

LAMPIRAN

Lampiran 1. Surat Balasan Observasi

SURAT KESEPAKATAN BIMBINGAN SKRIPSI

Kami yang bertanda tangan di bawah ini:

Pihak Pertama

Nama : Dhiya Shilmi Aulia
NIM : 21090044
Program Studi : Sarjana Terapan Teknik Informatika

Pihak Kedua

Nama : Ir. Ginanjar Wiro Sasmito, M.Kom. IPM., ASEAN Eng
Status : Dosen Tetap
NIDN : 0613028601
Jabatan Fungsional : Lektor Kepala
Pangkat/Golongan : Penata Tk. I / III d

Pada hari ini Kamis tanggal 6 Maret 2025 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I Skripsi Pihak Pertama dengan syarat Pihak Pertama wajib menyelesaikan bimbingan Skripsi minimal 8 kali kepada Pihak Kedua. Jika melebihi batas waktu tersebut maka, Pihak Kedua berhak untuk tidak melanjutkan proses bimbingan skripsi. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

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

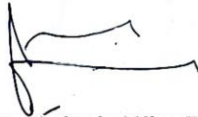
Tegal, 6 Maret 2025

Pihak Pertama,



Dhiya Shilmi Aulia

Pihak Kedua,



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

Mengetahui,
Ka. Prodi Sarjana Terapan Teknik Informatika



Dyah Aprianti, S.Ta, M.Kom.
NIPY-09/015.225

Lampiran 2 Surat Izin Observasi



POLITEKNIK HARAPAN BERSAMA
The True Vocational Campus

Sarjana Terapan Teknik Informatika

Nomor : 46.03/TI.PHB/V/2025
Lampiran : -
Hal : Permohonan Observasi
Kepada : Kepala Dinas Pariwisata Kab. Brebes
Yth. Di Tempat

Dengan hormat, mahasiswa dengan identitas berikut ini:

nama : Dhiya Shilmi Aulia
NIM : 21090044
prodi : Sarjana Terapan Teknik Informatika

Bermaksud melakukan penelitian untuk keperluan Skripsi dengan judul "**Sistem Rekomendasi Wisata Kabupaten Brebes Berbasis Web dengan Metode KNN**". Kami memohon Bapak/Ibu memberikan izin kepada mahasiswa yang bersangkutan agar dapat memperoleh data, keterangan, dan bahan yang diperlukan.

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

Tegal, 22 Mei 2025

Ka. Prodi S.T. Teknik Informatika,

Dyan Aprilia, S.T., M.Kom

NIPY : 09.015.225

Lampiran 3 Surat Kesepakatan Bimbingan

1. Surat Kesepakatan Bimbingan Pembimbing I

SURAT KESEPAKATAN BIMBINGAN SKRIPSI

Kami yang bertanda tangan di bawah ini:

Pihak Pertama

Nama : Dhiya Shilmi Aulia
NIM : 21090044
Program Studi : Sarjana Terapan Teknik Informatika

Pihak Kedua

Nama : Ir. Ginanjar Wiro Sasmito, M.Kom. IPM., ASEAN Eng
Status : Dosen Tetap
NIDN : 0613028601
Jabatan Fungsional : Lektor Kepala
Pangkat/Golongan : Penata Tk.I / III d

Pada hari ini Kamis tanggal 6 Maret 2025 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I Skripsi Pihak Pertama dengan syarat Pihak Pertama wajib menyelesaikan bimbingan Skripsi minimal 8 kali kepada Pihak Kedua. Jika melebihi batas waktu tersebut maka, Pihak Kedua berhak untuk tidak melanjutkan proses bimbingan skripsi. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

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

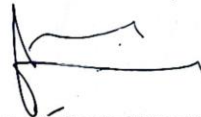
Tegal, 6 Maret 2025

Pihak Pertama,



Dhiya Shilmi Aulia

Pihak Kedua,



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

Mengetahui,
Ka. Prodi Sarjana Terapan Teknik Informatika



Dyah Aprianti S.T., M.Kom.
NIPY-09.015.225

2. Surat Kesepakatan Bimbingan Pembimbing II

SURAT KESEPAKATAN BIMBINGAN SKRIPSI

Kami yang bertanda tangan di bawah ini:

Pihak Pertama

Nama : Dhiya Shilmi Aulia
NIM : 21090044
Program Studi : Sarjana Terapan Teknik Informatika

Pihak Kedua

Nama : Dwi Intan Af'idah, S.T., M.Kom.
Status : Dosen
NIDN : 062008903
Jabatan Fungsional : Lektor
Pangkat/Golongan : Penata, III/c

Pada hari ini Kamis tanggal 6 Maret 2025 telah terjadi sebuah kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing II Skripsi Pihak Pertama dengan syarat adanya peningkatan progres dari proyek skripsi yang dilaporkan setiap satu pekan. Jika Pihak Pertama tidak memenuhi syarat tersebut, maka Pihak Kedua berhak memutuskan kesepakatan ini. Adapun waktu dan tempat pelaksanaan disepakati antar pihak.

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

Tegal, 6 Maret 2025

Pihak Pertama,



Dhiya Shilmi Aulia

Pihak Kedua,



Dwi Intan Af'idah, S.T., M.Kom

Mengetahui,
Ka. Prodi Sarjana Terapan Teknik Informatika



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

Lampiran 4 Lembar Bimbingan

1. Lembar Bimbingan Pembimbing I



**D IV TEKNIK INFORMATIKA
POLITEKNIK HARAPAN BERSAMA**

LEMBAR BIMBINGAN TUGAS AKHIR

Nama : Dhiya Shilmi Aulia
NIM : 21090044
No. Ponsel : 0895380217759
Judul TA : SISEM REKOMENDASI WISATA KABUPATEN
BREBES BERBASIS TRACKING MAPS DENGAN
METODE KNN
Dosen Pembimbing I : Ir. Ginanjar Wiro Sasmito, M.Kom. IPM., ASEAN Eng

No	Tanggal	Pemeriksaan	Perbaikan Yang Perlu Dilakukan	Paraf Pembimbing
1	24/25 9	konsep	rekomendasi	f.
2	23/25 5	produk	lanjut, rekomendasi system.	f.
3	20/25 6	produk	perhitungan manual.	f.
4	26/25 6	produk	Ok -	f.

11/25 7	Laporan	revisi	f
14/25 7	Laporan	revisi	f
	Laporan	oke	f
	fiyap	ujian	f

2. Lembar Bimbingan Pembimbing 2



**D IV TEKNIK INFORMATIKA
POLITEKNIK HARAPAN BERSAMA**

LEMBAR BIMBINGAN TUGAS AKHIR

Nama : Dhiya Shilmi Aulia
 NIM : 21090044
 No. Ponsel : 0895380217759
 Judul TA : SISEM REKOMENDASI WISATA KABUPATEN
 BREBES BERBASIS TRACKING MAPS DENGAN
 METODE KNN
 Dosen Pembimbing II : Dwi Intan Af'idah, S.T.,M.Kom

No	Tanggal	Pemeriksaan	Perbaikan Yang Perlu Dilakukan	Paraf Pembimbing
1.	25/4-2015	Konsep aplikasi	Partikan pemodelan sistem rekomendasi sesuai	<i>Dwi Intan Af'idah</i>
2.	23/5-2015	Sistem rekomendasi	Segera selesaikan sistem rekomendasi dengan tema KNN	<i>Dwi Intan Af'idah</i>
3.	2/6-2015	Sistem rekomendasi	Datasel dilengkapi, buang pemodelan, lalu diimplementasikan	<i>Dwi Intan Af'idah</i>
4.	13/6-2015	Sistem rekomendasi	-tambahkan evaluasi model KNN -implementasi model & detail wisata	<i>Dwi Intan Af'idah</i>

- search
 - filter by kategori

5.	20/6-2025	Aplikasi	- Aplikasi sudah cukup - Lanjut pengajuan hak cipta - Laporan	Dwi Afid
6.	24/6-2025	Draft hak cipta	- Lanjutan ke Laporan	Dwi Afid
7.	10/7-2025	Laporan	- Sesuaikan dengan catatan	Dwi Afid
8.	15/7-2025	Laporan	- sesuaikan dg catatan	Dwi Afid
9.	16/7-2025	Laporan	Acc	Dwi Afid

Tegal, 16 Juli 2025
Pembimbing II,



Dwi Intan Af'idah, S.T.,M.Kom
NIPY. 11.020.470

Lampiran 5 Surat Pengajuan HKI

1. Surat Pengalihan HKI

SURAT PENGALIHAN HAK CIPTA

Yang bertanda tangan di bawah ini :

1. Nama : Dhiya Shilmi Aulia
Kewarganegaraan : Indonesia
Alamat : Jl. Cendana V No. 14 Perum Dedy Jaya RT 003 RW 014. Kecamatan Brebes, Kabupaten Brebes, Provinsi Jawa Tengah
2. Nama : Ir. Ginanjar Wiro Sasmito, S.Kom., M.Kom
Kewarganegaraan : Indonesia
Alamat : Jl. Raya Kluwut Timur No. 24, Kecamatan Bulakamba, Kabupaten Brebes.
3. Nama : Dwi Intan Af'idah, S.T., M.Kom.
Kewarganegaraan : Indonesia
Alamat : Grinting RT03/RW02, Kecamatan Bulakamba, Kabupaten Brebes.

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 Tegal

Adalah **Pihak II** selaku Pemegang Hak Cipta berupa Program Komputer dengan judul "Aplikasi Sistem Rekomendasi Wisata Brebes Berbasis Website dengan Metode K-Nearest Neighbors" 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, 26 Juni 2025

Pemegang Hak Cipta
Kepala P3M



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

Pencipta



(Dhiya Shilmi Aulia)



(Ir. Ginanjar Wiro Sasmito, S.Kom., M.Kom)



(Dwi Intan Af'idah, S.T., M.Kom.)

2. Surat Pernyataan HKI

SURAT PERNYATAAN

Yang bertanda tangan di bawah ini, pemegang hak cipta:

1. Nama : Dhiya Shilmi Aulia
Kewarganegaraan : Indonesia
Alamat : Jl. Cendana V No. 14 Perum Dedy Jaya RT 003 RW 014
Kecamatan Brebes, Kabupaten Brebes, Provinsi Jawa Tengah
2. Nama : Ir. Ginanjar Wiro Sasmito, S.Kom., M.Kom
Kewarganegaraan : Indonesia
Alamat : Jln. Raya Kluwut Timur No. 24, Kecamatan Bulakamba, Kabupaten Brebes.
3. Nama : Dwi Intan Afidah, S.T., M.Kom.
Kewarganegaraan : Indonesia
Alamat : Grinting RT03/RW02, Kecamatan Bulakamba, Kabupaten Brebes.

Dengan ini menyatakan bahwa:

1. Karya Cipta yang saya mohonkan:
Berupa : Program Komputer
Berjudul : Aplikasi Sistem Rekomendasi Wisata Brebes berbasis Website
dengan Metode K-Nearest Neighbors.
 - 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 sebagaimana mestinya.

Tegal, 16 Juni 2025



(Dhiya Shilmi Aulia)
Pemegang Hak Cipta *

(Ir. Ginanjar Wiro Sasmito, S.Kom., M.Kom)
Pemegang Hak Cipta *

(Dwijanten Aridah, S.T., M.Kom.)
Pemegang Hak Cipta *

* Semua pemegang hak cipta agar menandatangani di atas materai.

REPUBLIK INDONESIA
KEMENTERIAN HUKUM

SURAT PENCATATAN CIPTAAN

Dalam rangka perlindungan 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 : EC002025078506, 30 Juni 2025

Pencipta

Nama : **DHIYA SHILMI AULIA, Ir.Ginanjari Sasmito, S.Kom., M.Kom dkk**

Alamat : **JL. CENDANA V NO 14 PERUM DEDY JAYA RT 003 RW 014
KELURAHAN PASARBATANG DESA PASARBATANG, Brebes,
Kab. Brebes, Jawa Tengah, 52211**

Kewarganegaraan : Indonesia

Pemegang Hak Cipta

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

Alamat : **Jalan Mataram No. 9, Pesurungan Lor, Kecamatan Margadana,
Margadana, Kota Tegal, Jawa Tengah, 52142**

Kewarganegaraan : Indonesia

Jenis Ciptaan : **Program Komputer**

Judul Ciptaan : **SISTEM REKOMENDASI WISATA KABUPATEN BREBES
BERBASIS TRACKING MAPS DENGAN METODE KNN**

Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia : 30 Juni 2025, di Kota Tegal

Jangka waktu perlindungan : Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.

Nomor Pencatatan : 000918767

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

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 diterbitkan 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 PENCIPTA

No	Nama	Alamat
1	DHIYA SHILMI AULIA	JL. CENDANA V NO 14 PERUM DEDY JAYA RT 003 RW 014 KELURAHAN PASARBATANG DESA PASARBATANG Brebes, Kab. Brebes
2	Ir.Ginanjari Sasmito, S.Kom., M.Kom	Jl. Raya Kluwut Timur No. 24 RT.003/002 Kabupaten Brebes. Bulakamba, Kab. Brebes
3	Dwi Intan Afidah, S.T., M.Kom.	Grinting RT03/RW02, Kec. Bulakamba, Kab. Brebes Bulakamba, Kab. Brebes



REKOMWISATA.



MANUAL BOOK

SISTEM REKOMENDASI WISATA KABUPATEN BREBES BERBASIS TRACKING MAPS DENGAN METODE KNN

Oleh:

Dhiya Shilmi Aulia

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

Dwi Intan Af'idah, S.T., M.Kom

1. Pendahuluan

1.1 Tujuan Pembuatan Dokumen

Dokumen “Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode KNN” ini dibuat untuk tujuan sebagai berikut:

1. Memberikan panduan bagi pengguna dalam memahami dan mengoperasikan sistem rekomendasi wisata Kabupaten Brebes berbasis tracking maps dengan metode K-Nearest Neighbors (KNN).
2. Menjelaskan Langkah-langkah teknis dan prosedur yang diperlukan untuk mengakses, menggunakan, dan memanfaatkan fitur-fitur sistem guna mendapatkan rekomendasi wisata sesuai kebutuhan dan preferensi pengguna.
3. Menyediakan dokumentasi lengkap bagi pengembang maupun pihak terkait dalam melakukan pemeriksaan, pengembangan lanjutan, atau integrasi sistem dengan teknologi lainnya.

1.2 Deskripsi umum Sistem

Sistem Rekomendasi Wisata Kabupaten Brebes berbasis tracking maps dengan metode KNN merupakan sebuah aplikasi yang dirancang untuk membantu wisatawan dalam menemukan dan menentukan destinasi wisata di kabupaten brebes sesuai dengan kebutuhan dan preferensi masing-masing. Aplikasi ini memanfaatkan teknologi tracking maps

guna menampilkan Lokasi tempat wisata secara interaktif dan memberikan navigasi yang jelas bagi pengguna.

1.3 Deskripsi umum Dokumen

Dokumen ini disusun sebagai panduan bagi pengguna dalam memahami dan mengoperasikan Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode *K-Nearest Neighbors (KNN)*. Dokumen ini menjabarkan struktur dan tata cara penggunaan sistem mulai dari Gambaran umum, Langkah teknis, hingga contoh penerapan untuk memaksimalkan pemanfaatannya.

1. BAB I

Berisi gambaan umum sistem, tujuan pembuatan dokumen, deskripsi sistem, dan ruang lingkup dari sistem yang dikembangkan, guna memberikan pemahaman awal bagi pembaca mengenai kebutuhan dan manfaat dari sistem ini.

2. BAB II

Menjelaskan tata cara atau Langkah-langkah teknis dalam mengoperasikan siste, termasuk prosedur *Login*, navigasi menu, pengaturan fitur, dan pemanfaatan sistem untuk mendapatkan rekomendasi tempat wisata sesuai kebutuhan pengguna.

3. BAB III

Berisi ringkasan dari isi dokumen, Kesimpulan mengenai penerapan sistem, serta saran-saran untuk pengembangan atau peningkatan fitur sistem di masa yang akan datang.

2. PERANGKAT YANG DIBUTUHKAN

2.1 Perangkat Lunak

Untuk dapat Mengembangkan, menjalankan, dan memelihara sistem ini diperlukan perangkat lunak sebagai berikut:

1. Sistem Operasi : Menggunakan *Windows*.
2. *Web Browser* (*Google Chrome, Microsoft Edge, dll*).

2.2 Perangkat Keras

Untuk dapat mengoperasikan dan memanfaatkan sistem ini dengan optiman diperlukan spesifikasi perangkat keras sebagai berikut:

1. Laptop maupun Komputer PC.
2. *Mouse*.
3. *Keyboard*.

3. MENU DAN CARA PENGGUNA

3.1 Struktur Menu

Adapun struktur halaman pada Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode KNN adalah sebagai berikut:

1. Halaman Utama

Pada halaman utama ini, sistem Rekomendasi Wisata Kabupaten Brebes menyajikan tampilan awal yang terlihat pertama kali oleh

pengunjung saat mengakses situs ada beberapa menu yaitu beranda ini berfungsi sebagai titik awal bagi pengguna untuk menjelajahi berbagai wisata di Kabupaten Brebes, menu Sejarah yang berfungsi untuk sumber informasi Sejarah yang ada di Kabupaten Brebes, dan menu rekomendasi berfungsi sebagai fitur inti dari sistem yang membantu pengguna menemukan tempat-tempat wisata di Kabupaten Brebes.

2. Halaman Utama *User*

Pada halaman utama *User* ada beberapa menu yang tersedia adalah sebagai berikut:

- a. Beranda
- b. Sejarah
- c. Rekomendasi

3. Halaman Utama Admin

Pada halaman utama admin ada beberapa menu yang tersedia adalah sebagai berikut:

- a. Dashboard
- b. Data Wisata
- c. Tambah Data

3.2 Pengguna

Pada bagian ini akan dijelaskan mengenai menu-menu yang tersedia di Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode KNN sebagai berikut:

3.2.1 Cara Membuka Sistem

Untuk menjalankan *website* Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode KNN adalah sebagai berikut:

1. Bukalah *website* RekomWisata melalui *web* browser yang tersedia seperti (*Google Chrome, Microsoft Edge, dll*).
2. Di tampilan utama *website* RekomWisata pada halaman terdapat menu beranda, sejarah, dan rekomendasi. Adapun tampilan utama dapat dilihat pada gambar 1.



Gambar 1 Tampilan Utama

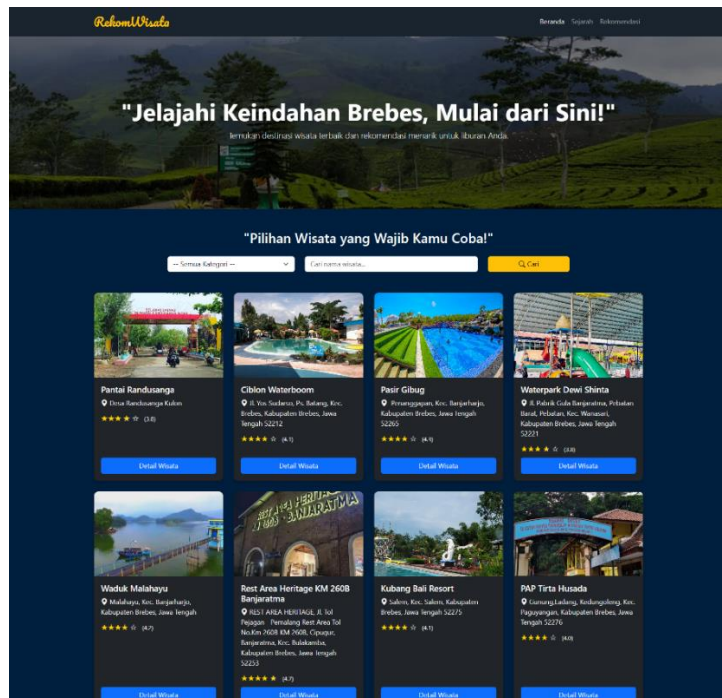
3.2.2 Halaman Utama

Pada *website* Sistem Rekomendasi Wisata Kabupaten Brebes berbasis Tracking Maps dengan Metode KNN terdapat halaman utama yang terdiri dari berbagai menu sebagai berikut:

1. Menu Beranda

Langkah-langkah Pengguna:

1. Pengguna dapat melihat daftar tempat wisata.
 2. Pengguna menekan tombol **“Cari”** untuk menampilkan daftar tempat wisata sesuai dengan pilihan kategori atau kata kunci.
 3. Pengguna melihat **daftar tempat wisata** yang ditampilkan, lengkap dengan gambar, alamat, dan rating.
 4. Lalu pengguna juga dapat menekan tombol **“Detail Wisata”** pada salah satu destinasi untuk melihat informasi lebih lengkapnya.
 5. Pengguna juga dapat Kembali ke halaman sebelumnya atau melakukan pencarian lainnya sesuai kebutuhannya.
- Adapun tampilan halaman beranda dapat dilihat pada gambar 2.



Gambar 2 Tampilan Beranda

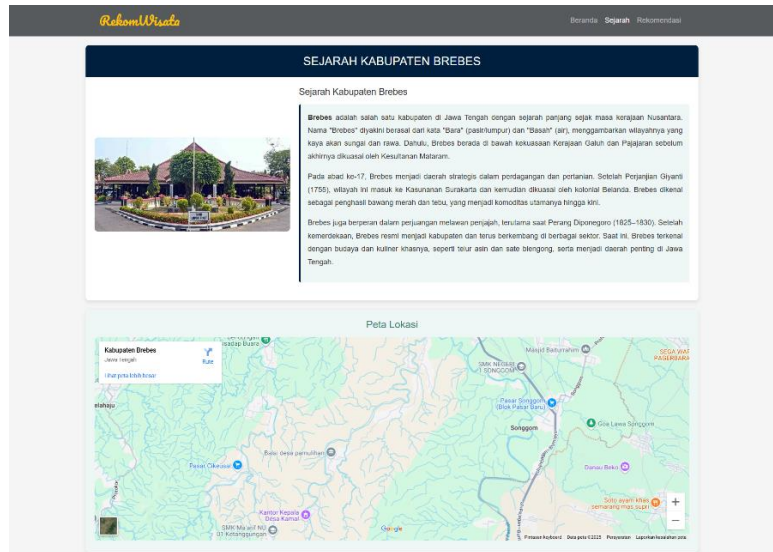
2. Menu Sejarah

Menu sejarah dalam sistem ini digunakan untuk memberikan informasi lengkap terkait Sejarah Kabupaten Brebes, agar pengunjung dapat memahami nilai historis dan konteks daerah sebelum menjelajahi berbagai tempat wisata.

Langkah-langkah Pengguna:

1. Pengguna dapat mengeklik menu **“Sejarah”** di bagian navigasi atas untuk masuk kehalaman sejarahnya.
2. Pengguna juga dapat membaca **deskripsi** yang menjelaskan perkembangan Kabupaten Brebes dari masa Kerajaan hingga era modern.

3. Dan langkah selanjutnya pengguna dapat melihat **peta lokasi** yang disediakan untuk mendapatkan gambaran geografis wilayah Kabupaten Brebes. Adapun tampilan halaman sejarah dapat dilihat pada gambar 3.



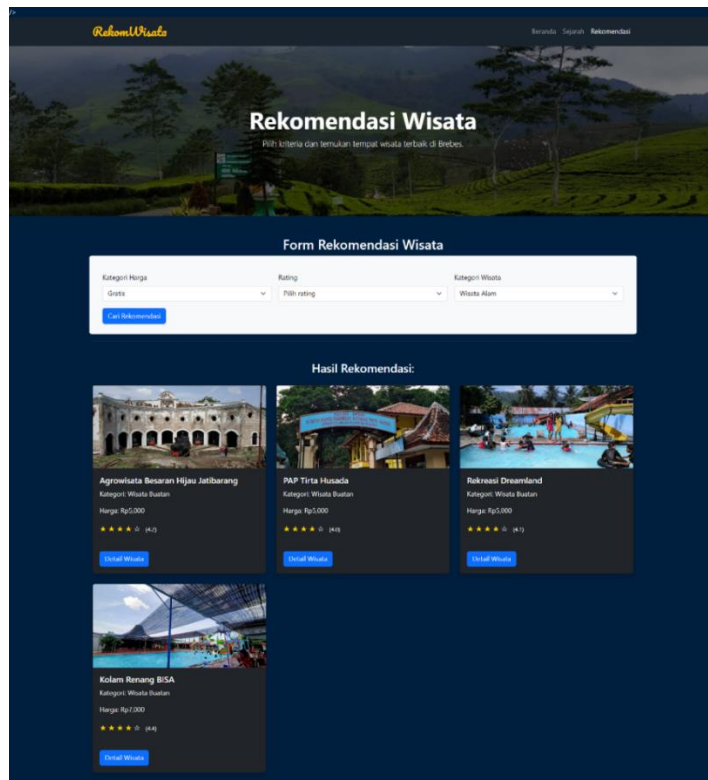
Gambar 3 Tampilan Sejarah

3. Menu Rekomendasi

Langkah-langkah pengguna:

1. Pengguna dapat mengeklik menu **“Rekomendasi”** dibagian navigasi atas untuk membuka halaman rekomendasi wisata.
2. Pengguna mengisi form filter rekomendasi wisata berdasarkan:
 - Kategori Harga(Gratis(0) ,Mahal(50k) ,Sedang(25k), Murah(10k)).

- Rating (4 bintang keatas)
 - Kategori Wisata (Wisata alam, buatan. religi)
3. Pengguna juga setelah mengisi form filter rekomendasi pengguna juga mengeklik tombol **“Cari Rekomendasi”** untuk memproses pencarian berdasarkan filter yang dipilih.
4. Sistem akan menampilkan hasil rekomendasi wisata sesuai dengan kriteria, termasuk:
- Nama Wisata
 - Kategori Wisata
 - Harga Tiket Wisata
 - Ranting Bintang
 - Tombol **”DetailWisata”**
5. Pengguna juga dapat mengeklik tombol **“Detail Wisata”** untuk melihat informasi lengkapnya tentang wisata dan juga pengguna dapat mengeklik google maps untuk tujuan wisata tersebut. Adapaun tampilan halaman rekomendasi dapat dilihat pada gambar 4.



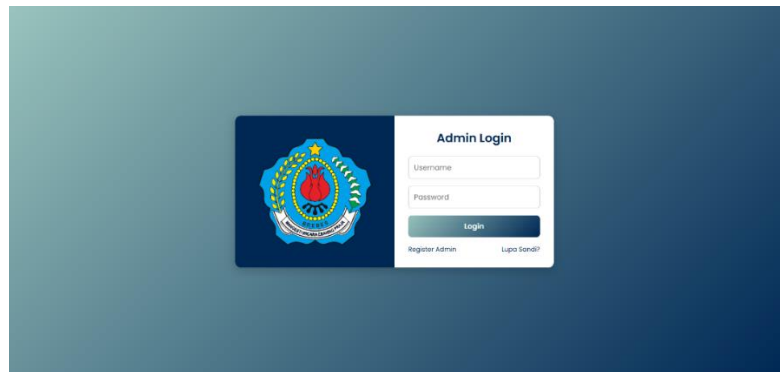
Gambar 4 Tampilan Rekomendasi

3.2.3 Halaman Admin *Login*

Langkah-langkah Pengguna Admin:

1. Admin membuka halam Admin *Login* melalui URL atau tautan yang disediakan (misalnya; <http://adminLogin>)
2. Admin memasukan *Username* dan Password ke dalam kolom yang tersedia.
3. Admin mengeklik tombol “*Login*” untuk masuk ke sistem admin.

4. Jika admin berhasil, admin akan diarahkan ke *Dashboard* admin untuk mengelola data wisata tersebut.
5. Jika admin belum memiliki akun admin, admin dapat mengeklik **"Register Admin"** untuk mendaftar sebagai admin baru.
6. Jika lupa password, pengguna dapat mengeklik **"Lupa Sandi"** untuk memulai proses reset kata sandi. Adapaun tampilan halaman admin *Login* dapat dilihat pada gambar 5.



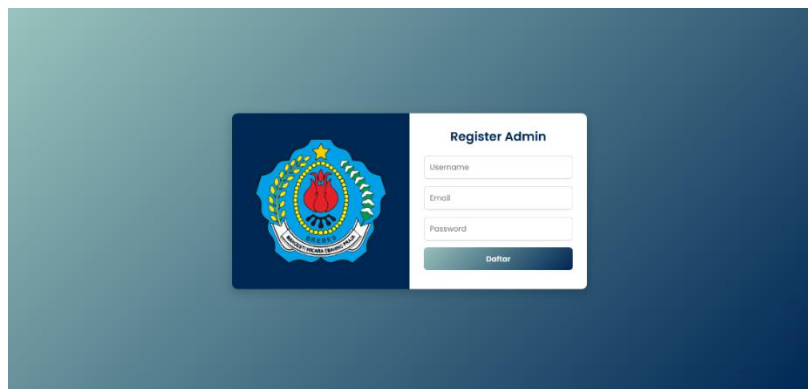
Gambar 5 Halaman *Login*

3.2.4 Halaman Registrasi

Langkah-langkah Pengguna:

1. Admin membuka halaman Register Admin melalui tautan halaman *Login* atau langsung melalui URL (<http://registeradmin>).

2. Admin mengisi kolom *Username* sesuai nama yang ingin digunakan sebagai akun admin.
3. Admin mengisi kolom **Email** yang aktif dan valid.
4. Admin mengisi kolom Password untuk keamanan akun.
5. Admin menekan tombol “**Daftar**” untuk menyelesaikan proses registrasi admin.
6. Setelah berhasil mendaftar, pengguna dapat diarahkan ke halaman *Login* untuk melakukan **Login sebagai admin**.
Adapaun tampilan halaman admin registrasi dapat dilihat pada gambar 6.



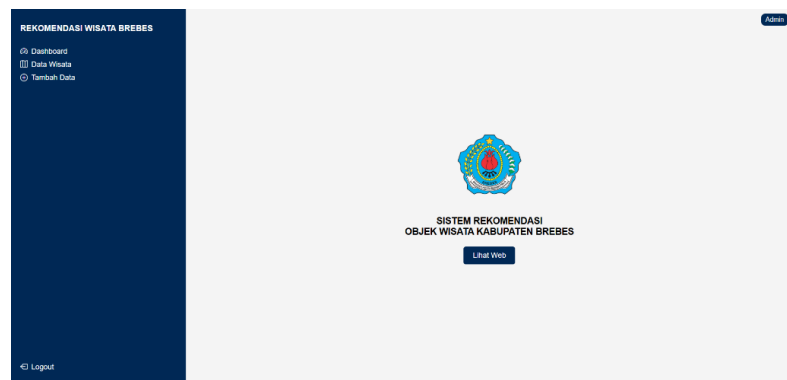
Gambar 6 Halaman Registrasi Admin

3.2.5 Halaman Dasbhoard Admin

Langkah-langkah Pengguna:

1. Admin dapat mengelola **data wisata** melalui tampilan *Dashboard* yang disediakan setelah berhasil *Login*.
2. Admin dapat memilih menu “**Data Wisata**” untuk melihat, mengedit, atau menghapus data wisata yang sudah ada.

3. Admin dapat memilih menu **“Tambah Data”** untuk menambahkan destinasi wisata baru ke dalam sistem.
4. Admin dapat keluar dari sistem dengan mengeklik tombol **“Logout”** yang berada di pojok kiri bawah *Dashboard*.
5. Halaman *Dashboard* ini berfungsi sebagai **pusat pengelolaan dan pengawasan** seluruh data wisata Kabupaten Brebes oleh admin. Adapun tampilan halaman admin dashboard dapat dilihat pada gambar 7.



Gambar 7 Halaman Dasbhoard Admin

3.2.6 Halaman Tambah Data

Langkah-langkah Pengguna:

1. Admin masuk ke menu **“Tambah Data”** melalui sidebar pada *Dashboard*.
2. Admin mengisi formulir data wisata, mulai dari:
 - Nama wisata
 - Alamat

- Jam operasional
- Status wisata
- Fasilitas
- Harga
- Form maps
- Rating
- From gambar

3. Setelah semua data lengkap, admin menekan tombol **“Simpan”** untuk menyimpan data wisata kedalam sistem. Adapaun tampilan halaman dapat dilihat pada tambah data gambar 8.

REKOMEDASI WISATA BREBES

- Dashboard
- Data Wisata
- Tambah Data

Tambah Data Tempat Wisata

Tambah Data

Nama Wisata
Masukkan nama wisata

Alamat
Masukkan alamat wisata

Jam Operasional
Contoh: 08.00 - 17.00

Status Wisata
Aktif

Deskripsi
Masukkan deskripsi wisata

Harga Tiket
Masukkan harga tiket

Unggah Gambar Utama
Choose File | No file chosen

Unggah Gambar Kedua
Choose File | No file chosen

Unggah Gambar Ketiga
Choose File | No file chosen

Link Google Maps
Masukkan link Google Maps

Jenis Wisata
Wisata Alam

Rating (1.0 - 5.0)
Masukkan rating (contoh: 4.5)

Latitude
Masukkan koordinat latitude

Longitude
Masukkan koordinat longitude

Fasilitas

- Kamar Mandi:** Ada
- Masjid:** Ada
- Cafe & Resto:** Ada
- Parkir Mobil:** Ada
- Parkir Motor:** Ada
- Warung:** Ada

Simpan

Gambar 8 Tampilan Tambah Data

3.2.7 Halaman Data Wisata

Langkah-langkah Pengguna:

1. Admin masuk ke menu **“Data Wisata”** untuk melihat daftar tempat wisata yang sudah terdaftar di sistem.
2. Admin dapat meninjau data tempat wisata yang ditampilkan, termasuk Nama Wisata dan Alamat dari setiap entri.
3. Admin dapat menekan tombol **“+ Tambah Data”** di bagian atas jika ingin menambahkan tempat wisata baru.

- Admin dapat menggunakan kolom **Pencarian** untuk mencari tempat wisata tertentu secara cepat.
- Data tempat wisata ditampilkan dalam bentuk **tabel dengan pagination**, sehingga admin dapat berpindah antar halaman untuk melihat seluruh data.
- Dengan fitur ini, admin dapat **mengelola dan memantau** seluruh informasi wisata yang sudah terinput dengan mudah dan efisien. Adapaun tampilan halaman data wisata dapat dilihat pada gambar 9.

No	Nama Wisata	Alamat
1	Pantai Randusanga	Desa Randusanga Kulon
2	Ciblon Waterboom	Jl. Yos Sudarso, Ps. Batang, Kec. Brebes, Kabupaten Brebes, Jawa Tengah 52212
3	Pastir Gribag	Penanggapan, Kec. Banjarharjo, Kabupaten Brebes, Jawa Tengah 62265
4	Waterpark Dewi Shinta	Jl. Pabrik Gula Banjaratma, Peboatan Baran, Kec. Wanasari, Kabupaten Brebes, Jawa Tengah 52221
5	Waduk Malahayu	Malahayu, Kec. Banjarharjo, Kabupaten Brebes, Jawa Tengah
6	Rest Area Heritage KM 260B Banjaratma	REST AREA HERITAGE, Jl. Tol Pejagan - Pemalang Rest Area Tol No.Km 260B KM 260B, Cipugur, Banjaratma, P
7	Kubang Bali Resort	Salem, Kec. Salem, Kabupaten Brebes, Jawa Tengah 52276
8	PAP Tirta Husada	GunungLadang, Kadungoleng, Kec. Paguyangan, Kabupaten Brebes, Jawa Tengah 52276
9	Hutan Mangrove	Pandansari, Kalihilingi, Kec. Brebes, Kabupaten Brebes, Jawa Tengah 52212
10	Waduk Perjalin	Jalan Raya Waduk Perjalin 41, RT 03/ 01, Waduk Perjalin, Winduaji, Paguyangan, Winduaji, SawahLadang, Win

Gambar 9 Tampilan Data Wisata

REKOMWISATA.



TEKNIKAL BOOK

SISTEM REKOMENDASI WISATA KABUPATEN BREBES BERBASIS TRACKING MAPS DENGAN METODE KNN

Oleh:

Dhiya Shilmi Aulia

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

Dwi Intan Af'idah, S.T., M.Kom

Selamat datang di *Website* RekomWisata, platform digital yang dirancang khusus untuk membantu anda menjelajahi keindahan alam, kekayaan budaya dan ragam destinasi menarik yang ada di Kabupaten Brebes, Jawa Tengah. *Website* ini hadir sebagai panduan lengkap bagi wisatawan lokal maupun luar daerah yang ingin mengenal dan mengeksplorasi pesona brebes lebih dalam. Peristiwa merupakan salah satu sektor yang terus tumbuh dan berkembang pesat di berbagai daerah di Indonesia. Beragam destinasi wisata dengan keunikan dan daya Tarik masing – masing telah menjadi daya pikat bagi wisatawan domestic maupun mancanegara. Namun, seiring dengan banyaknya pilihan tempat wisata yang tersedia, wisatawan sering kali menghadapi kesulitan dalam menentukan destinasi yang sesuai dengan kebutuhan, minat, dan anggaran mereka. Untuk menjawab kebutuhan ini, sistem RekomWisata dikembangkan sebagai sebuah platform digital yang dapat membantu wisatawan dalam memilih destinasi wisata yang sesuai dengan preferensi dan kebutuhan masing-masing. Dengan memanfaatkan teknologi informasi dan sistem rekomendasi, RekomWisata diharapkan dapat memberikan pengalaman berwisata yang lebih efisien, informatif, dan memuaskan. Seiring dengan perkembangan teknologi dan pola konsumsi informasi yang semakin digital, kebutuhan akan sistem yang dapat memberikan rekomendasi tempat wisata yang relevan juga semakin tinggi. Banyak wisatawan yang belum sepenuhnya memanfaatkan teknologi untuk merencanakan perjalanan, sehingga kesulitan mendapatkan informasi lengkap terkait daya Tarik, fasilitas dan biaya dari suatu destinasi wisata. Saat ini, Sebagian besar wisatawan hanya mengandalkan media

sosial, situs *web*, atau rekomendasi dari kerabat dekat untuk menentukan tujuan perjalanan. Hal ini dapat mengakibatkan risiko pilihan tempat wisata yang kurang sesuai dengan kebutuhan atau ekspektasi pengunjung. Padahal, dengan teknologi yang ada, berbagai data terkait tempat wisata dapat dikumpulkan dan dianalisis guna memberikan rekomendasi yang lebih spesifik dan personal.

Di era digital saat ini, kebutuhan Masyarakat terhadap informasi yang cepat, akurat, dan mudah diakses semakin meningkat, termasuk dalam hal pencarian destinasi wisata. Banyak wisatawan, khususnya generasi muda, lebih mengandalkan internet daripada brosur atau informasi lisan untuk merencanakan perjalanan mereka. Namun, informasi yang tersedia di internet sering kali tersebar, tidak terstruktur, dan kurang terpercaya. Pengembangan aplikasi ini juga merupakan bentuk adaptasi terhadap transformasi digital, sekaligus kontribusi nyata dalam mendukung Pembangunan pariwisata berbasis teknologi. Dengan tampilan yang sederhana dan fitur interaktif, RekomWisata diharapkan dapat menjadi jembatan antara pengelola wisata lokal.

Perangkat yang dibutuhkan

1. Perangkat Lunak

Perangkat lunak yang digunakan adalah sebagai berikut:

- Sistem operasi seperti Windows / macOS / Linux
- Visual Studio Code
- *Web* Browser seperti Google Chrome, Microsoft Edge dll

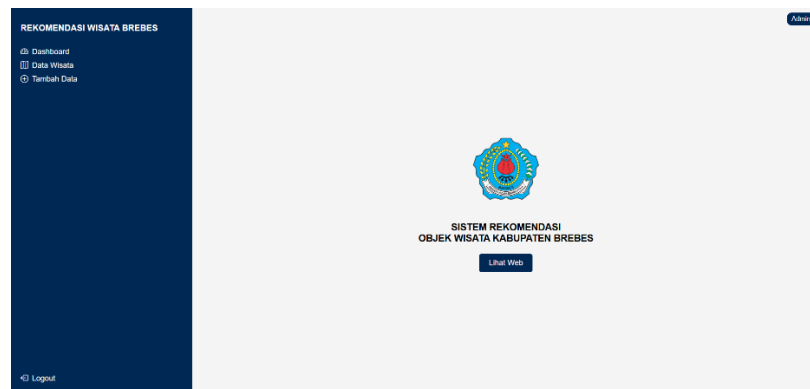
2. Perangkat Keras

Perangkat keras yang digunakan sebagai berikut:

- Laptop atau Komputer PC
- Mouse

3. Berikut uraian spesifikasi modul

- Modul untuk developer



Bagian ini hanya dapat diakses oleh pengembang dan dirancang khusus untuk mempermudah proses pembaruan konten dalam *website*, terutama pada bagian deskripsi. Dengan adanya fitur ini, developer dapat dengan mudah mengelola, memperbarui, dan menyesuaikan informasi yang ditampilkan kepada pengguna.

- Modul untuk pengguna



Bagian ini adalah *website* RekomWisata bagian yang akan ditampilkan untuk pengguna beserta beragam fitur yang disediakan oleh *web* berupa Beranda, Sejarah, dan Rekomendasi.

1. Import Library

```
1 from flask import Flask, Response, render_template, request, redirect, url_for, session, jsonify
2 from flask_mysqlldb import MySQL
3 from flask_bcrypt import Bcrypt
4 from werkzeug.utils import secure_filename
5 from flask_mysqlldb import MySQL
6 import MySQLdb.cursors
7 import os
8 import numpy as np
9 from flask import Flask, render_template, request
10 import pandas as pd
11 import joblib
12 from sklearn.metrics.pairwise import euclidean_distances
13 from sklearn.neighbors import NearestNeighbors
14 from sklearn.preprocessing import LabelEncoder, StandardScaler
15 import base64
16 from flask import flash
```

Berikut penjelasan singkat mengenai bagian-bagian dari script tersebut:

- Flask : Framework *web* berbasis python dan untuk membuat instance aplikasi *web*.
- Response, jsonify : Untuk mengatur respon HTTP dan data JSON.
- render_template : Menampilkan file HTML.
- request : Mengakses data yang dikirim dari form, URL, atau metode HTTP lain.
- redirect : Mengalihkan pengguna ke URL lain.
- session : Menyimpan data pengguna secara sementara (misalnya saat *Login*).
- jsonify : Mengubah data Python (seperti dictionary) menjadi JSON yang bisa dikirim ke client.
- from flask_mysqlldb import MySQL : Menghubungkan aplikasi Flask ke database MySQL.

- `from flask_bcrypt import Bcrypt` : Digunakan untuk mengaktifkan fitur keamanan password di aplikasi *web* Flask, yaitu dengan cara mengenkripsi password sebelum disimpan.
- `from werkzeug.utils import secure_filename` : Digunakan untuk mengamankan nama file yang diunggah pengguna ke server.
- `import MySQLdb.cursors` : digunakan untuk mengelola dan memformat hasil query dari database MySQL dalam aplikasi Python. Modul ini memungkinkan pemilihan jenis cursor yang sesuai (seperti dictionary) sehingga proses pengolahan data dari database menjadi lebih fleksibel dan aman.
- `import os` : untuk mengakses fungsi sistem seperti path, environment variable dll.
- `import numpy as np` : untuk memuat library NumPy ke dalam program, sehingga dapat digunakan untuk operasi statistic dan manipulasi data numerik secara efisien.
- `import pandas as pd` : digunakan untuk mengaktifkan fitur analisis dan manipulasi data berbasis table dalam program Python.
- `import joblib` : untuk mengimpor library joblib yang berfungsi untuk menyimpan dan memuat objek Python ke file.
- `from sklearn.metrics.pairwise import euclidean_distances` : digunakan untuk menghitung jarak Euclidean antara pasangan data, yang berguna dalam proses pencocokan, pengelompokan, atau rekomendasi berbasis kemiripan data.

- `from sklearn.neighbors import NearestNeighbors` : digunakan untuk mencari data yang paling mirip berdasarkan jarak, sistem rekomendasi pencarian dan analisis kemiripan antar datanya.
- `from sklearn.preprocessing import LabelEncoder, StandardScaler` : digunakan dalam proses pra-pemrosesan data, agar data kategori bisa diubah menjadi angka dan data numerik memiliki skala yang seragam.
- `import base64` : digunakan untuk mengubah data biner (seperti gambar atau file) menjadi format teks agar bisa disimpan di database yang dikirim melalui *web* (seperti dalam `url.json`, atau `html`).
- `from flask import flash` : digunakan untuk mengirim pesan sementara kepada pengguna, seperti peringatan atau konfirmasi, yang akan ditampilkan di halaman *web* berikutnya dan menghilang setelah dibaca. (sistem *Login*).

2. Konfigurasi Database

```
app = Flask(__name__) # Pastikan ini ada sebelum konfigurasi MySQL

app.secret_key = "aull"
# Konfigurasi MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root' # Ganti dengan username MySQL Anda
app.config['MYSQL_PASSWORD'] = '' # Ganti dengan password MySQL Anda
app.config['MYSQL_DB'] = 'sig' # Ganti dengan nama database Anda
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'

mysql = MySQL(app)
bcrypt = Bcrypt(app)
```

Berikut penjelasan dari konfigurasi database : kode tersebut merupakan konfigurasi awal dalam aplikasi *web* Flask yang berfungsi untuk menghubungkan aplikasi ke database MySQL serta mengaktifkan fitur

enkripsi password menggunakan Bcrypt. Baris pertama membuat instance Flask sebagai inti aplikasi, lalu secret-key ditetapkan untuk mengamankan session dan fitur seperti flash message. Selanjutnya, pengaturan koneksi database diatur melalui app.config, termasuk host, *Username*, password, nama database, dan jenis cursor yang digunakan (dictCursor) agar hasil query dapat diakses seperti dictionary. Terakhir, MySQL(app) dan Bcrypt(app) menginisialisasi koneksi database dan sistem enkripsi ke dalam aplikasi Flask agar siap digunakan dalam proses autentikasi dan manajemen data.

3. Memuat Dataset CSV

```
# Load dataset
df = pd.read_csv('data/datasetrekomendasi.csv', delimiter=';')

# Fungsi buat label harga berdasarkan harga numerik
def label_harga(h):
    if h == 0:
        return 'gratis'
    elif 1000 <= h < 10000:
        return 'murah'
    elif 10000 <= h < 15000:
        return 'sedang'
    elif 15000 <= h <= 50000:
        return 'mahal'
    else:
        return 'lainnya' # Opsional, jika ada harga di luar jangkauan
```

Script code diatas ini bertujuan untuk membaca data dari file CSV yang berisi data tempat wisata atau rekomendasi, lalu mengelompokkan nilai harga numerik ke dalam label kategori seperti gratis, murah, sedang, dan mahal, agar data lebih mudah dianalisis atau ditampilkan kepada pengguna. Fungsi ini untuk sistem rekomendasi berbasis preferensi harga.

4. Pra-Pemrosesan Data

```
49 # Tambah kolom harga_kategori ke dataframe
50 df['harga_kategori'] = df['harga'].apply(label_harga)
51
52 # Load model dan encoder
53 knn = joblib.load('models/model/model_knn1.pkl')
54 label_encoder = joblib.load('models/model/label_encoder_kategori1.pkl')
55 le_harga_kat = joblib.load('models/model/label_encoder_harga1.pkl')
56 scaler = joblib.load('models/model/scaler1.pkl')
57
58 # Encode kolom kategori dan harga_kategori pada dataframe
59 df['kategori_encoded'] = label_encoder.transform(df['kategori'])
60 df['harga_kategori_encoded'] = le_harga_kat.transform(df['harga_kategori'])
61
62 # Pastikan latitude dan longitude bertipe float
63 df['latitude'] = df['latitude'].astype(float)
64 df['longitude'] = df['longitude'].astype(float)
65
66 ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}
67
68 # Fungsi untuk mengecek ekstensi file yang diizinkan
69 def allowed_file(filename):
70     return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
```

Berikut penjelasan singkat pada bagian script code diatas ini:

- `df['harga_kategori'] = df['harga'].apply(label_harga)` : Fungsi `label_harga` digunakan untuk mengubah nilai numerik harga menjadi label kategori (gratis, murah, sedang, dan mahal) pada kolom baru `harga_kategori`.
- Model dan encoder yang telah dilatih sebelumnya dimuat dari file `.pkl` menggunakan `joblib`. Termasuk model KNN, label encoder untuk kategori wisata dan harga, serta scaler untuk normalisasi data.
- Melakukan Encoding pada kolom kategori dan harga : Data kategori (kategori) dan hasil pelabelan harga (`harga_kategori`) diubah menjadi nilai numerik agar bisa di proses oleh model machine learningnya.
- Konversi *Latitude* dan *Longitude* ke tipe float : Memastikan kolom *latitude* dan *longitude* memiliki tipe data float yang diperlukan untuk perhitungan jarak Lokasi atau input model.

- Ekstensi File yang diizinkan : Menentukan jenis ekstensi file yang boleh diunggah oleh pengguna, biasanya digunakan dalam form upload gambar dengan tipe file (png, jpg, jpeg, gif).
- Fungsi validasi ekstensi file : Fungsi `allowed_file()` mengecek apakah nama file memiliki ekstensi yang sesuai dengan yang diizinkan, dan untuk pengguna agar tidak mengunggah file sembarangan misalnya (.exe atau .py).

5. Route Rekomendasi

```

72 # =====
73 # ROUTE REKOMENDASI
74 # =====
75
76 @app.route('/rekomendasi', methods=['GET', 'POST'])
77 def rekomendasi():
78     rekomendasi_list = []
79
80     # Ambil data wisata dari database
81     cur = mysql.connection.cursor()
82     cur.execute("SELECT * FROM wisata")
83     wisata_data = cur.fetchall()
84     cur.close()
85
86     # Ubah hasil query ke DataFrame
87     df_db = pd.DataFrame(wisata_data)
88
89     if df_db.empty:
90         return "Data wisata kosong."

```

- `@app.route('/rekomendasi', methods=['GET', 'POST'])` : Mendefinisikan rute URL /rekomendasi yang bisa diakses dengan metode HTTP GET dan POST, Ketika pengguna mengakses rute ini, fungsi rekomendasi() akan dijalankan. (decorator route)
- Fungsi dan variable awal

def rekomendasi():

rekomendasi_list = [] : Fungsi rekomendasi () didefinisikan untuk menangani proses pengambilan dan pengolahan data.

Rekomendasi_list disiapkan sebagai list kosong untuk menampung hasil rekomendasinya.

- Mengambil Data dari database

`cur = mysql.connection.cursor()` : membuat cursor koneksi ke database

`cur.execute("SELECT * FROM wisata")` : menjalankan query SQL untuk mengambil semua data dari table wisata.

`wisata_data = cur.fetchall()`: data disimpan dalam variable wisata_data lalu koneksi ditutup, data biasanya berupa list of tuples
`cur.close()` .

- Mengonversi ke dataframe `df_db = pd.DataFrame(wisata_data)`: data hasil query SQL diubah menjadi dataframe menggunakan pandas agar bisa lebih mudah diolah(misalnya difilter, dihitung jaraknya, dsb).

- Validasi jika data kosong

`if df_db.empty:`

`return "Data wisata kosong."` : mengecek apakah dataframe kosong jika tidak ada data wisata yang diambil dari database, maka akan mengembalikan pesan “ Data Wisata Kosong”.

6. Route Rekomendasi

```
92 # Fungsi label harga
93 def label_harga(h):
94     if h == 0:
95         return 'gratis'
96     elif 1000 <= h < 10000:
97         return 'murah'
98     elif 10000 <= h < 15000:
99         return 'sedang'
100     elif 15000 <= h <= 50000:
101         return 'mahal'
102     else:
103         return 'lainnya'
104
105 # Tambahkan kolom harga_kategori
106 df_db['harga_kategori'] = df_db['harga'].apply(label_harga)
107
108 try:
109     # Encode kategori dan harga
110     df_db['kategori_encoded'] = label_encoder.transform(df_db['kategori'])
111     df_db['harga_kat_encoded'] = le_harga_kat.transform(df_db['harga_kategori'])
112 except Exception as e:
113     return f"Encoding gagal: {e}"
```

- Fungsi `label_harga(h)` :
Gratis : jika harga 0
Murah : 1.000 - 10.000
Sedang : 10.000 - 15.000
Mahal : 15.000 – 50.000
- Menambahkan kolom kategori harga ke dataframe
`df_db['harga_kategori'] = df_db['harga'].apply(label_harga)` :
menggunakan fungsi `label_harga` untuk membuat kolom baru
`harga_kategori` di dataframe `df_db` berdasarkan kolom `harga`.
- Penanganan eror jika encoding gagal
`except Exception as e:`
`return f"Encoding gagal: {e}"` : jika terjadi kesalahan saat
encoding (misalnya label tidak dikenali), sistem akan
mengembalikan pesan eror yang menjelaskan penyebabnya. Ini

mencegah aplikasi langsung crash dan memberi informasi kepada pengembang atau pengguna.

```
115 # Konversi latitude dan longitude ke float
116 try:
117     df_db['latitude'] = df_db['latitude'].astype(float)
118     df_db['longitude'] = df_db['longitude'].astype(float)
119 except Exception as e:
120     return f"Konversi latitude/longitude gagal: {e}"
121
122 # Pastikan kolom rating tidak null
123 df_db['rating'] = df_db['rating'].fillna(0).astype(float)
124
125 # Ambil kategori unik untuk dropdown
126 kategori_list = sorted(df_db['kategori'].unique())
127
128 if request.method == 'POST':
129     try:
130         kategori_input = request.form['kategori']
131         harga_kategori_input = request.form['harga_kategori']
132         rating_input = float(request.form['rating'])
133
134         # Validasi input
135         if kategori_input not in label_encoder.classes_ or harga_kategori_input not in le_harga_kat.classes_:
136             return "Input kategori atau harga tidak valid."
```

- Menangani kolom rating yang kosong `df_db['rating'] = df_db['rating'].fillna(0).astype(float)` : mengisi nilai kosong (null) pada kolom rating dengan 0. Lalu dikonversi ke float agar bisa digunakan dalam perhitungan numerik atau model machine learning.
- Mengambil daftar kategori untuk dropdown `kategori_list = sorted(df_db['kategori'].unique())`: mengambil semua nilai unik dari kolom kategori, digunakan untuk menampilkan pilihan kategori wisata di form HTML.
- Penangan form input dari metode POST

```
if request.method == 'POST':
```

```
    try:
```

```
        kategori_input = request.form['kategori']
```

```
        harga_kategori_input = request.form['harga_kategori']
```

rating_input = float(request.form['rating']) : jika halaman /rekomendasi diakses melalui metode POST biasanya dari form HTML maka data input dibaca dari form kategori_input → input kategori dari pengguna.

- Validasi input if kategori_input not in label_encoder.classes_ or harga_kategori_input not in le_harga_kat.classes_ :

return "Input kategori atau harga tidak valid." : mengecek apakah input dari pengguna terdapat dalam kelas yang dikenal (classes_) oleh encoder. Jika input tidak valid, maka sistem akan menolak dan menampilkan pesan error.

7. Route AdminLogin

```
198 # Route login admin
199 @app.route('/adminlogin', methods=['GET', 'POST'])
200 def adminlogin():
201     if request.method == 'POST':
202         username = request.form['username']
203         password = request.form['password']
204
205         cur = mysql.connection.cursor()
206         cur.execute("SELECT * FROM admins WHERE username = %s", (username,))
207         admin = cur.fetchone()
208         cur.close()
209
210         if admin and bcrypt.check_password_hash(admin['password'], password):
211             session['admin_id'] = admin['id']
212             session['admin_username'] = admin['username']
213             return redirect(url_for('dashboard')) # Arahkan ke dashboard jika login sukses
214         else:
215             error = "Username atau password salah!"
216             return render_template('admin/adminlogin.html', error=error)
217
218     return render_template('admin/adminlogin.html')
```

Pada script code route adminLogin adalah rute pada aplikasi Flask yang digunakan untuk menangani proses Login admin melalui metode GET dan POST pada URL /adminLogin. Saat pengguna mengakses halaman ini dengan metode GET, maka akan ditampilkan halaman form Login dari template admin /adminLogin.html. Namun, jika metode yang digunakan

adalah POST maka data *Username* dan password yang diinputkan pada form akan diambil dan diproses. Selanjutnya, aplikasi melakukan query ke database untuk mencari data admin berdasarkan *Username* yang dimasukan. Jika data admin ditemukan dan password yang dimasukan sesuai dengan hash password yang tersimpan di database (divalidasi menggunakan fungsi `bcrypt.check_password_hash`), maka proses *Login* dianggap berhasil. Pada kondisi ini, aplikasi akan menyimpan informasi *Login* ke dalam session seperti `admin_id` dan `admin_Username`, lalu mengarahkan admin ke halaman *Dashboard* menggunakan fungsi `redirect`. Namun jika data tidak ditemukan atau password salah, maka akan ditampilkan Kembali halaman *Login* dengan pesan kesalahan “ *Username* atau password salah!”. Dengan demikian, route ini bertugas penuh dalam menangani validasi *Login* admin secara aman dan mengelola alur masuk ke sistem.

8. Route Register Admin

```
220 # Route register admin
221 @app.route('/registeradmin', methods=['GET', 'POST'])
222 def registeradmin():
223     if request.method == 'POST':
224         username = request.form['username']
225         email = request.form['email']
226         password = request.form['password']
227         hashed_password = bcrypt.generate_password_hash(password).decode('utf-8')
228
229         cur = mysql.connection.cursor()
230         cur.execute("INSERT INTO admins (username, email, password) VALUES (%s, %s, %s)", (username, email, hashed_password))
231         mysql.connection.commit()
232         cur.close()
233
234         success_message = f"Admin '{username}' berhasil didaftarkan!"
235         return render_template('registeradmin.html', success_message=success_message, username=username, email=email)
236
237     return render_template('admin/registeradmin.html')
```

Bagian route `registeradmin` pada code flask ini digunakan untuk menangani proses pendaftaran admin baru melalui metode GET dan POST pada URL. Saat pengguna mengakses halaman ini dengan metode GET.

/registeradmin.html. Jika metode yang digunakan adalah POST, maka aplikasi akan mengambil data yang diinputkan oleh pengguna, yaitu *Username*, email, dan password. Password yang diterima kemudian akan di hash menggunakan `bcrypt.generate_password_hash()` untuk keamanan, lalu dikonversi menjadi string UTF-8 agar dapat di simpan dalam database. Setelah itu, aplikasi membuat koneksi ke database dan mengeksekusi perintah `INSERT INTO admins` untuk menyimpan data admin baru ke dalam table `admins`. Setelah berhasil menyimpan data, koneksi database ditutup dan aplikasi menampilkan Kembali halaman form dengan pesan sukses berupa “Admin berhasil didaftarkan!”, yang dikirimkan ke template `registeradmin.html`. Dengan demikian, route ini berfungsi untuk menangani input data pendaftaran admin, mengamankan password, dan menyimpan data tersebut ke database.

9. Route `forgotpassword`

```
239 @app.route('/forgotpassword', methods=['GET', 'POST'])
240 def forgotpassword():
241     message = None
242     if request.method == 'POST':
243         username = request.form['username']
244         new_password = request.form['new_password']
245
246         cursor = mysql.connection.cursor()
247
248         # Cek apakah username ada
249         cursor.execute("SELECT * FROM admins WHERE username = %s", (username,))
250         user = cursor.fetchone()
251
252         if user:
253             hashed_password = bcrypt.generate_password_hash(new_password).decode('utf-8')
254             cursor.execute("UPDATE admins SET password = %s WHERE username = %s", (hashed_password, username))
255             mysql.connection.commit()
256             message = f"Password untuk '{username}' berhasil diubah."
257         else:
258             message = f"Username '{username}' tidak ditemukan."
259         cursor.close()
260
261     return render_template('admin/forgotpassword.html', success_message=message)
262
263
```

Route `forgotpassword` pada aplikasi Flask ini berfungsi untuk menangani proses reset password admin yang lupa password. Rute ini mendukung metode GET untuk menampilkan form reset password, serta POST untuk memproses data yang dikirimkan melalui form. Ketika pengguna

mengirimkan form dengan metode POST, aplikasi mengambil data *Username* dan *new_password* dari input form. Kemudian, dilakukan pengecekan ke database menggunakan query SELECT untuk memastikan bahwa *Username* tersebut benar-benar ada di tabel *admins*. Jika ditemukan, password baru yang dimasukkan akan di-hash menggunakan *bcrypt* agar lebih aman sebelum disimpan. Setelah itu, aplikasi menjalankan query UPDATE untuk mengganti password lama di database dengan password baru yang sudah di-hash. Jika proses berhasil, aplikasi menampilkan pesan bahwa password berhasil diubah. Namun jika *Username* tidak ditemukan, maka akan ditampilkan pesan bahwa *Username* tidak ditemukan. Akhirnya, template *admin/forgotpassword.html* akan dirender dengan menampilkan pesan sukses atau kesalahan tersebut.

10. Route Adminlogout

```
264 @app.route('/adminlogout')
265 def adminlogout():
266     session.pop('admin_id', None)
267     session.pop('admin_username', None)
268     flash('Berhasil logout.', 'success')
269     return redirect(url_for('adminlogin'))
270
```

Route *adminlogout* adalah fungsi pada aplikasi Flask yang digunakan untuk menangani proses logout admin. Ketika route ini diakses melalui URL */adminlogout*, fungsi *adminlogout()* akan dijalankan. Di dalam fungsi ini, dua item pada *session* yaitu *'admin_id'* dan *'admin_Username'* dihapus menggunakan *session.pop()* untuk menghapus data sesi pengguna, sehingga status *Login* admin menjadi tidak aktif. Setelah itu, fungsi *flash()* digunakan

untuk menampilkan pesan "Berhasil logout." kepada pengguna dengan kategori pesan 'success', yang biasanya digunakan untuk notifikasi di tampilan antarmuka. Terakhir, pengguna diarahkan kembali ke halaman *Login* admin (*adminLogin*) menggunakan `redirect(url_for('adminLogin'))`. Dengan demikian, route ini memastikan bahwa admin keluar dari sesi secara aman dan diarahkan kembali ke form *Login*.

11. Route Home

```
277 @app.route('/home')
278 def home():
279     try:
280         cur = mysql.connection.cursor()
281
282         # Ambil kategori unik dari DB
283         cur.execute("SELECT DISTINCT kategori FROM wisata")
284         kategori_data = cur.fetchall()
285         kategori_list = [row['kategori'] for row in kategori_data]
286
287         # Ambil parameter dari URL
288         kategori_filter = request.args.get('kategori', '').strip()
289         search_query = request.args.get('search', '').strip()
290
291         # Buat query dasar
292         query = "SELECT id, nama_wisata, gambar, alamat, rating FROM wisata WHERE 1=1"
293         params = []
294
295         # Tambah filter kategori jika dipilih
296         if kategori_filter:
297             query += " AND kategori = %s"
298             params.append(kategori_filter)
299
300         # Tambah filter pencarian jika diisi
301         if search_query:
302             query += " AND nama_wisata LIKE %s"
303             params.append(f"%{search_query}%")
304
```

Route home pada aplikasi Flask ini menangani tampilan halaman utama (*/home*) yang berisi daftar tempat wisata dan memungkinkan pengguna melakukan filter berdasarkan kategori dan pencarian nama wisata. Berikut penjelasan alurnya:

- Pertama, program membuka koneksi ke database dengan `cursor = mysql.connection.cursor()`. Lalu, dieksekusi `query SELECT DISTINCT kategori FROM wisata` untuk mengambil semua kategori unik dari tabel wisata, yang kemudian diubah menjadi list `kategori_list` agar dapat ditampilkan ke *User* sebagai opsi filter.

Selanjutnya, program mengambil dua parameter dari URL yaitu kategori (untuk filter berdasarkan kategori wisata) dan search (untuk pencarian berdasarkan nama wisata), menggunakan `request.args.get()`. Kedua nilai ini kemudian digunakan untuk menyusun query SQL dinamis.

- Query dasar yang dibuat adalah: *SELECT id, nama_wisata, gambar, alamat, rating FROM wisata WHERE 1=1*

Query ini akan bertambah secara dinamis tergantung apakah kategori_filter atau search_query diisi. Jika kategori_filter tidak kosong, maka query ditambahkan dengan AND kategori = %, dan jika search_query tidak kosong, query ditambahkan dengan AND nama_wisata LIKE %. Nilai parameter yang sesuai dimasukkan ke dalam list params.

12. Route Detail Wisata

```
@app.route('/detail_wisata/<int:id>', methods=['GET', 'POST']) ###
def detail_wisata(id):
    cur = mysql.connection.cursor(MySQLdb.cursors.DictCursor)

    # Ambil data wisata
    cur.execute("SELECT * FROM wisata WHERE id = %s", (id,))
    wisata = cur.fetchone()
    if not wisata:
        cur.close()
        return "Wisata tidak ditemukan", 404

    # Tambah ulasan
    if request.method == 'POST':
        if 'user' not in session:
            flash("Anda harus login dengan Google sebelum memberi ulasan", "warning")
            return redirect(url_for('login'))

        user_id = session['user']['id']
        rating = int(request.form['rating'])
        komentar = request.form['komentar']
```

Potongan kode pada route `@app.route('/detail_wisata/<int:id>', methods=['GET', 'POST'])` berfungsi untuk menampilkan detail informasi wisata sekaligus menangani proses penambahan ulasan oleh pengguna. Fungsi `detail_wisata(id)` menerima parameter `id` yang merepresentasikan identitas unik suatu wisata, kemudian melakukan koneksi ke database dan menjalankan perintah *SQL* untuk mengambil data berdasarkan ID tersebut. Jika data tidak ditemukan, sistem akan menutup koneksi dan menampilkan pesan kesalahan “Wisata tidak ditemukan” dengan status kode 404. Apabila metode yang digunakan adalah *POST*, sistem akan memeriksa apakah pengguna sudah login; jika belum, pengguna akan diarahkan ke halaman login dengan pesan peringatan menggunakan fungsi *flash()*. Setelah login diverifikasi, sistem mengambil data ulasan dari form berupa rating dan komentar yang diberikan pengguna, serta menyimpan ID pengguna dari sesi aktif. Secara keseluruhan, route ini berperan penting dalam mengelola tampilan detail wisata serta memfasilitasi pengguna untuk memberikan ulasan yang akan mendukung proses evaluasi dan rekomendasi pada sistem.

```
# Simpan gambar ulasan
gambar_path = None
if 'gambar' in request.files:
    file = request.files['gambar']
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        unique_name = str(uuid.uuid4()) + "_" + filename
        filepath = os.path.join("static/uploads", unique_name)
        os.makedirs("static/uploads", exist_ok=True)
        file.save(filepath)
        gambar_path = filepath

# Simpan ulasan ke DB
cur.execute(
    "INSERT INTO ulasan (wisata_id, user_id, rating, komentar, gambar) VALUES (%s, %s, %s, %s, %s)",
    (id, user_id, rating, komentar, gambar_path)
)
mysql.connection.commit()
```

Potongan kode tersebut merupakan bagian dari fungsi `detail_wisata()` yang bertanggung jawab untuk **menyimpan gambar ulasan dan data ulasan pengguna ke dalam database**. Pada bagian ini, sistem terlebih dahulu melakukan pengecekan terhadap file gambar yang diunggah melalui form menggunakan `request.files['gambar']`. Jika file gambar tersedia dan telah memenuhi kriteria ekstensi yang diizinkan (melalui fungsi validasi `allowed_file()`), maka sistem akan membuat nama file baru yang unik menggunakan kombinasi fungsi `uuid.uuid4()` dan nama asli file untuk menghindari duplikasi nama file. File tersebut kemudian disimpan ke dalam direktori `static/uploads` menggunakan metode `save()`, dan jalur penyimpanan file disimpan dalam variabel `gambar_path`. Selanjutnya, bagian kode di bawahnya berfungsi untuk menyimpan seluruh data ulasan ke dalam database melalui perintah `SQL INSERT INTO` ulasan, yang mencakup `wisata_id`, `user_id`, `rating`, `komentar`, serta `gambar_path` yang berisi lokasi file gambar ulasan. Setelah perintah `SQL` dijalankan, fungsi `mysql.connection.commit()` digunakan untuk memastikan data tersimpan permanen di dalam database. Dengan demikian, bagian kode ini berperan penting dalam mengelola proses penyimpanan data ulasan yang mencakup teks dan media gambar, sehingga mendukung fitur interaktif pada sistem rekomendasi wisata.

```
# Hitung ulang rating_total
cur.execute("SELECT AVG(rating) as avg_rating FROM ulasan WHERE wisata_id = %s", (id,))
avg_rating = cur.fetchone()['avg_rating']
cur.execute("UPDATE wisata SET rating_total = %s WHERE id = %s", (avg_rating, id))
mysql.connection.commit()

flash("Ulasan berhasil ditambahkan!", "success")
return redirect(url_for('detail_wisata', id=id))
```

Potongan kode tersebut merupakan bagian akhir dari fungsi `detail_wisata()` yang digunakan untuk **menghitung ulang nilai rata-rata rating** suatu wisata setelah pengguna menambahkan ulasan baru. Proses diawali dengan eksekusi perintah *SQL SELECT AVG(rating) as avg_rating FROM ulasan WHERE wisata_id = %s*, yang bertujuan untuk mengambil nilai rata-rata (AVG) dari seluruh data rating yang terdapat pada tabel ulasan berdasarkan `wisata_id` tertentu. Nilai rata-rata tersebut kemudian disimpan dalam variabel `avg_rating` menggunakan metode `fetchone()`. Setelah nilai rata-rata diperoleh, sistem melakukan pembaruan data pada tabel wisata melalui perintah *UPDATE wisata SET rating_total = %s WHERE id = %s*, yang berarti kolom `rating_total` akan diisi dengan nilai rata-rata terbaru sesuai dengan ID wisata terkait. Proses ini diakhiri dengan pemanggilan `mysql.connection.commit()` agar perubahan data tersimpan secara permanen di dalam database. Selanjutnya, fungsi `flash()` digunakan untuk menampilkan pesan notifikasi bahwa ulasan berhasil ditambahkan dengan status sukses, dan pengguna akan diarahkan kembali ke halaman detail wisata menggunakan `redirect(url_for('detail_wisata', id=id))`. Secara keseluruhan, bagian kode ini berfungsi untuk memastikan bahwa setiap kali pengguna memberikan ulasan baru, sistem secara otomatis memperbarui nilai rating keseluruhan destinasi wisata, sehingga informasi yang ditampilkan tetap akurat dan dinamis.

```
# Ambil ulasan
cur.execute("""
    SELECT u.rating, u.komentar, DATE_FORMAT(u.tanggal, '%d %M %Y') as tanggal, u.gambar, usr.name, usr.picture
    FROM ulasan u
    JOIN users usr ON u.user_id = usr.id
    WHERE u.wisata_id = %s
    ORDER BY u.tanggal DESC
""", (id,))
ulasan = cur.fetchall()
```

Potongan kode tersebut berfungsi untuk mengambil dan menampilkan data ulasan pengguna pada destinasi wisata tertentu. Proses ini dilakukan dengan menjalankan perintah *SQL* yang mengambil data dari tabel *ulasan* dan menggabungkannya dengan tabel *users* untuk menampilkan informasi pengguna seperti nama dan foto profil. Data yang diambil mencakup rating, komentar, gambar, serta tanggal ulasan yang diformat agar mudah dibaca, kemudian diurutkan berdasarkan tanggal terbaru. Hasil query disimpan dalam variabel *ulasan* menggunakan metode *fetchall()*. Secara keseluruhan, kode ini menampilkan daftar ulasan secara lengkap dan terstruktur untuk memperkaya tampilan halaman detail wisata.

```
# Hitung rating rata-rata & jumlah ulasan
cur.execute("""
    SELECT AVG(rating) as avg_rating, COUNT(*) as total_ulasan
    FROM ulasan
    WHERE wisata_id = %s
""", (id,))
rating_info = cur.fetchone()
avg_rating = rating_info['avg_rating'] if rating_info['avg_rating'] else 0
total_ulasan = rating_info['total_ulasan']
```

Potongan kode tersebut berfungsi untuk menghitung rata-rata rating dan jumlah ulasan dari suatu destinasi wisata berdasarkan ID yang diberikan. Melalui perintah *SQL*, sistem mengambil nilai rata-rata menggunakan fungsi *AVG(rating)* dan jumlah ulasan dengan *COUNT(*)*, kemudian menyimpannya dalam variabel *rating_info*. Jika belum ada ulasan, nilai rating akan diatur menjadi 0. Hasil perhitungan ini digunakan untuk

menampilkan data statistik berupa rata-rata rating dan total ulasan sebagai indikator kualitas dan popularitas destinasi wisata.

```
# Hitung distribusi rating (1-5 bintang)
cur.execute("""
    SELECT rating, COUNT(*) as jumlah
    FROM ulasan
    WHERE wisata_id = %s
    GROUP BY rating
""", (id,))
rating_distribution = cur.fetchall()

rating_counts = {i: 0 for i in range(1, 6)}
for row in rating_distribution:
    rating_counts[row['rating']] = row['jumlah']

rating_percentages = {
    i: (rating_counts[i] / total_ulasan * 100 if total_ulasan > 0 else 0)
    for i in range(1, 6)
}
```

Potongan kode tersebut berfungsi untuk menghitung distribusi rating ulasan pengguna pada suatu destinasi wisata berdasarkan jumlah bintang (1–5). Data diambil dari tabel *ulasan* dengan perintah SQL yang mengelompokkan nilai rating menggunakan GROUP BY dan menghitung jumlahnya dengan COUNT(*). Hasilnya disimpan dalam variabel *rating_distribution*, lalu diproses untuk menghitung jumlah dan persentase setiap rating terhadap total ulasan. Jika belum ada ulasan, nilai persentase diatur menjadi nol. Secara keseluruhan, kode ini menghasilkan data statistik distribusi rating yang dapat digunakan untuk menilai tingkat kepuasan pengguna terhadap destinasi wisata

13. Route Dashboard

```
13 # Route dashboard setelah login sukses
14 @app.route('/dashboard', methods=['GET'])
15 def dashboard():
16     if 'admin_id' not in session:
17         return redirect(url_for('adminlogin')) # Jika belum login, kembali ke login
18     return render_template('dashboard.html', admin_username=session['admin_username'])
19
20
```

Route *Dashboard* pada aplikasi Flask ini digunakan untuk menampilkan halaman *Dashboard* admin setelah berhasil *Login*. Route ini hanya menerima metode *GET* dan dapat diakses melalui URL */dashboard*. Di dalam fungsi *Dashboard()*, pertama-tama dilakukan pengecekan apakah *'admin_id'* ada dalam session. Ini berfungsi sebagai validasi apakah admin sudah *Login* atau belum. Jika *'admin_id'* tidak ada di session (artinya *User* belum *Login*), maka akan dilakukan redirect ke halaman *Login* admin (*adminLogin*) menggunakan *redirect(url_for('adminLogin'))*. Jika admin sudah *Login*, maka halaman *Dashboard.html* akan dirender, dan nama admin (*session['admin_username']*) dikirimkan ke template sebagai variabel *admin_username*. Hal ini memungkinkan halaman *Dashboard* menampilkan nama admin yang sedang *Login* sebagai bagian dari antarmukanya.

14. Route *Dashboard_data*

```
422 # Route ke halaman dashboard data
423 @app.route('/dashboard_data')
424 def dashboard_data():
425     cur = mysql.connection.cursor()
426     cur.execute("SELECT id, nama_wisata, alamat, jam_operasional, status, harga, maps, kategori, kamar_mandi, masjid, cafe_resto, parkir_mu")
427     data_wisata = cur.fetchall()
428     cur.close()
429
430     return render_template('admin/dashboard_data.html', data_wisata=data_wisata)
431
```

Route *Dashboard_data* pada aplikasi Flask berfungsi untuk menampilkan halaman *Dashboard* yang berisi data lengkap tempat wisata. Route ini

diakses melalui URL `/dashboard_data`. Ketika fungsi `Dashboard_data()` dipanggil, aplikasi membuka koneksi ke database menggunakan cursor, lalu mengeksekusi query SQL:

- `SELECT id, nama_wisata, alamat, jam_operasional, status, harga, maps, kategori, kamar_mandi, masjid, cafe_resto, parkir_motor, ...`

Query ini mengambil berbagai informasi detail mengenai tempat wisata dari tabel database, termasuk nama, alamat, jam operasional, fasilitas, dan lainnya. Seluruh data hasil query disimpan dalam variabel `data_wisata` melalui `cur.fetchall()`. Setelah data diambil, koneksi database ditutup dengan `cur.close()`, dan data tersebut dikirim ke halaman template `admin/dashboard_data.html` melalui fungsi `render_template()`, dengan data disimpan dalam variabel `data_wisata`.

15. Route Tambah data

```
432 @app.route('/tambah_data', methods=['GET', 'POST'])
433 def tambah_data():
434     if request.method == 'POST':
435         nama_wisata = request.form['nama_wisata']
436         alamat = request.form['alamat']
437         jam_operasional = request.form['jam_operasional']
438         status = request.form['status']
439         deskripsi = request.form['deskripsi']
440         harga = request.form['harga']
441         maps = request.form['maps']
442         latitude = request.form['latitude']
443         longitude = request.form['longitude']
444         kategori = request.form['kategori']
445         rating = request.form['rating']
446
447         fasilitas = {
448             "kamar_mandi": request.form['kamar_mandi'],
449             "masjid": request.form['masjid'],
450             "cafe_resto": request.form['cafe_resto'],
451             "parkir_mobil": request.form['parkir_mobil'],
452             "parkir_motor": request.form['parkir_motor'],
453             "warung": request.form['warung']
454         }
455
456         gambar_blob = request.files['gambar'].read() if 'gambar' in request.files and allowed_file(request.files['gambar'].filename) else None
457         gambar2_blob = request.files['gambar2'].read() if 'gambar2' in request.files and allowed_file(request.files['gambar2'].filename) else None
458         gambar3_blob = request.files['gambar3'].read() if 'gambar3' in request.files and allowed_file(request.files['gambar3'].filename) else None
459
```

Route `tambah_data` pada aplikasi Flask digunakan untuk menambahkan data baru tempat wisata ke dalam database. Route ini mendukung metode

'GET' dan 'POST', dan diakses melalui URL `/tambah_data`. Saat request menggunakan metode 'POST', fungsi `tambah_data()` akan mengambil data dari *form HTML* melalui `request.form`, seperti nama tempat wisata, alamat, jam operasional, status, deskripsi, harga, lokasi (`maps`, `latitude`, `longitude`), kategori, dan rating. Selain itu, data fasilitas juga dikumpulkan dalam bentuk dictionary, seperti `kamar_mandi`, `masjid`, `cafe_resto`, `parkir_mobil`, `parkir_motor`, dan `warung`.

Fungsi ini juga menangani unggahan gambar (`gambar`, `gambar2`, dan `gambar3`) dengan memverifikasi file menggunakan fungsi `allowed_file()`. Jika file valid dan terdapat dalam `request.files`, file tersebut akan dibaca sebagai blob dan disimpan dalam variabel (`gambar_blob`, dll). Jika tidak, nilainya akan `None`. Meskipun bagian insert ke database belum terlihat dalam potongan ini, fungsi ini jelas bertujuan untuk mengumpulkan seluruh data input dari form tambah wisata, termasuk fasilitas dan gambar, untuk nantinya disimpan ke dalam database.

16. Route Edit data

```
558 @app.route('/edit_data/<int:id>', methods=['GET', 'POST'])
559 def edit_data(id):
560     cur = mysql.connection.cursor()
561     cur.execute("SELECT * FROM wisata WHERE id = %s", (id,))
562     wisata = cur.fetchone()
563
564     if request.method == 'POST':
565         nama_wisata = request.form['nama_wisata']
566         alamat = request.form['alamat']
567         jam_operasional = request.form['jam_operasional']
568         status = request.form['status']
569         deskripsi = request.form['deskripsi']
570         harga = request.form['harga']
571         maps = request.form['maps']
572         kategori = request.form['kategori']
573         rating = request.form['rating']
574
575         # Tambahan: ambil latitude dan longitude dari form
576         latitude = float(request.form['latitude']) # atau dibagi 1e7 jika masih dalam format E7
577         longitude = float(request.form['longitude'])
578
579         kamar_mandi = request.form['kamar_mandi']
580         masjid = request.form['masjid']
581         cafe_resto = request.form['cafe_resto']
582         parkir_mobil = request.form['parkir_mobil']
583         parkir_motor = request.form['parkir_motor']
584         warung = request.form['warung']
585
```

Route `edit_data` merupakan endpoint Flask yang digunakan untuk menampilkan dan memproses formulir edit data tempat wisata berdasarkan id. Route ini dapat diakses melalui URL `/edit_data/<int:id>` dan mendukung metode `'GET'` dan `'POST'`. Ketika halaman diakses dengan metode `'GET'`, sistem akan membuka koneksi ke database dan menjalankan query `SELECT * FROM wisata WHERE id = %s` untuk mengambil data tempat wisata yang sesuai dengan id yang diberikan. Hasilnya disimpan dalam variabel `wisata`, yang nantinya digunakan untuk menampilkan data lama di form edit. Jika request menggunakan metode `'POST'`, artinya pengguna telah mengirimkan form update. Dalam blok ini, semua data form seperti `nama_wisata`, `alamat`, `jam_operasional`, `status`, `deskripsi`, `harga`, `maps`, `kategori`, dan `rating` akan diambil dari `request.form`. *Latitude* dan *longitude* juga diambil dan dikonversi ke tipe float. Kemudian data fasilitas seperti `kamar_mandi`, `masjid`, `cafe_resto`, `parkir_mobil`, `parkir_motor`, dan `warung` juga diambil dari form. Meski tidak ditampilkan di gambar, biasanya setelah ini data tersebut akan digunakan untuk menjalankan perintah `UPDATE` ke database agar data lama diperbarui.

17. Route Sejarah

```
628 # Route ke halaman Sejarah
629 @app.route('/sejarah')
630 def sejarah():
631     return render_template('sejarah.html')
632
633 if __name__ == '__main__':
634     app.run(debug=True)
635
```

Route sejarah adalah sebuah endpoint Flask yang menangani permintaan ke URL `/sejarah`. Fungsi `sejarah()` di dalam route ini sangat sederhana: ketika pengguna mengakses URL tersebut, server akan merespons dengan menampilkan halaman HTML bernama `sejarah.html`. Route ini tidak memproses data atau melakukan interaksi dengan database. Fungsinya hanya untuk menyajikan tampilan statis yang berisi informasi mengenai sejarah, yang kemungkinan merupakan bagian dari konten situs *web* (seperti sejarah tempat wisata, daerah, atau instansi tertentu).