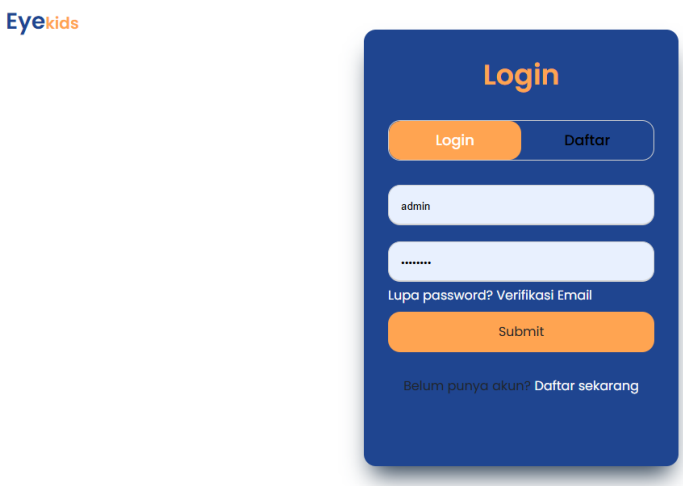
Fungsi proses_admin didefinisikan untuk menangani permintaan *POST* pada *route* /login_admin/proses. Pertama-tama, fungsi ini mengambil data *username* dan *password* dari *JSON* dalam permintaan *POST* menggunakan *request.json*['*username*'] dan *request.json*['*password*']. Setelah itu, fungsi mencari pengguna di pengguna_datastore berdasarkan penggunaname yang diberikan dengan memanggil

pengguna_*datastore.find*_pengguna(*username=username*). Jika pengguna ditemukan, fungsi akan melanjutkan untuk memeriksa apakah pengguna tersebut memiliki peran 'admin'. Hal ini dilakukan dengan memeriksa apakah 'admin' ada dalam daftar nama peran yang dimiliki oleh pengguna tersebut (*if 'admin' in* [*role.name for role in pengguna.roles*]).

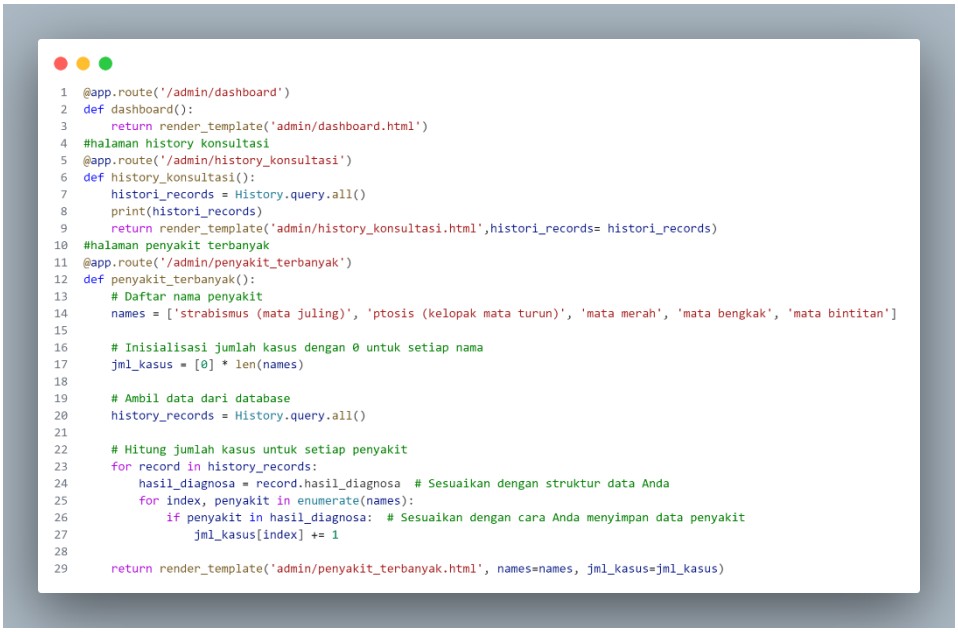
Jika pengguna memiliki peran 'admin', fungsi kemudian memvalidasi *password* yang diberikan dengan mencocokkannya dengan *hash password* yang disimpan menggunakan *bcrypt.check_password_hash*(pengguna.*password*, *password*). Jika *password* cocok, fungsi akan membuat token akses menggunakan *create_access_token* dengan *username* sebagai identitas. Token akses ini kemudian disimpan dalam sesi bersama dengan informasi peran pengguna (*role*) dan nama lengkap pengguna (*full_name*). Pesan *flash* 'Login Berhasil' dikirim untuk ditampilkan di antarmuka pengguna, dan token akses dikembalikan dalam format *JSON* sebagai respon berhasil.

Namun, jika *password* yang diberikan tidak cocok dengan *hash password* yang disimpan, fungsi akan mengembalikan respon *JSON* dengan pesan "*password* salah" dan status kode 401. Jika pengguna yang dicari tidak ditemukan di pengguna_datastore, fungsi akan mengembalikan respon *JSON* dengan pesan "penggunaname salah" dan status kode 404. Jika pengguna ditemukan tetapi tidak memiliki peran 'admin', fungsi akan mengembalikan respon *JSON* dengan pesan "Pengguna is not an admin" dan status kode 403. Dengan demikian, fungsi proses_admin memastikan bahwa hanya pengguna dengan peran 'admin' yang memiliki akses dan bahwa kredensial yang diberikan adalah benar sebelu m memberikan akses lebih lanjut.

Berikut tampilan login admin



9. Halaman Dashboard admin



```python
@app.route('/admin/dashboard')
def dashboard():
    return render_template('admin/dashboard.html')
#halaman history konsultasi
@app.route('/admin/history_konsultasi')
def history_konsultasi():
    histori_records = History.query.all()
    print(histori_records)
    return render_template('admin/history_konsultasi.html',histori_records= histori_records)
#halaman penyakit terbanyak
@app.route('/admin/penyakit_terbanyak')
def penyakit_terbanyak():
    # Daftar nama penyakit
    names = ['strabismus (mata juling)', 'ptosis (kelopak mata turun)', 'mata merah', 'mata bengkak', 'mata bintitan']

    # Inisialisasi jumlah kasus dengan 0 untuk setiap nama
    jml_kasus = [0] * len(names)

    # Ambil data dari database
    history_records = History.query.all()

    # Hitung jumlah kasus untuk setiap penyakit
    for record in history_records:
        hasil_diagnosa = record.hasil_diagnosa  # Sesuaikan dengan struktur data Anda
        for index, penyakit in enumerate(names):
            if penyakit in hasil_diagnosa:  # Sesuaikan dengan cara Anda menyimpan data penyakit
                jml_kasus[index] += 1

    return render_template('admin/penyakit_terbanyak.html', names=names, jml_kasus=jml_kasus)
```

Pertama, *route* /admin/*dashboard* didefinisikan dengan fungsi *dashboard()*, yang akan menghubungkan dengan template template HTML admin/*dashboard.html* ketika diakses.

Kedua, *route* /admin/*history*_konsultasi didefinisikan dengan fungsi *history*_konsultasi(), yang akan mengambil semua catatan riwayat konsultasi dari database melalui *History.query.all().* Data ini kemudian dicetak di konsol untuk tujuan debugging dan dikirim ke template admin/history_konsultasi.html untuk dirender di halaman web. Di template tersebut, variabel *history_records* akan digunakan untuk menampilkan data riwayat konsultasi yang telah diambil.

Ketiga, route /admin/penyakit_terbanyak didefinisikan dengan fungsi penyakit_terbanyak(), yang akan menghitung jumlah kasus setiap penyakit tertentu berdasarkan data riwayat konsultasi. Fungsi ini dimulai dengan mendefinisikan daftar names yang berisi nama-nama penyakit yang akan dihitung, seperti '*strabismus* (mata juling)', *'ptosis* (kelopak mata turun)', 'mata merah', 'mata bengkak', dan 'mata bintitan'. Selanjutnya, sebuah daftar jml_kasus diinisialisasi dengan nilai nol untuk setiap penyakit, bertujuan menyimpan jumlah kasus dari setiap penyakit tersebut.

Data riwayat konsultasi kemudian diambil dari database menggunakan *History.query.all().* Untuk setiap catatan riwayat konsultasi, fungsi akan memeriksa apakah hasil diagnosis mengandung nama penyakit yang didefinisikan dalam daftar names. Jika nama penyakit ditemukan dalam

hasil diagnosis, jumlah kasus untuk penyakit tersebut akan ditambahkan satu. Setelah perhitungan selesai, fungsi akan merender template admin/penyakit_terbanyak.html dengan variabel names dan jml_kasus, yang digunakan untuk menampilkan nama-nama penyakit dan jumlah kasusnya di halaman web.

Berikut tampilan dashboard admin

10. Halaman history admin

```python
@app.route('/admin/history_konsultasi/<id>')
def admin_hasil_diagnosa(id):
    if 'full_name' not in session:
        abort(403)  # Forbidden, user tidak terautentikasi

    # Query data history berdasarkan id dan username dari session
    history_record = History.query.filter_by(id=id).first()
    if not history_record:
        abort(404)  # Not found, data history tidak ditemukan

    # Pastikan hasil_diagnosa adalah dictionary
    hasil_diagnosa_str = history_record.hasil_diagnosa

    hasil_diagnosa = hasil_diagnosa_str.split(",")

    if hasil_diagnosa[-1] == '':
        hasil_diagnosa.pop()
    print(hasil_diagnosa)
    # Query semua rekomendasi
    rekomendasi_records = Rekomendasi.query.all()
    print(rekomendasi_records)
    rekomendasi_list = [record.serialize() for record in rekomendasi_records]
    print(rekomendasi_list)

    # Gabungkan rekomendasi yang relevan dengan hasil diagnosa
    rekomendasi_diagnosa = {}
    for penyakit in hasil_diagnosa:
        for rekomendasi in rekomendasi_list:
            if rekomendasi['nama'] == penyakit:
                rekomendasi_diagnosa[penyakit] = rekomendasi
    print(rekomendasi_diagnosa)
    diagnosa = {
        'nama_user': history_record.nama_user,
        'nama_anak': history_record.nama_anak,
        'usia_anak': history_record.usia_anak,
        'tanggal_konsultasi': history_record.tanggal_konsultasi,
        'file_deteksi': history_record.file_deteksi,
        'hasil_diagnosa': hasil_diagnosa,
        'rekomendasi_diagnosa': rekomendasi_diagnosa,
    }
    print(diagnosa)

    return render_template('user/hasil_diagnosa.html', diagnosa=diagnosa)
```

Kode di atas bertanggung jawab untuk menampilkan hasil Diagnosis detail dari riwayat konsultasi berdasarkan ID yang diberikan dalam *URL*.

fungsi akan mencoba mengambil catatan riwayat konsultasi dari database menggunakan *History.query.filter_by(id=id).first(),* yang mencari catatan

berdasarkan ID yang diberikan. Jika catatan tidak ditemukan, fungsi akan mengembalikan status *404 Not Found*, menandakan bahwa data riwayat yang diminta tidak tersedia.

Setelah catatan riwayat ditemukan, fungsi memastikan bahwa hasil_Diagnosis adalah string yang dipisahkan dengan koma, kemudian mengubahnya menjadi daftar menggunakan *split(",").* Jika elemen terakhir dari daftar kosong, elemen tersebut akan dihapus dengan *pop().* Daftar hasil diagnosis ini kemudian dicetak ke konsol untuk tujuan debugging.

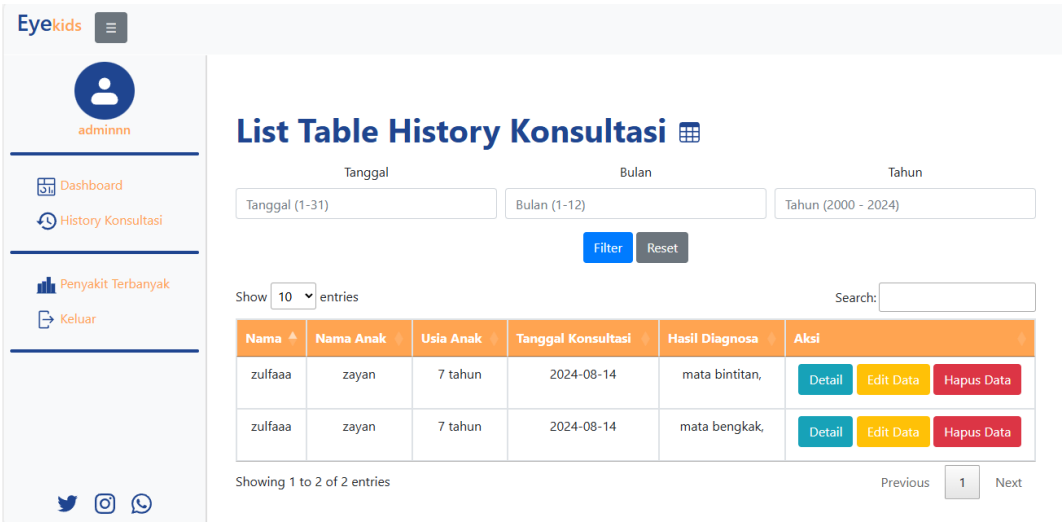
Fungsi selanjutnya mengambil semua catatan rekomendasi dari database dengan *rekomendasi.query.all*(), mengubah catatan-catatan ini menjadi daftar yang dapat digunakan dengan memanggil metode *serialize()* pada setiap catatan. Daftar rekomendasi ini juga dicetak ke konsol.

Untuk setiap penyakit dalam hasil diagnosis, fungsi akan mencari rekomendasi yang sesuai dalam daftar rekomendasi yang telah diambil. Jika ditemukan kecocokan, fungsi akan menambahkan rekomendasi tersebut ke dalam kamus rekomendasi_diagnosis dengan nama penyakit sebagai kunci.

Setelah proses penggabungan rekomendasi selesai, fungsi membuat sebuah kamus diagnosis yang berisi informasi lengkap tentang riwayat konsultasi, termasuk nama pengguna, nama anak, usia anak, tanggal konsultasi, file deteksi, hasil diagnosis, dan rekomendasi diagnosis. Kamus diagnosis ini dicetak ke konsol untuk memastikan bahwa semua data telah dikumpulkan dengan benar.

Terakhir, fungsi akan menghubungkan template *HTML* pengguna/hasil_Diagnosis.html dengan mengirimkan Diagnosis ke template tersebut, sehingga detail hasil diagnosis dapat ditampilkan di halaman web. Kode ini memastikan bahwa informasi yang relevan diambil dari database, diproses, dan ditampilkan kepada pengguna dengan cara yang terstruktur dan aman.

Berikut tampilan history admin



11. Halaman penyakit terbanyak admin

```python
@app.route('/admin/penyakit_terbanyak')
def penyakit_terbanyak():
    # Daftar nama penyakit
    names = ['strabismus (mata juling)', 'ptosis (kelopak mata turun)', 'mata merah', 'mata bengkak', 'mata bintitan']

    # Inisialisasi jumlah kasus dengan 0 untuk setiap nama
    jml_kasus = [0] * len(names)

    # Ambil data dari database
    history_records = History.query.all()

    # Hitung jumlah kasus untuk setiap penyakit
    for record in history_records:
        hasil_diagnosa = record.hasil_diagnosa  # Sesuaikan dengan struktur data Anda
        for index, penyakit in enumerate(names):
            if penyakit in hasil_diagnosa:  # Sesuaikan dengan cara Anda menyimpan data penyakit
                jml_kasus[index] += 1

    return render_template('admin/penyakit_terbanyak.html', names=names, jml_kasus=jml_kasus)
```

Kode ini merupakan *route* `/admin/penyakit_terbanyak` dibuat untuk menghasilkan halaman yang menampilkan informasi tentang penyakit-penyakit yang paling sering terjadi berdasarkan data dari database.

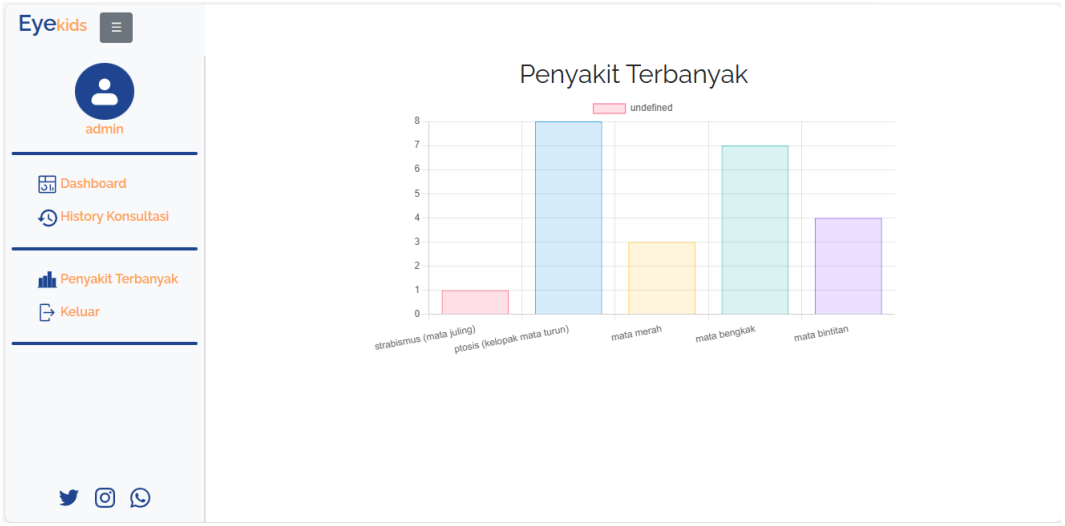
Pertama, kita memiliki daftar `names` yang berisi nama-nama penyakit seperti '*strabismus* (mata juling)', '*ptosis* (kelopak mata turun)', dan lain-lain. List ini akan digunakan untuk menentukan penyakit mana yang akan dihitung jumlah kasusnya.

Kemudian, `jml_kasus` diinisialisasi sebagai list dengan panjang yang sama dengan `names`, diatur untuk menyimpan jumlah kasus untuk setiap penyakit. Semua nilai di `jml_kasus` diatur ke 0 pada awalnya.

Data dari database diambil dengan menggunakan `History.query.all()`, yang asumsinya menggunakan *SQLAlchemy* untuk mengakses data. Setiap record dari `history_records` dievaluasi untuk `hasil_diagnosis`. Perulangan dilakukan untuk setiap record, dan untuk setiap penyakit dalam `names`, dicek apakah nama penyakit tersebut ada dalam `hasil_Diagnosis`. Jika ya, jumlah kasus untuk penyakit tersebut diinkrementasi.

Terakhir, fungsi `render_template()` digunakan untuk merender halaman *HTML* `admin/penyakit_terbanyak.html`. Data `names` dan `jml_kasus` disediakan ke template ini untuk ditampilkan kepada pengguna.

Berikut tampilan penyakit terbanyak admin



Lampiran 6 Sertifikat HKI

REPUBLIK INDONESIA
KEMENTERIAN HUKUM DAN HAK ASASI MANUSIA

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	:	EC00202466114, 16 Juli 2024
Pencipta		
Nama	:	**Zulfatun Nisa, Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng. dkk**
Alamat	:	Jln.Akasia Rt 38/Rw 08 No 33 Desa Tembok Luwung Kec.Adiwerna Kab.Tegal Provinsi Jawa Tengah 52194, Adiwerna, 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, Tegal, Jawa Tengah 52142
Kewarganegaraan	:	Indonesia
Jenis Ciptaan	:	**Program Komputer**
Judul Ciptaan	:	**SISTEM DIAGNOSIS PENYAKIT MATA PADA ANAK MENGGUNAKAN ALGORITMA YOLO V8**
Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia	:	16 Juli 2024, di Tegal
Jangka waktu pelindungan	:	Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.
Nomor pencatatan	:	000641469

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 DAN HAK ASASI MANUSIA
DIREKTUR JENDERAL KEKAYAAN INTELEKTUAL
u.b
Direktur Hak Cipta dan Desain Industri

IGNATIUS M.T. SILALAHI
NIP. 196812301996031001

Disclaimer:
Dalam hal pemohon memberikan keterangan tidak sesuai dengan surat pernyataan, Menteri berwenang untuk mencabut surat pencatatan permohonan.

LAMPIRAN PENCIPTA

No	Nama	Alamat
1	Zulfatun Nisa	Jln.Akasia Rt 38/Rw 08 No 33 Desa Tembok Luwung Kec.Adiwerna Kab.Tegal Provinsi Jawa Tengah 52194, Adiwerna, Tegal
2	Ir. Ginanjar Wiro Sasmito, M.Kom., IPM., ASEAN Eng.	Desa Kluwut RT/RW 003/002 Kec.Bulakamba Kab. Brebes Provinsi Jawa Tengah 52253, Bulakamba, Brebes
3	Ardi Susanto ,S.Kom.,M.Cs.	Margadana Kota Tegal Provinsi Jawa Tengah 52143, Margadana, Tegal

Lampiran 7 Lembar Bimbingan Skripsi

SARJANA TERAPAN TEKNIK INFORMATIKA
POKITEKNIK HARAPAN BERSAMA

LEMBAR BIMBINGAN SKRIPSI

Nama : Zulfatun Nisa
Nim : 20090143
No. Ponsel : 082137704652
Judul TA : SISTEM DIAGNOSA PENYAKIT MATA PADA ANAK MENGGUNAKAN ALGORITMA YOLO V8
Dosen Pembimbing I : Ir. Ginanjar Wiro Sasmito, M.Kom.

No	Tanggal	Pemeriksaan	Perbaikan yang perlu dilakukan	Paraf Pembimbing
1.	18/29/03	Konsep	Revisi	
2	2/29/4	UI/UX.	perbaiki	
3	22/29/4	UI/UX	oke.	
4.	17/29/05	Dataset	oke. Lanjut	
5	2/29/7	Aplikasi	- Dataset ditambah agar akurasi meningkat - Tampilan di rapihkan. - Diagnosis dari foto.	

SARJANA TERAPAN TEKNIK INFORMATIKA
POKITEKNIK HARAPAN BERSAMA

	9 Juli	Aplikasi	- Dataset - 6MAP. (Lokasi terdekat). - Verifikasi email/WA / SMS	f.
	17 Juli	Aplikasi Laporan	- Akurasi revisi	f.
	18 Juli 24	Laporan Siap ujian	oke	f. f

SARJANA TERAPAN TEKNIK INFORMATIKA
POKITEKNIK HARAPAN BERSAMA

Tegal, Juli 2024
Dosen Pembimbing I

Ir. Ginanjar Wiro Sasmito, M.Kom.
NIPY. 10.007.032

SARJANA TERAPAN TEKNIK INFORMATIKA
POKITEKNIK HARAPAN BERSAMA

LEMBAR BIMBINGAN SKRIPSI

Nama : Zulfatun Nisa
Nim : 20090143
No. Ponsel : 082137704652
Judul TA : SISTEM DIAGNOSA PENYAKIT MATA PADA ANAK MENGGUNAKAN ALGORITMA YOLO V8
Dosen Pembimbing II : Ardi Susanto, S.Kom., M.Cs.

No	Tanggal	Pemeriksaan	Perbaikan yang perlu dilakukan	Paraf Pembimbing
1.	7-4-2024	UI/UX.	- Pembuatan model ~~- Aplikasi~~	
2.	26-4-2024	~~-~~ Sudah dimodel	- pembuatan website	
3.	26-6-2024	- Aplikasi	- Rapihkan tampilan - selesaikan	
4		Aplikasi	selesaikan	
5		Aplikasi	selesaikan	
6.		Aplikasi	selesaikan	
7.		Aplikas	selesaikan	
8.		Aplikasi	selesaikan	