LAPORAN PROYEK SAINS DATA

SISTEM REKOMENDASI DESTINASI WISATA DI PROVINSI LAMPUNG MENGGUNAKAN CONTENT-BASED FILTERING DAN LOCATION-BASED SERVICE

Disusun Oleh:

Ukasyah Muntaha	(122450028)
Syalaisha Andina Putriansyah	(122450121)
Nurul Alfajar Gumel	(122450127)

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA

2025

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sektor pariwisata menjadi salah satu sektor penting dalam perekonomian Indonesia. Kontribusinya tidak hanya mendukung peningkatan pendapatan daerah, tetapi juga mendorong pertumbuhan industri kreatif, transportasi, serta pemberdayaan masyarakat lokal. Pada Agustus 2025, kunjungan wisatawan mancanegara (wisman) mencapai 1,51 juta yang menunjukkan peningkatan sekitar 12,33% dibandingkan pada tahun sebelumnya [1]. Peningkatan juga terjadi pada wisatawan nusantara, dengan total 901,90 juta perjalanan pada Januari sampai September 2025 atau naik sebesar 18,99% dibandingkan tahun 2024 [2]. Hal tersebut menunjukkan bahwa sektor pariwisata, baik internasional maupun domestik, terus berkembang secara signifikan.

Sebagai bagian dari dinamika tersebut, Provinsi Lampung menjadi salah satu daerah dengan perkembangan pariwisata yang sangat signifikan. Berdasarkan data Dinas Pariwisata dan Ekonomi Kreatif Provinsi Lampung, jumlah perjalanan wisatawan nusantara pada tahun 2024 meningkat sekitar 29,90% dibandingkan tahun sebelumnya. Bahkan pada tahun 2025, pemerintah daerah memproyeksikan kunjungan wisatawan dapat melampaui 20 juta orang. Peningkatan ini memperlihatkan bahwa Lampung memiliki daya tarik wisata yang kuat, mulai dari wisata alam, budaya, hingga kawasan rekreasi modern.

Beragamnya pilihan destinasi menimbulkan kendala bagi wisatawan. Hal ini menyebabkan para wisatawan kesulitan dalam memilih destinasi yang paling sesuai dengan preferensi atau minat pribadi serta dalam menyusun rencana perjalanan yang efisien, terutama Ketika ingin mengunjungi beberapa lokasi. Kondisi ini diperburuk oleh fakta bahwa informasi mengenai karakteristik objek wisata dan jarak antar lokasi umumnya masih tersedia secara terpisah. Sebagian besar platform informasi wisata hanya menampilkan daftar lokasi tanpa memberikan rekomendasi personal maupun pertimbangan geografis. Akibatnya, wisatawan cenderung memilih destinasi berdasarkan popularitas, sehingga perjalanan menjadi kurang optimal dari segi kesesuaian minat maupun efisiensi waktu tempuh. Kondisi tersebut menunjukkan perlunya sistem yang mampu memberikan rekomendasi destinasi wisata yang lebih adaptif, relevan, dan bermanfaat bagi perencanaan perjalanan.

Sejumlah penelitian terkait sistem rekomendasi wisata telah dilakukan di Indonesia dengan menggunakan berbagai pendekatan. Pada Penelitian yang dilakukan oleh Dinda Pratiwi, Asrianda, dan Lidya Rosnita, menunjukkan bahwa metode *content-based filtering* menunjukkan kemampuan dalam menghasilkan rekomendasi berdasarkan kesesuaian karakteristik destinasi dengan preferensi pengguna [3]. Penelitian lain yang berjudul "Rekomendasi Wisata Kabupaten Magelang menggunakan Metode Content Based Filtering dan Location-Based Service" berhasil mengintegrasikan metode CBF dan LBS untuk menghasilkan rekomendasi wisata yang relevan dan sesuai dengan preferensi wisatawan di Kabupaten Magelang [4]. Meskipun demikian, sebagian besar penelitian masih terbatas pada wilayah berskala kabupaten dan belum banyak membahas bagaimana kecocokan konten destinasi dapat diterapkan bersamaan dengan pertimbangan jarak maupun urutan kunjungan pada wilayah dengan tingkat keberagaman destinasi yang lebih luas, seperti Provinsi Lampung. Selain itu, kajian mengenai pemanfaatan jarak geografis untuk mendukung efisiensi rute perjalanan wisata masih jarang ditemukan, sehingga wisatawan belum memperoleh sarana yang dapat membantu menyusun rencana perjalanan yang optimal.

Berdasarkan kebutuhan tersebut, proyek penelitian ini memiliki tujuan untuk mengembangkan sistem rekomendasi wisata Provinsi Lampung yang mampu membantu wisatawan memilih destinasi sesuai preferensi sekaligus mempertimbangkan kedekatan lokasi dan estimasi waktu tempuh. Penerapan metode Content-Based Filtering dan Location-Based Service (LBS) diharapkan dapat menghasilkan rekomendasi yang lebih adaptif serta mendukung proses pengambilan keputusan wisatawan. Pendekatan ini penting untuk menyediakan solusi yang tidak hanya menampilkan daftar destinasi, tetapi juga menawarkan rekomendasi yang relevan dan bermanfaat dalam penyusunan rencana perjalanan yang lebih efektif.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam proyek ini adalah:

1. Bagaimana menghasilkan rekomendasi destinasi wisata di Lampung berdasarkan kemiripan atribut konten menggunakan metode *Content-Based Filtering*?
2. Bagaimana mengintegrasikan jarak geografis antar destinasi wisata menggunakan Location-Based Service?
3. Bagaimana performa sistem rekomendasi yang menggabungkan

Content-Based Filtering dan Location-Based Service?

1.3 Tujuan Proyek

1.3.1 Tujuan Umum

Mengembangkan sistem rekomendasi destinasi wisata di Lampung yang mampu memberikan rekomendasi personal berdasarkan preferensi pengguna dan jarak geografis, sehingga wisatawan memperoleh kemudahan dalam menentukan destinasi yang sesuai kebutuhan serta menyusun rute perjalanan yang optimal.

1.3.2 Tujuan Khusus

1. Membangun sistem rekomendasi destinasi wisata di Lampung berdasarkan kemiripan atribut konten dengan menggunakan metode Content-Based Filtering.
2. Mengimplementasikan Location-Based Service untuk menghitung jarak antar destinasi wisata dan mengintegrasikan informasi tersebut ke dalam proses perankingan rekomendasi.
3. Mengetahui performa sistem rekomendasi yang menggabungkan Content-Based Filtering dan Location-Based Service.

1.4 Sasaran Proyek

1. Mengembangkan model rekomendasi destinasi wisata berbasis content-based filtering yang menghasilkan rekomendasi yang relevan dengan preferensi.
2. Mengintegrasikan fitur location-based service untuk memberikan rekomendasi yang sesuai preferensi pengguna dengan jarak antar destinasi yang berdekatan, sehingga memudahkan wisatawan menyusun rute perjalanan yang efisien.
3. Menghasilkan minimal 5 rekomendasi destinasi wisata yang relevan untuk setiap profil pengguna berdasarkan preferensi pengguna.
4. Membangun dashboard yang menyajikan hasil rekomendasi dengan *response time* ≤ 2 detik.

1.5 Manfaat Proyek

Proyek ini diharapkan dapat memberikan kontribusi terhadap pengembangan ilmu pengetahuan, khususnya dalam integrasi algoritma *Content-Based Filtering* dengan *Location-Based Service* (LBS). Penelitian ini memperkaya literatur

mengenai cara menyeimbangkan akurasi rekomendasi berdasarkan preferensi konten dengan efisiensi spasial (jarak lokasi).

1.5.1 Manfaat Teoritis

Proyek ini diharapkan dapat memberikan kontribusi terhadap pengembangan ilmu pengetahuan, khususnya dalam integrasi algoritma Content-Based Filtering dengan Location-Based Service (LBS). Penelitian ini memperkaya literatur mengenai cara menyeimbangkan akurasi rekomendasi berdasarkan preferensi konten dengan efisiensi spasial (jarak lokasi).

1.5.2 Manfaat Praktis

- Bagi Organisasi/Perusahaan: Memberikan media promosi digital yang efektif bagi para pemilik dan pengelola objek wisata agar destinasi mereka lebih mudah dikenali oleh masyarakat luas. Dengan terdatanya destinasi mereka dalam sistem rekomendasi, peluang untuk menjangkau calon wisatawan baru menjadi lebih besar, yang pada akhirnya diharapkan dapat meningkatkan jumlah kunjungan dan pendapatan usaha pariwisata tersebut.
- Bagi Masyarakat: Memudahkan wisatawan (masyarakat umum) dalam menemukan destinasi wisata di Lampung yang paling relevan dengan minat mereka dan terjangkau dari lokasi terkini. Dengan adanya sistem yang responsif, masyarakat dapat merencanakan perjalanan wisata dengan lebih efisien serta hemat waktu.
- Bagi Peneliti: Menjadi sarana implementasi nyata dari teori Data Science yang telah dipelajari, khususnya dalam aspek Data Engineering, pemodelan algoritma, dan visualisasi data. Hasil proyek ini juga dapat menjadi referensi atau baseline bagi penelitian selanjutnya yang ingin mengembangkan sistem rekomendasi dengan fitur interaktivitas tinggi dan fokus pada optimasi performa aplikasi.

1.6 Ruang Lingkup dan Batasan

1.6.1 Ruang Lingkup

1. Pengumpulan dan Preparasi Data: Pengumpulan data destinasi wisata dilakukan melalui dua sumber, yaitu SISPARNAS Provinsi Lampung untuk atribut nama destinasi, kategori wisata, kabupaten/kota, dan fasilitas; serta Google Maps untuk memperoleh koordinat geografis (latitude–longitude)

dan rating destinasi. Proses persiapan data meliputi pembersihan data, penyelarasan format, serta penanganan atribut kategorikal sesuai kebutuhan.

2. *Exploratory Data Analysis* (EDA): Analisis eksploratif dilakukan untuk memahami karakteristik destinasi wisata di Lampung melalui distribusi kategori wisata, pemerataan fasilitas, persebaran destinasi. Visualisasi dapat dilakukan menggunakan grafik dan tabel.
3. *Feature Engineering*: Proses rekayasa fitur mencakup konversi atribut kategorikal menjadi vektor numerik melalui one-hot encoding. One-hot encoding digunakan pada atribut kategori wisata dan fasilitas untuk menghasilkan binary feature vector yang dapat diolah oleh algoritma kemiripan. Selain itu, fitur jarak dihitung menggunakan rumus *Haversine*, kemudian dinormalisasi untuk menyamakan skala.
4. Pengembangan Model: Pengembangan model dilakukan menggunakan pendekatan *Content-Based Filtering* berbasis *cosine similarity* untuk mengukur kemiripan antara preferensi pengguna dan atribut destinasi. Nilai kemiripan kemudian digabungkan dengan nilai jarak hasil perhitungan *Location-Based Service* (LBS). Model dirancang untuk menghasilkan daftar rekomendasi berdasarkan *aggregated score* yang mempertimbangkan bobot atribut.
5. Evaluasi dan Validasi: Evaluasi sistem rekomendasi dilakukan menggunakan metrik precision, recall, dan F1-score untuk menilai relevansi rekomendasi terhadap preferensi pengguna. Verifikasi jarak antar destinasi dengan membandingkan hasil jarak *Haversine* dengan jarak yang ditampilkan *Google Maps* pada sejumlah sampel destinasi. Hasil evaluasi ini digunakan untuk menilai ketepatan rekomendasi dan kesesuaian perhitungan jarak dalam sistem.
6. Implementasi dan Deployment: Sistem rekomendasi yang dihasilkan diimplementasikan dalam bentuk dashboard interaktif berbasis Streamlit. Implementasi mencakup pemrosesan input pengguna, eksekusi model rekomendasi secara langsung, menampilkan daftar rekomendasi, serta visualisasi lokasi wisata.
7. Analisis Dampak Bisnis: Penelitian ini menghasilkan dokumentasi teknis terkait alur pengembangan model, mekanisme rekomendasi, arsitektur dashboard, serta interpretasi hasil rekomendasi. Analisis dilakukan untuk menilai kelebihan, keterbatasan, serta peluang pengembangan sistem pada penelitian selanjutnya.

1.6.2 Batasan

Batasan dalam proyek ini adalah:

1. Periode Data: Data yang digunakan merupakan data terakhir yang tersedia pada sumber data pada saat proses pengambilan data dilakukan.
2. Sumber Data: Data destinasi wisata pada Provinsi Lampung terbatas pada informasi yang tersedia di Sistem Informasi Pariwisata Nasional (SISPARNAS), sehingga cakupan destinasi hanya mencerminkan objek wisata yang tercatat pada platform tersebut. Informasi tambahan berupa koordinat geografis dan rating destinasi diperoleh dari *Google Maps*, sehingga kualitas dan kelengkapan data bergantung pada akurasi serta ketersediaan informasi pada layanan tersebut.
3. Scope Implementasi: Pengembangan sistem rekomendasi dibatasi pada tahap pembuatan prototipe yang di-deploy melalui dashboard Streamlit, tanpa mencakup implementasi penuh pada lingkungan produksi atau integrasi mendalam dengan sistem Pariwisata daerah maupun server resmi pemerintah.
4. Integrasi Sistem: Integrasi sistem dibatasi pada penggabungan data dari dua sumber, yaitu SISPARNAS Provinsi Lampung dan *Google Maps*, tanpa melakukan koneksi langsung melalui API atau layanan integrasi otomatis. Sistem rekomendasi hanya mengolah data yang telah dikumpulkan dan dipersiapkan sebelumnya.
5. Algoritma dan Teknologi: Penelitian menggunakan algoritma *Content-Based Filtering* dan pemanfaatan *Location-Based Service* (LBS) untuk perhitungan jarak geografis. Teknologi yang digunakan dibatasi pada perangkat lunak open-source serta pustaka yang tersedia dalam lingkungan pengembangan yang digunakan, tanpa melakukan eksperimen dengan algoritma rekomendasi lain seperti *hybrid filtering, collaborative filtering,* atau model berbasis machine learning yang lebih kompleks.
6. Sumber Daya: Pengembangan sistem dibatasi oleh keterbatasan komputasi, waktu pengerjaan proyek, dan anggaran yang telah dialokasikan. Pengujian serta optimasi kinerja dilakukan sesuai kapasitas perangkat dan sumber daya yang tersedia, sehingga evaluasi tidak mencakup pengujian berskala besar atau pemanfaatan infrastruktur komputasi tingkat lanjut.
7. Compliance: Seluruh proses pengumpulan dan pengolahan data mengikuti ketentuan privasi, kebijakan penggunaan data publik, dan etika pemanfaatan API dari masing-masing platform. Penelitian tidak melakukan pengambilan data yang melanggar aturan, scraping ilegal, atau penggunaan data pribadi

pengguna.

1.7 Sistematika Penulisan

Laporan ini disusun dengan sistematika sebagai berikut:

1.7.1 BAB I PENDAHULUAN

Berisi latar belakang, rumusan masalah, tujuan, sasaran, manfaat, ruang lingkup, dan batasan proyek.

1.7.2 BAB II TINJAUAN PUSTAKA

Berisi landasan teori, penelitian terdahulu, dan kerangka pemikiran yang mendukung proyek.

1.7.3 BAB III PERENCANAAN PROYEK

Berisi perencanaan lengkap meliputi metodologi, jadwal, anggaran, dan alokasi sumber daya.

1.7.4 BAB IV PELAKSANAAN PROYEK

Berisi proses pelaksanaan proyek mulai dari pengumpulan data hingga pengembangan model.

1.7.5 BAB V HASIL DAN PEMBAHASAN

Berisi hasil yang diperoleh dan analisis mendalam terhadap hasil tersebut.

1.7.6 BAB VI PENUTUP

Berisi kesimpulan dan saran untuk pengembangan lebih lanjut.

BAB II

PERENCANAAN PROYEK

2.1 Metodologi Proyek

2.1.1 Pendekatan Proyek

Dalam pelaksanaan penelitian ini, pendekatan yang digunakan mengacu pada kerangka kerja standar industri yaitu CRISP-DM *(Cross-Industry Standard Process for Data Mining)*. Metodologi ini dipilih karena memiliki struktur sistematis yang mencakup seluruh siklus hidup pengembangan sistem berbasis data, mulai dari pemahaman kebutuhan bisnis hingga implementasi akhir dalam bentuk aplikasi. Berdasarkan metodologi CRISP-DM, tahapan pengerjaan proyek ini terdiri dari enam fase sebagai berikut:

1. *Business Understanding*
2. *Data Understanding*
3. *Data Preparation*
4. *Modeling*
5. *Evaluation*

2.1.2 Tahapan Pelaksanaan

Pelaksanaan proyek ini dilakukan secara bertahap mengikuti alur metodologi CRISP-DM yang terdiri dari beberapa fase. Setiap fase mencakup rangkaian kegiatan yang disusun secara sistematis dan dilaksanakan secara berurutan, dengan estimasi waktu yang telah ditetapkan pada masing-masing fase sebagai berikut:

1. Fase 1: Business Understanding (Minggu 1-2)
   Fase ini berfokus pada identifikasi masalah utama, yaitu kesulitan wisatawan dalam menemukan destinasi wisata di Lampung yang sesuai dengan preferensi dan lokasi mereka. Tujuan tahap ini adalah mendefinisikan kebutuhan sistem rekomendasi yang mampu meningkatkan visibilitas objek wisata lokal serta membantu wisatawan merencanakan perjalanan dengan lebih efisien.
2. Fase 2: Data Understanding (Minggu 3-4)
   Pada fase ini, dilakukan proses pengumpulan data mentah terkait destinasi wisata di Provinsi Lampung. Kegiatan mencakup eksplorasi awal untuk memahami atribut data yang tersedia (seperti nama wisata, kategori,

deskripsi fasilitas, rating, dan titik koordinat), serta mengidentifikasi kualitas data guna menentukan langkah pembersihan yang diperlukan.

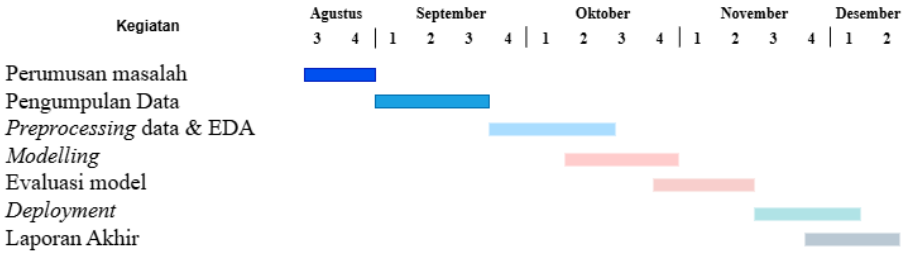
3. Fase 3: Data preparation (Minggu 3-4)
   Fase ini melibatkan proses preprocessing untuk mengubah data mentah menjadi format yang siap dimodelkan. Aktivitas utamanya meliputi pembersihan data *(data cleaning)*, penanganan nilai yang hilang *(missing values)*, normalisasi teks untuk kebutuhan *Content-Based Filtering*, serta validasi format koordinat *(latitude/longitude)* untuk kebutuhan *Location-Based Service*.
4. Fase 4: Modellimg (Minggu 5-6)
   Fase ini merupakan inti dari pengembangan sistem, di mana algoritma sistem rekomendasi dibangun. Peneliti menerapkan metode Content-Based Filtering menggunakan perhitungan Cosine Similarity untuk mencari kemiripan karakteristik wisata, serta mengintegrasikan rumus Haversine dalam fitur Location-Based Service (LBS) untuk menghitung jarak akurat antar destinasi.
5. Fase 5: Evaluation (Minggu 6-7)
   Setelah model terbentuk, dilakukan evaluasi menyeluruh untuk mengukur kualitas rekomendasi yang dihasilkan. Evaluasi tidak hanya berfokus pada relevansi rekomendasi, tetapi juga pada performa teknis sistem, khususnya pengukuran response time (waktu respons) untuk memastikan dashboard dapat menyajikan data secara cepat tanpa jeda yang signifikan.
6. Fase 6: Deployment (Minggu 8)
   Fase akhir adalah implementasi model ke dalam bentuk dashboard interaktif berbasis web. Dashboard dijalankan pada lingkungan lokal *(localhost)* dan dirancang untuk menampilkan hasil rekomendasi destinasi wisata berdasarkan preferensi pengguna dan lokasi. Tahap ini bertujuan untuk memastikan dashboard sistem rekomendasi dapat berjalan dengan baik secara fungsional serta siap digunakan untuk keperluan pengujian dan demonstrasi.

2.2 Jadwal dan Timeline Proyek

Jadwal dan timeline proyek disusun untuk memastikan seluruh tahapan pengembangan sistem rekomendasi destinasi wisata dapat berjalan secara terstruktur dan tepat waktu.

2.2.1 *Gantt Chart*

Setiap kegiatan disusun secara berurutan dan saling berkaitan untuk memastikan proses pengembangan sistem rekomendasi berjalan secara sistematis digambarkan pada *Gantt Chart* berikut.



Gambar 2.1 Gantt Chart Timeline Proyek

2.2.2 Milestone Proyek

Milestone proyek ditetapkan sebagai titik pencapaian utama pada setiap tahapan kegiatan proyek. Penentuan *Milestone* bertujuan untuk memastikan bahwa setiap fase proyek dapat diselesaikan sesuai dengan target yang telah direncanakan.

Tabel 2.1 Milestone Pelaksanaan Proyek

No	Milestone	Target Waktu
1	Data terkumpul	Minggu ke-10
2	Preprocessing dan EDA selesai	Minggu ke-12
3	Model sistem rekomendasi terbentuk	Minggu ke-14
4	Evaluasi dan deployment sistem selesai	Minggu ke-16
5	Laporan akhir selesai	Minggu ke-16

2.3 Estimasi Biaya dan Anggaran

Estimasi biaya dan anggaran proyek disusun untuk memastikan seluruh kebutuhan pelaksanaan proyek sistem rekomendasi destinasi wisata di Lampung terpenuhi secara efisien dan terkontrol sesuai dengan ruang lingkup dan kebutuhan proyek.

Tabel 2.2 Rincian Anggaran Proyek

Kategori	**Item**	**Biaya (Rp)**
Data	Pengumpulan Data	0
Operasional	Paket data / internet	100.000
	Dokumentasi dan administrasi	50.000
Contingency	Cadangan biaya (10%)	45.000
	TOTAL	**195.000**

2.4 Alokasi Sumber daya

2.4.1 Struktur Tim Proyek

Struktur tim proyek penelitian ini terdiri dari tiga peran utama, yaitu Data Scientist, Data Engineer, dan Data Analyst, yang masing-masing memiliki tanggung jawab berbeda dalam mendukung pelaksanaan dan keberhasilan proyek sebagaimana yang disajikan pada tabel 2.3 berikut.

Tabel 2.3 Struktur Tim Proyek

Nama	**Role**	**Tanggung Jawab**
Ukasyah Muntaha	*Data Engineer*	Mengumpulkan data destinasi wisata, menyiapkan data agar siap diolah, serta melakukan deployment prototipe sistem rekomendasi berbasis Streamlit pada lingkungan lokal (localhost).
Nurul Alfajar Gumel	*Data Scientist*	Merancang dan mengembangkan model sistem rekomendasi berbasis content-based filtering dengan integrasi location-based service (LBS), serta mengevaluasi hasil rekomendasi menggunakan skor similarity.
Syalaisha Andina Putriansyah	*Data Analyst*	Melakukan exploratory data analysis (EDA) serta mengevaluasi kinerja sistem rekomendasi.

2.4.2 Matriks RACI

Pembagian peran dalam setiap aktivitas proyek dirumuskan menggunakan Matriks RACI untuk memastikan kejelasan tanggung jawab dan koordinasi antar anggota tim.

Tabel 2.4 Matriks RACI Pembagian Peran Proyek

Aktivitas	***Data Engineer***	***Data scientist***	***Data Analyst***
Perencanaan proyek	I	R	C
Pengumpulan data	R	I	C
Preprocessing data dan EDA	R	C	R
Pemodelan	C	R	I
Evaluasi dan deployment	R	C	R

2.5 Kriteria Keberhasilan

Kriteria keberhasilan proyek digunakan sebagai acuan dalam menilai ketercapaian proyek Sistem Rekomendasi Destinasi Wisata di Provinsi Lampung berbasis *Content-Based Filtering* dan *Location-Based Service* sesuai tujuan.

2.5.1 *Technical Success*

Proyek dinyatakan berhasil secara teknis apabila:

1. Data destinasi wisata terkumpul minimal 80% dari total destinasi di Provinsi Lampung
2. Model sistem rekomendasi dan dapat dijalankan tanpa kesalahan (*error*).
3. Sistem mampu menghasilkan minimal 5 rekomendasi destinasi wisata yang relevan untuk setiap preferensi pengguna.

2.5.2 *Project Success*

Proyek dinyatakan berhasil dari sisi pelaksanaan apabila:

1. Seluruh tahapan proyek terlaksana sesuai dengan perencanaan dan timeline yang telah ditetapkan.
2. *Deployment* sistem rekomendasi berbasis Streamlit pada lingkungan lokal *(localhost)* berhasil dilakukan sebagai *prototipe* sistem.

2.5.3 *Business Success*

Proyek dinyatakan berhasil dari sisi manfaat apabila:

1. Sistem rekomendasi mampu membantu pengguna dalam memperoleh

rekomendasi destinasi wisata di Provinsi Lampung yang sesuai dengan preferensi pribadi.

2. Luaran penelitian relevan untuk pengembangan penelitian lanjutan.

BAB III

PELAKSANAAN PROYEK

3.1 Pengumpulan Data

3.1.1 Sumber Data

Pada proyek ini, data yang digunakan merupakan data destinasi tempat wisata yang ada di Provinsi Lampung. Dataset dikumpulkan dari beberapa sumber terbuka salah satunya adalah situs Sistem Informasi Kepariwisataan Nasional. Sistem Informasi Kepariwisataan Nasional ini mencakup seluruh destinasi tempat wisata yang ada di seluruh Indonesia.

3.1.2 Deskripsi Dataset

Dataset yang digunakan dalam proyek ini berisi informasi utama setiap destinasi wisata di Provinsi Lampung. Ringkasan karakteristik dataset yang digunakan ditampilkan pada tabel 3.1 berikut.

Tabel 3.1 Karakteristik Dataset

Atribut	**Keterangan**
Jumlah data	439 destinasi tempat wisata
Jumlah fitur	8 kolom
Cakupan wilayah	Provinsi Lampung
Missing values (rating)	6 data (1.14%)
Duplikasi data	Tidak ditemukan

Dataset destinasi tempat wisata Provinsi Lampung terdiri dari sejumlah entri destinasi wisata yang merepresentasikan objek wisata di seluruh kabupaten/kota yang ada di Provinsi Lampung. Setiap baris data merepresentasikan satu destinasi wisata, sedangkan kolom merepresentasikan atribut atau fitur dari destinasi tersebut. Dataset yang digunakan mencakup informasi umum mengenai destinasi wisata, seperti nama tempat wisata, kategori wisata, lokasi administratif (kabupaten/kota), fasilitas yang tersedia, rata-rata rating pengunjung, serta koordinat geografis (longitude & latitude).

3.1.3 Variabel Penelitian

Penelitian ini melibatkan sejumlah variabel yang menjadi fokus analisis. Berikut adalah daftar variabel yang digunakan dalam penelitian beserta tipe data dan deskripsinya, seperti ditampilkan pada tabel berikut.

Tabel 3.2 Daftar Variabel Penelitian

No	Nama Variabel	Tipe Data	Deskripsi
1	Nama tempat	Categorical	Nama destinasi wisata di Provinsi Lampung
2	kabupaten kota	Categorical	Lokasi administratif destinasi wisata
3	kategori	Categorical	Jenis atau kategori wisata (alam, budaya, buatan, religi, lainnya.)
4	latitude	Float	Koordinat lintang destinasi wisata
5	longitude	Float	Koordinat bujur destinasi wisata
6	Fasilitas	Categorical	Informasi fasilitas yang tersedia pada destinasi wisata
7	rating	Float	Penilaian pengunjung terhadap destinasi wisata

3.2 *Exploratory Data Analysis* (EDA)

3.2.1 Analisis Data Statistik

Pada proyek ini, analisis statistik dilakukan untuk memberikan gambaran umum mengenai variabel numerik yang digunakan. Ringkasan statistik deskriptif disajikan pada Tabel 3.3.

Tabel 3.3 Statistik Deskriptif Rating Destinasi Wisata

Statistik	Nilai
Rata-rata (Mean)	4.40
Nilai Minimum	1.0
Nilai Maksimum	5.0
Jumlah Missing Value	6

3.2.2 Distribusi Data

Distribusi data pada variabel kategorikal yang digunakan dalam penelitian ini disajikan pada Tabel 3.4 dan Tabel 3.5

Tabel 3.4 Distribusi Destinasi Wisata Berdasarkan Kategori

Kategori Wisata	Jumlah Destinasi
Wisata Buatan	202
Wisata Alam	195
Wisata Budaya	28
Wisata Lainnya	8
Wisata Religi	6

Tabel 3.5 Distribusi Destinasi Wisata Berdasarkan Kabupaten/Kota

Kabupaten/Kota	Jumlah Destinasi
Tanggamus	57
Pesawaran	39
Lampung Selatan	37
Pringsewu	36
Bandar Lampung	35
Lampung Timur	33
Lampung Tengah	31
Lampung Barat	29
Tulang Bawang	29
Lampung Utara	25
Tulang Bawang Barat	20
Mesuji	19
Way Kanan	17
Metro	17
Pesisir Barat	15

3.2.3 Analisis Korelasi

Analisis korelasi tidak dilakukan pada projek ini karena sebagian besar variabel dalam dataset bersifat kategorikal dan deskriptif. Variabel numerik yang ada pada dataset ini yaitu rating, longitude, dan latitude sehingga tidak memungkinkan dilakukan pengujian korelasi antar variabel numerik secara statistik.

3.2.4 Insight dari EDA

Berdasarkan hasil Exploratory Data Analysis (EDA) terhadap dataset destinasi wisata di Provinsi Lampung, diperoleh sejumlah insight penting yang menjadi dasar dalam pengembangan sistem rekomendasi wisata. Analisis ini memberikan

gambaran mengenai karakteristik data, distribusi destinasi, serta kualitas informasi yang tersedia dalam dataset.

Dari sisi kategori wisata, diketahui bahwa destinasi wisata di Provinsi Lampung didominasi oleh wisata buatan dan wisata alam. Kedua kategori ini memiliki jumlah destinasi yang jauh lebih besar dibandingkan kategori wisata budaya, religi, dan kategori lainnya. Hal ini menunjukkan bahwa potensi pariwisata di Provinsi Lampung lebih berfokus pada wisata rekreasi dan alam terbuka. Dominasi kategori ini menjadi pertimbangan penting dalam perancangan sistem rekomendasi, karena rekomendasi yang dihasilkan akan lebih optimal apabila memanfaatkan kategori dengan cakupan data yang luas.

Berdasarkan persebaran wilayah, jumlah destinasi wisata tidak terdistribusi secara merata di setiap kabupaten/kota. Kabupaten Tanggamus tercatat memiliki jumlah destinasi wisata terbanyak, diikuti oleh Kabupaten Pesawaran dan Lampung Selatan. Sementara itu, beberapa wilayah lain memiliki jumlah destinasi yang relatif lebih sedikit. Insight ini menunjukkan adanya wilayah dengan potensi wisata yang lebih dominan, sehingga informasi lokasi menjadi faktor penting dalam memberikan rekomendasi destinasi yang relevan bagi pengguna.

Analisis terhadap data rating menunjukkan bahwa secara umum destinasi wisata memiliki tingkat penilaian yang tinggi, dengan nilai rata-rata rating sebesar 4.39 dan rentang nilai antara 1 hingga 5. Tingginya nilai rata-rata rating mengindikasikan bahwa sebagian besar destinasi wisata mendapatkan respons positif dari pengunjung. Namun, ditemukan pula sejumlah data rating yang tidak tersedia (missing value). Oleh karena itu, diperlukan penanganan khusus terhadap data rating agar tidak memengaruhi kinerja model rekomendasi.

Secara keseluruhan, struktur dataset yang didominasi oleh data kategorikal seperti kategori wisata, kabupaten/kota, dan fasilitas, serta didukung oleh informasi geografis dan rating, menunjukkan bahwa dataset ini sangat sesuai untuk dikembangkan menggunakan pendekatan Content-Based Filtering. Penggabungan informasi kesamaan konten dan jarak geografis dalam model hybrid memungkinkan sistem rekomendasi memberikan hasil yang lebih personal, relevan, dan kontekstual sesuai dengan preferensi pengguna.

3.3 Data Preprocessing

3.3.1 Penanganan Missing Values

Missing value pada kolom rating ditangani dengan cara tidak dilakukan imputasi, melainkan diabaikan karena jumlahnya sangat kecil dan rating bukan merupakan fitur utama dalam model yang dibangun untuk sistem rekomendasi ini.

3.3.2 Transformasi Data

Pada tahap ini dilakukan transformasi data terhadap dua fitur utama, yaitu *kategori* dan *fasilitas*, agar keduanya dapat direpresentasikan dalam bentuk numerik dan digunakan pada proses perhitungan kesamaan dalam sistem rekomendasi. Fitur *kategori* yang bersifat kategorikal tunggal ditransformasikan menggunakan teknik *one-hot encoding*. Setiap kategori wisata direpresentasikan ke dalam variabel biner, di mana setiap destinasi wisata hanya memiliki satu nilai kategori dengan nilai 1 pada kolom kategori yang sesuai dan 0 pada kolom lainnya. Selain itu, fitur *fasilitas* yang bersifat multi-label juga ditransformasikan ke dalam bentuk representasi biner. Setiap jenis fasilitas direpresentasikan sebagai satu kolom tersendiri dengan nilai 1 jika fasilitas tersedia pada destinasi wisata tersebut dan 0 jika fasilitas tidak tersedia. Transformasi ini bertujuan untuk menyederhanakan informasi kategori dan fasilitas sehingga dapat digunakan secara efektif dalam proses pemodelan sistem rekomendasi berbasis kesamaan.

Tabel 3.6 Contoh Data Hasil Transformasi Kategori

Nama Tempat	K1	K2	K3	K4	K5
Air Terjun Cengkaan	1	0	0	0	0
Air Terjun Semantung	1	0	0	0	0
Curug Gimo	1	0	0	0	0
Danau Asam Suoh	1	0	0	0	0

Tabel 3.7 Keterangan Kode Kategori Wisata

Kode	**Kategori**
K1	Wisata Alam
K2	Wisata Buatan
K3	Wisata Budaya
K4	Wisata Religi
K5	Wisata Lainnya

Tabel 3.8 Contoh Data Hasil Transformasi Fasilitas

Nama Tempat	F1	F2	F3	F4
Air Terjun Cengkaan	1	1	1	1
Air Terjun Semantung	1	0	0	0
Curug Gimo	1	1	1	0
Danau Asam Suoh	1	1	1	1

Tabel 3.9 Keterangan Kode Fasilitas Wisata

Kode	**Fasilitas**
F1	Parkir
F2	Restoran
F3	Toilet
F4	Mushola

3.4 Pengembangan Model

3.4.1 Baseline Model

Baseline model dibangun sebagai acuan awal untuk mengevaluasi performa sistem rekomendasi sebelum dilakukan pengembangan lebih lanjut. Pada proyek ini, baseline model yang digunakan adalah *Content-Based Filtering* dengan pendekatan *cosine similarity*. Model ini merekomendasikan destinasi wisata berdasarkan tingkat kemiripan fitur antar destinasi, tanpa mempertimbangkan interaksi pengguna lain. Setiap destinasi direpresentasikan dalam bentuk vektor fitur numerik. Nilai kemiripan antara dua destinasi dihitung menggunakan *cosine similarity*, di mana semakin tinggi nilai kemiripan, maka semakin relevan destinasi tersebut untuk direkomendasikan. Baseline model ini berfungsi sebagai pembanding untuk menilai peningkatan performa pada model yang lebih kompleks.

3.4.2 Eksperimen dengan *Multiple Algorithms*

Pada tahap ini dilakukan eksperimen pengembangan sistem rekomendasi dengan mengombinasikan dua pendekatan algoritma yang berbeda, yaitu *Content-Based Filtering (CBF)* dan *Location-Based Services (LBS)*. Pendekatan ini bertujuan untuk menghasilkan rekomendasi yang tidak hanya relevan secara konten, tetapi juga mempertimbangkan kedekatan lokasi geografis antar destinasi wisata.

Content-Based Filtering digunakan untuk mengukur kemiripan antar destinasi berdasarkan atribut kategori dan fasilitas. Sementara itu, *Location-Based Services*

digunakan untuk menghitung jarak geografis antar destinasi berdasarkan koordinat lintang (latitude) dan bujur (longitude). Jarak tersebut kemudian dikonversi menjadi skor kedekatan lokasi, di mana destinasi dengan jarak yang lebih dekat akan memiliki skor yang lebih tinggi. Kombinasi antara CBF dan LBS diharapkan mampu menghasilkan rekomendasi yang lebih kontekstual, di mana faktor jarak dan aksesibilitas lokasi menjadi aspek penting dalam pengambilan keputusan pengguna.

3.4.3 Hyperparameter Tuning

Tahap hyperparameter tuning dilakukan untuk mengoptimalkan kinerja model hybrid dengan menyesuaikan bobot kontribusi masing-masing komponen rekomendasi. Pada penelitian ini, digunakan tiga parameter utama, yaitu:

- α_1 : bobot untuk skor kemiripan kategori wisata
- α_2 : bobot untuk skor kemiripan fasilitas wisata
- β : bobot untuk skor kedekatan lokasi berbasis LBS

Penentuan nilai parameter dilakukan melalui eksperimen dengan beberapa kombinasi bobot. Kombinasi parameter terbaik dipilih berdasarkan pengamatan terhadap relevansi hasil rekomendasi yang dihasilkan, dengan tujuan untuk menyeimbangkan pengaruh kemiripan konten dan kedekatan lokasi dalam sistem rekomendasi wisata.

3.5 Testing dan Validasi

3.5.1 Unit Testing

Unit testing dilakukan untuk menguji setiap fungsi utama yang digunakan dalam sistem rekomendasi secara terpisah. Tujuan dari unit testing adalah untuk memastikan bahwa setiap fungsi dapat berjalan dengan benar sesuai dengan tujuan perancangannya sebelum diintegrasikan ke dalam sistem secara keseluruhan.

Pada penelitian ini, unit testing difokuskan pada beberapa fungsi utama, yaitu:

- Fungsi pembersihan dan prapemrosesan data
- Fungsi transformasi fitur (one-hot encoding kategori dan fasilitas)
- Fungsi perhitungan jarak berbasis Location-Based Services (LBS)
- Fungsi perhitungan skor rekomendasi hybrid

Pengujian dilakukan dengan memberikan input data yang valid dan memeriksa apakah keluaran yang dihasilkan sesuai dengan ekspektasi. Sebagai contoh, fungsi

perhitungan jarak diuji dengan dua koordinat geografis yang diketahui untuk memastikan hasil jarak yang dihasilkan masuk akal. Selain itu, fungsi rekomendasi diuji untuk memastikan bahwa sistem dapat mengembalikan sejumlah destinasi rekomendasi tanpa menghasilkan kesalahan (*error*) saat dijalankan. Hasil dari unit testing menunjukkan bahwa seluruh fungsi utama dapat berjalan dengan baik dan tidak ditemukan kesalahan logika yang signifikan. Dengan demikian, sistem siap untuk dilanjutkan ke tahap pengujian integrasi dan validasi hasil rekomendasi secara keseluruhan.

BAB IV

HASIL DAN PEMBAHASAN

4.1 *Modeling*

4.1.1 *Pra processing*

Pra processing pertama yang dilakukan adalah pengecekan kategori unik dalam dataset dan pembersihan fasilitas yang ada. Kategori yang ditemukan adalah sebagai berikut:

Tabel 4.1 Distribusi Kategori dalam Dataset

Kategori	Jumlah Destinasi
Wisata Alam	199
Wisata Buatan	193
Wisata Budaya	28
Wisata Lainnya	7
Wisata Religi	6

Distribusi kategori menunjukkan bahwa kategori "Wisata Alam" dan "Wisata Buatan" mendominasi, dengan jumlah destinasi yang jauh lebih banyak dibandingkan dengan kategori lainnya.Selain itu, setelah dilakukan *preprocessing*, ditemukan empat jenis fasilitas yang tersedia pada destinasi wisata. Seluruh fasilitas tersebut kemudian diterjemahkan menjadi fitur biner (0/1) melalui proses *one-hot encoding*, sehingga setiap destinasi dapat direpresentasikan berdasarkan ada/tidaknya fasilitas berikut:

- **Parkir**: menunjukkan apakah destinasi menyediakan area parkir.
- **Restoran**: menunjukkan apakah destinasi memiliki fasilitas restoran/rumah makan.
- **Toilet**: menunjukkan apakah destinasi menyediakan fasilitas toilet.
- **Mushola**: menunjukkan apakah destinasi menyediakan mushola/tempat ibadah.

Dengan representasi biner ini, proses analisis dan pemodelan menjadi lebih terstruktur karena fitur fasilitas dapat langsung digunakan sebagai variabel input.

4.1.2 One-Hot Encoding Kategori dan Fasilitas

Kategori dan fasilitas telah diubah menjadi format biner melalui proses *one-hot encoding*. Berikut adalah contoh hasil *one-hot encoding* untuk kategori dan fasilitas.

Tabel 4.2 One-hot Encoding Kategori

Nama Tempat	K1	K2	K3	K4	K5
Air Terjun Cengkaan	1	0	0	0	0
Air Terjun Semantung	1	0	0	0	0
Curug Gimo	1	0	0	0	0

Berdasarkan Tabel 4.2, nilai 1 menunjukkan bahwa suatu destinasi termasuk ke dalam kategori tertentu, sedangkan nilai 0 menunjukkan bahwa destinasi tidak termasuk dalam kategori tersebut. Pada contoh yang ditampilkan, seluruh destinasi memiliki nilai 1 pada K1 dan 0 pada K2–K5, sehingga seluruhnya terklasifikasi pada kategori K1.

Tabel 4.3 One-hot Encoding Fasilitas

Nama Tempat	F1	F2	F3	F4
Air Terjun Cengkaan	1	1	1	1
Air Terjun Semantung	1	0	0	0
Curug Gimo	1	1	1	0

Mengacu pada Tabel 4.3, nilai 1 menunjukkan fasilitas tersedia dan nilai 0 menunjukkan fasilitas tidak tersedia pada destinasi tersebut. Air Terjun Cengkaan memiliki seluruh fasilitas (F1–F4) yang ditandai dengan nilai 1 pada semua kolom. Air Terjun Semantung hanya memiliki F1, sedangkan fasilitas lain (F2–F4) tidak tersedia. Curug Gimo memiliki F1, F2, dan F3, tetapi tidak memiliki F4.

4.1.3 Evaluasi Similaritas (CBF)

Setelah tahap pemrosesan fitur kategori dan fasilitas selesai dilakukan, langkah berikutnya adalah menghitung tingkat kemiripan (*similarity*) antar destinasi wisata berdasarkan representasi fitur tersebut. Pengukuran kemiripan pada penelitian ini menggunakan *Cosine Similarity*, karena mampu merepresentasikan kedekatan antar vektor fitur biner secara efisien dan konsisten untuk kebutuhan *content-based filtering* (CBF).

Tabel 4.4 Ringkasan Evaluasi Similaritas Berdasarkan Kategori

Evaluasi	Nilai
Intra Mean	1.0
Inter Mean	0.0
Separation	1.0

Berdasarkan tabel 4.4, nilai *intra mean* sebesar 1.0 mengindikasikan bahwa destinasi yang berada pada kategori yang sama memiliki kemiripan yang sangat tinggi. Sebaliknya, nilai *inter mean* sebesar 0.0 menunjukkan bahwa destinasi dari

kategori yang berbeda memiliki tingkat kemiripan yang sangat rendah. Nilai *separation* sebesar 1.0 mempertegas adanya pemisahan yang sangat jelas antara kelompok destinasi dalam kategori yang sama dan destinasi dari kategori berbeda. Kondisi ini menunjukkan bahwa fitur kategori mampu membedakan destinasi secara tegas, sehingga sistem rekomendasi cenderung menghasilkan rekomendasi yang sangat terfokus pada destinasi dengan kategori yang identik.

Tabel 4.5 Evaluasi Coverage dan Diversity pada Rekomendasi

Evaluasi	Nilai
Coverage	1.0
Diversity	0.0

Berdasarkan tabel 4.5, nilai *coverage* sebesar 1.0 menunjukkan bahwa seluruh destinasi (100%) memperoleh setidaknya satu rekomendasi yang memenuhi ambang batas (*threshold*) kemiripan yang ditetapkan. Dengan demikian, model mampu memberikan cakupan rekomendasi yang menyeluruh pada data yang digunakan. Namun, nilai *diversity* sebesar 0.0 mengindikasikan bahwa rekomendasi yang dihasilkan memiliki variasi yang sangat rendah. Hal ini mengarah pada pola rekomendasi yang homogen, karena model secara dominan memprioritaskan destinasi yang memiliki kemiripan sangat tinggi (misalnya kategori yang sama), sehingga daftar rekomendasi antar pengguna atau antar item cenderung serupa.

4.1.4 Hasil Rekomendasi Hybrid (CBF + LBS)

Setelah menerapkan pendekatan *hybrid* yang menggabungkan *content-based filtering* (CBF) dan *location-based service* (LBS), sistem menghasilkan rekomendasi destinasi yang mempertimbangkan tiga dimensi utama, yaitu kemiripan kategori, kesesuaian fasilitas, dan kedekatan lokasi geografis. Dengan integrasi tersebut, rekomendasi tidak hanya bergantung pada kesamaan atribut konten, tetapi juga memperhitungkan konteks spasial sehingga hasil yang diberikan lebih relevan untuk kebutuhan perjalanan.

Tabel 4.6 Contoh Hasil Rekomendasi Hybrid (CBF + LBS)

Nama Tempat	Kategori	Score Hybrid	Score Kategori	Score Fasilitas	Score Lokasi
Wisata Alam Batu Langit	Wisata Alam	0.834	1.0	1.0	0.17
Way Besai Rafting	Wisata Alam	0.832	1.0	1.0	0.16
Wisata Cai Kahuripan	Wisata Alam	0.829	1.0	1.0	0.14

Berdasarkan Tabel 4.6, ketiga destinasi yang direkomendasikan memiliki *score kategori* dan *score fasilitas* yang maksimum (1.0), yang menunjukkan adanya kesesuaian konten yang sangat kuat dengan destinasi acuan atau preferensi yang

diuji. Variasi nilai *score lokasi* (0.14–0.17) memperlihatkan bahwa perbedaan kedekatan geografis ikut memengaruhi pemeringkatan akhir, sehingga *score hybrid* menjadi pembobot gabungan yang menyeimbangkan kemiripan konten dan konteks lokasi.

4.1.5 Evaluasi Sistem Rekomendasi Hybrid

Evaluasi performa sistem *hybrid* dilakukan untuk menilai sejauh mana rekomendasi yang dihasilkan bersifat relevan dan merata, sekaligus mengukur tingkat variasi rekomendasi yang diberikan. Pengukuran dilakukan melalui metrik *intra mean*, *inter mean*, *separation*, *coverage*, dan *diversity*.

Tabel 4.7 Evaluasi Sistem Rekomendasi Hybrid

Evaluasi	Nilai
Intra Mean	0.782
Inter Mean	0.278
Separation	0.504
Coverage	1.0
Diversity	0.173

Mengacu pada Tabel 4.7, nilai *intra mean* sebesar 0.782 lebih tinggi dibandingkan *inter mean* sebesar 0.278, yang menunjukkan bahwa destinasi dalam kelompok yang relatif serupa (misalnya kategori/fitur yang sejalan) cenderung memiliki kedekatan yang lebih tinggi dibandingkan destinasi yang berbeda. Nilai *separation* sebesar 0.504 menunjukkan adanya jarak pemisah yang cukup jelas antara kemiripan dalam kelompok dan antar kelompok, sehingga sistem memiliki kemampuan diskriminasi yang memadai dalam proses pemeringkatan rekomendasi.

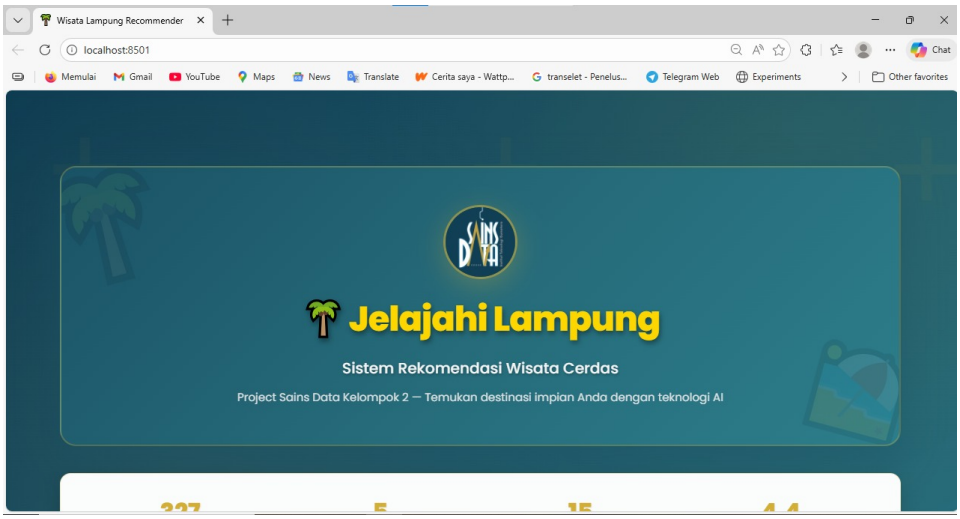
Selain itu, nilai *coverage* sebesar 1.0 mengindikasikan bahwa seluruh destinasi memperoleh setidaknya satu rekomendasi yang memenuhi kriteria, sehingga sistem tidak hanya efektif pada sebagian kecil item, tetapi mampu memberikan cakupan rekomendasi yang menyeluruh. Namun, nilai *diversity* sebesar 0.173 menandakan bahwa variasi rekomendasi masih relatif terbatas. Hal ini mengimplikasikan bahwa model tetap cenderung mengarahkan rekomendasi pada destinasi yang mirip, meskipun komponen LBS telah meningkatkan konteks spasial dan sedikit memperluas variasi dibandingkan pendekatan berbasis konten semata.

4.2 *Deploying*

4.2.1 Uji Coba

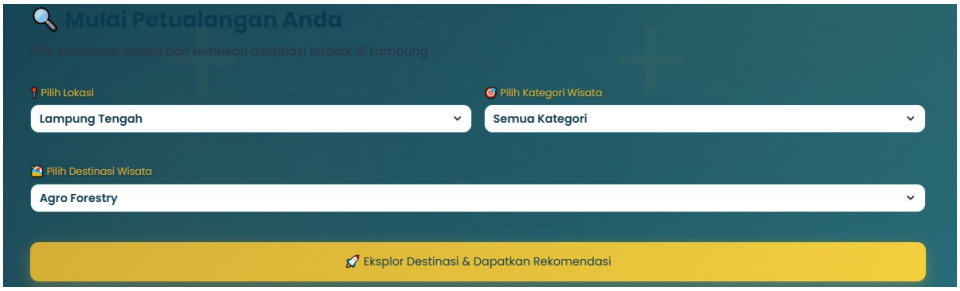
Uji coba sistem rekomendasi wisata ini dilakukan pada lingkungan *localhost* sebagai tahap awal validasi terhadap fungsionalitas dan performa antarmuka pengguna (*user interface*) serta logika sistem yang telah dikembangkan. Proses ini bertujuan untuk memastikan bahwa seluruh komponen sistem, mulai dari pemrosesan input hingga penyajian rekomendasi, berjalan sesuai dengan rancangan.

Pada tampilan awal aplikasi yang ditunjukkan pada Gambar 4.1, sistem memperkenalkan diri sebagai *Jelajahi Lampung* — sebuah sistem rekomendasi berbasis kecerdasan buatan yang dikembangkan dalam kerangka Proyek Sains Data. Informasi statistik yang disajikan, seperti jumlah destinasi (327), kategori (5), kabupaten/kota (15), dan rating rata-rata pengguna (4.4), memberikan gambaran mengenai cakupan data yang digunakan dalam sistem.



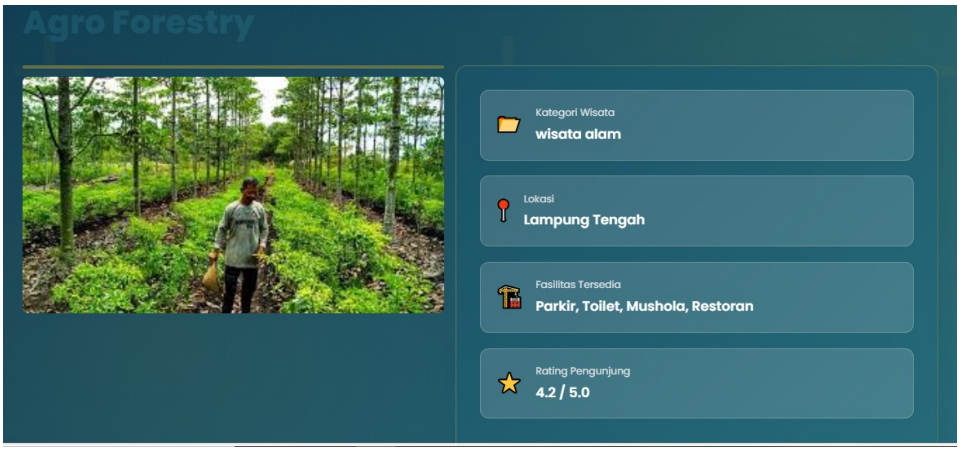
Gambar 4.1 Tampilan awal aplikasi *Jelajahi Lampung*

Selanjutnya, dilakukan uji coba eksplorasi destinasi wisata melalui antarmuka interaktif (Gambar 4.2). Pengguna diberikan opsi untuk menentukan parameter pencarian berupa lokasi administratif (kabupaten/kota), kategori wisata, dan nama destinasi spesifik. Dalam simulasi ini, pengguna memilih *Semua Kabupaten/Kota*, *Semua Kategori*, dan destinasi *Air Terjun Cengkaan*. Setelah konfigurasi selesai, tombol *Eksplor Destinasi & Dapatkan Rekomendasi* ditekan untuk memproses permintaan.



Gambar 4.2 Antarmuka eksplorasi fitur destinasi wisata

Hasil dari proses tersebut ditampilkan pada halaman detail destinasi (Gambar 4.3), dengan contoh destinasi *Agro Forestry*. Informasi yang ditampilkan meliputi kategori wisata (*wisata alam*), lokasi administratif (*Lampung Tengah*), fasilitas umum yang tersedia (seperti parkir, toilet, musola, dan restoran), serta rating pengguna sebesar 4.2 dari skala 5.



Gambar 4.3 Hasil detail destinasi wisata: Agro Forestry

Sebagai bagian dari implementasi AI, sistem juga menyajikan rekomendasi destinasi serupa berdasarkan analisis kemiripan fitur antar destinasi (Gambar 4.4). Tiga destinasi yang direkomendasikan, yaitu *Embung Tanjung Anom* (skor 0.827), *Lembah Milenium* (0.811), dan *Taman Umbul Kapuk* (0.809), ditampilkan lengkap dengan jarak geografis dari lokasi awal serta lokasi kabupaten terkait.

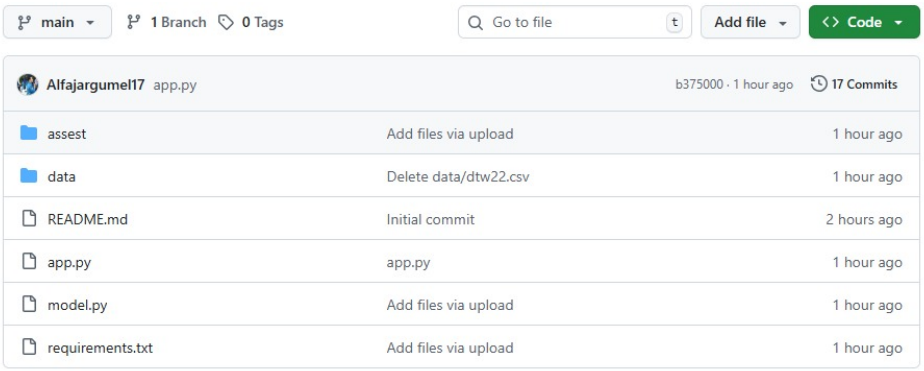
Gambar 4.4 Rekomendasi destinasi serupa yang dihasilkan sistem

Secara keseluruhan, hasil uji coba menunjukkan bahwa sistem mampu menangani alur interaksi pengguna dengan baik, menampilkan informasi destinasi secara informatif, dan menghasilkan rekomendasi yang relevan berdasarkan masukan pengguna.

4.2.2 Streamlit

Setelah sistem berhasil melalui tahap uji coba lokal dengan hasil yang sesuai ekspektasi, langkah selanjutnya adalah proses *deployment* agar aplikasi dapat diakses secara daring oleh pengguna umum. Untuk keperluan ini, dipilih platform *Streamlit*, yang memungkinkan integrasi cepat dengan repositori *GitHub* serta menyediakan antarmuka web yang ringan dan interaktif.

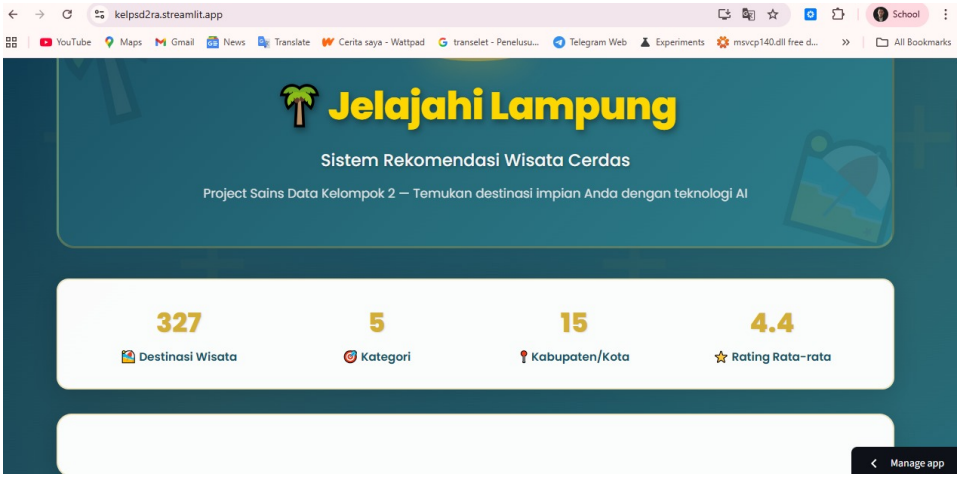
Langkah pertama yang dilakukan adalah mengunggah seluruh kode sumber, termasuk model, skrip Python, file pendukung, serta dokumentasi teknis ke dalam sebuah repositori GitHub. Repositori ini disusun secara terstruktur agar memudahkan proses integrasi dan pemeliharaan. Tampilan struktur repositori GitHub dapat dilihat pada Gambar 4.5.



Gambar 4.5 Struktur repositori proyek pada GitHub

Setelah repositori tersedia secara publik, proses *deployment* ke Streamlit dilakukan melalui autentikasi dan koneksi langsung ke GitHub. Platform Streamlit secara otomatis mengambil file utama (`app.py` atau `main.py`) dan melakukan proses build untuk menampilkan antarmuka pengguna melalui web. Hal ini memungkinkan pengguna akhir untuk mengakses aplikasi tanpa perlu mengunduh atau menjalankan kode secara lokal.

Hasil dari proses *deployment* ditunjukkan pada Gambar 4.6, yang memperlihatkan tampilan aplikasi seperti saat dijalankan secara lokal, namun kini dapat diakses melalui browser. Semua fitur yang telah diuji secara lokal tetap dapat digunakan dengan baik dalam versi daring ini.



Gambar 4.6 Tampilan aplikasi setelah berhasil dideploy di Streamlit

Adapun hasil dari proses *deployment* ini menghasilkan sebuah tautan publik yang dapat diakses oleh pengguna untuk mencoba aplikasi secara langsung. Tautan ini juga dicantumkan dalam dokumentasi proyek sebagai bagian dari diseminasi dan uji pemakaian pengguna secara real-time.

BAB V

Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil analisis dan pembahasan yang telah dilakukan, diperoleh beberapa kesimpulan sebagai berikut:

1. Sistem rekomendasi destinasi wisata di Provinsi Lampung berhasil dikembangkan menggunakan metode *Content-Based Filtering* (CBF) berbasis *cosine similarity*, dengan performa baseline yang sangat baik (intra mean 1,0, inter mean 0,0, dan separation 1,0), namun menghasilkan rekomendasi yang homogen dengan nilai diversity 0,0.
2. Integrasi *Location-Based Service* (LBS) menggunakan perhitungan jarak Haversine berhasil memperkaya sistem rekomendasi dengan konteks spasial, sehingga rekomendasi mempertimbangkan kemiripan konten dan kedekatan geografis.
3. Sistem rekomendasi *hybrid* (CBF dan LBS) menunjukkan performa yang baik dengan coverage 100% serta nilai evaluasi intra mean 0,782, inter mean 0,278, dan separation 0,504, meskipun nilai diversity masih relatif rendah (0,173).

5.2 Saran

Berdasarkan hasil penelitian dan keterbatasan yang ditemukan, beberapa saran untuk pengembangan lebih lanjut adalah sebagai berikut:

1. Mengembangkan mekanisme peningkatan diversity rekomendasi, seperti *re-ranking*, Maximum Marginal Relevance (MMR), atau penyesuaian bobot parameter *hybrid*.
2. Mengintegrasikan data *real-time* melalui API eksternal, seperti SISPARNAS dan Google Maps, agar informasi destinasi selalu diperbarui.
3. Menambahkan pendekatan *collaborative filtering* untuk menghasilkan rekomendasi yang lebih personal dan beragam.
4. Melakukan pengujian langsung kepada pengguna (*user testing*) untuk mengevaluasi tingkat kepuasan dan *usability* sistem.
5. Mengembangkan fitur visualisasi rute perjalanan antar destinasi secara interaktif.

6. Memperluas cakupan dan kualitas data destinasi wisata.
7. Mengoptimalkan performa sistem melalui peningkatan efisiensi kode dan penggunaan *caching*.
8. Mengimplementasikan sistem *feedback* pengguna untuk mendukung peningkatan akurasi rekomendasi secara berkelanjutan.

DAFTAR PUSTAKA

[1] A. News, “Lampung proyeksi jumlah kunjungan wisata 2025 lebih dari 20 juta orang”, *Antara News*, Feb. 2025. sumber: www.antaranews.com

[2] BCA Sekuritas. “Bps: Kunjungan wisman di indonesia capai 1,51 juta pada agustus 2025”. sumber: https://bcasekuritas.co.id/en/latest-news/news/bps-kunjungan-wisman-di-indonesia-capai-151-juta-pada-agustus-2025

[3] D. Pratiwi, A. Asrianda, dan L. Rosnita, “Penerapan metode content-based filtering dalam sistem rekomendasi objek wisata di aceh tamiang”, *Jurnal Ilmu Komputer dan Informatika*, vol. 4, no. 2, hlmn. 85–96, 2024.

[4] A. D. Aryanto, A. Primadewi, dan N. A. Prabowo, “Rekomendasi wisata kabupaten magelang menggunakan metode content-based filtering dan location-based service”, *JURNAL FASILKOM*, vol. 15, no. 1, hlmn. 172–178, 2025.

LAMPIRAN A

Kode Program

A.1 Data Preprocessing

```
import pandas as pd

# Load dataset
df = pd.read_csv("data/dtw_lpg_fixx.csv")

# Cek kategori unik
kategori_unik = df['kategori'].unique()

print("Kategori unik dalam dataset:")
for k in kategori_unik:
    print("-", k)

df.head(5)

print("Distribusi kategori:")
print(df['kategori'].value_counts())

fasilitas_set = set()
for row in df['Fasilitas']:
    items = [f.strip().title() for f in row.split(',')]
    fasilitas_set.update(items)

print("Fasilitas unik yang ditemukan:")
print(fasilitas_set)

df['kategori'] = df['kategori'].str.lower().str.strip()

# Mapping kategori
kategori_map = {
    'wisata alam': 'K1',
    'wisata buatan': 'K2',
    'wisata budaya': 'K3',
    'wisata religi': 'K4',
```

```
    'wisata lainnya': 'K5'
}
# Normalisasi Teks Kategori
df['kategori_kode'] = df['kategori'].map(kategori_map)

#Feature Engineering
# One-hot encoding kategori
for kode in ['K1', 'K2', 'K3', 'K4', 'K5']:
    df[kode] = (df['kategori_kode'] == kode).astype(int)


# One-hot encoding fasilitas
def clean_fasilitas(fasilitas_str):
    if pd.isna(fasilitas_str):
        return []
    return [f.strip().title() for f in fasilitas_str.split(',')]

df['fasilitas_list'] = df['Fasilitas'].apply(clean_fasilitas)

# Mapping fasilitas
fasilitas_map = {
    'Parkir': 'F1',
    'Restoran': 'F2',
    'Toilet': 'F3',
    'Mushola': 'F4'
}

for kode in ['F1', 'F2', 'F3', 'F4']:
    df[kode] = df['fasilitas_list'].apply(
        lambda x: 1 if fasilitas_map.get(x[0], None) == kode
            or kode in [fasilitas_map.get(item) for item in x] else 0
    )
# Tampilkan hasil
print("One-hot encoding kategori:")
print(df[['Nama tempat', 'K1','K2','K3','K4', 'K5']].head(10))
print("\nOne-hot encoding fasilitas:")
```

```
print(df[['Nama tempat', 'F1','F2','F3','F4']].head(10))

def clean_coordinates(df):
    df = df.copy()

    # Hapus duplikat
    df = df.drop_duplicates()

    # Konversi latitude dan longitude
    df['latitude'] = pd.to_numeric(df['latitude'], errors='
       coerce')
    df['longitude'] = pd.to_numeric(df['longitude'], errors=
       'coerce')

    # Hapus baris yang tidak punya koordinat
    df = df.dropna(subset=['latitude', 'longitude'])

    # Hapus koordinat outlier
    df = df[
        df['latitude'].between(-90, 90) &
        df['longitude'].between(-180, 180)
    ]

    # Reset index
    df = df.reset_index(drop=True)

    return df
```

A.2 Summary model

```
import numpy as np
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
# Fungsi deteksi kolom J dan F
def detect_k_f_cols(df, k_prefixes=('K',), f_prefixes=('F',)
   ):
    k_cols = [c for c in df.columns if any(c.startswith(pref
       ) for pref in k_prefixes) and c[1:].isdigit()]
    f_cols = [c for c in df.columns if any(c.startswith(pref
       ) for pref in f_prefixes) and c[1:].isdigit()]
```

```
    k_cols = sorted(k_cols, key=lambda s: int(''.join(filter
        (str.isdigit, s))))
    f_cols = sorted(f_cols, key=lambda s: int(''.join(filter
        (str.isdigit, s))))
    return k_cols, f_cols
# Ambil matriks fitur CBF
def get_cbf_matrices(df):
    k_cols, f_cols = detect_k_f_cols(df, k_prefixes=('K',),
        f_prefixes=('F',))
    if len(k_cols) == 0 and len(f_cols) == 0:
        raise ValueError("Tidak ditemukan kolom K* atau F*.
            Pastikan one-hot sudah dibuat.")
    mat_k = df[k_cols].fillna(0).astype(float).values if len
        (k_cols) > 0 else None
    mat_f = df[f_cols].fillna(0).astype(float).values if len
        (f_cols) > 0 else None
    return mat_k, mat_f, k_cols, f_cols


def cbf_similarity(df):
    k_cols, f_cols = detect_k_f_cols(df)

    mat_k = df[k_cols].values
    mat_f = df[f_cols].values

    sim_k = cosine_similarity(mat_k)
    sim_f = cosine_similarity(mat_f)

    return sim_k, sim_f

# Hitung similarity (cosine)
def compute_similarity_matrices(df):
    mat_k, mat_f, k_cols, f_cols = get_cbf_matrices(df)
    sim_k = cosine_similarity(mat_k) if mat_k is not None
        else None
    sim_f = cosine_similarity(mat_f) if mat_f is not None
        else None
    return sim_k, sim_f, k_cols, f_cols

# Fungsi rekomendasi CBF
```

```
def recommend_cbf(df, target, top_n=10, alpha_k=0.5, alpha_f
   =0.5, by_name=False):
    sim_k, sim_f, k_cols, f_cols =
       compute_similarity_matrices(df)

    if by_name:
        if 'Nama tempat' not in df.columns:
            raise ValueError("Kolom 'Nama tempat' tidak
               ditemukan di DataFrame.")
        matches = df.index[df['Nama tempat'].str.lower() ==
           str(target).lower()].tolist()
        if not matches:
            raise ValueError(f"Tidak ditemukan tempat dengan
                nama '{target}'. Pastikan penulisan benar.")
        target_idx = matches[0]
    else:
        target_idx = int(target)
        if target_idx not in df.index:
            if target_idx < 0 or target_idx >= len(df):
                raise IndexError("target index di luar
                   jangkauan.")
            target_idx = df.index[target_idx]

    n = len(df)

    score_k = np.zeros(n)
    score_f = np.zeros(n)

    if sim_k is not None:
        score_k = sim_k[target_idx]
    if sim_f is not None:
        score_f = sim_f[target_idx]

    def minmax(arr):
        lo, hi = arr.min(), arr.max()
        if hi - lo <= 1e-9:
            return np.zeros_like(arr)
        return (arr - lo) / (hi - lo)

    score_k_n = minmax(score_k) if sim_k is not None else np
```

```
        .zeros_like(score_k)
    score_f_n = minmax(score_f) if sim_f is not None else np
        .zeros_like(score_f)

    final_score = alpha_k * score_k_n + alpha_f * score_f_n

    res = df.copy()
    res['cbf_score'] = final_score

    res = res.drop(index=target_idx)

    res = res.sort_values(by='cbf_score', ascending=False)

    return res.head(top_n).reset_index(drop=False).rename(
        columns={'index':'original_index'})

# LBS
def haversine_vectorized(lat1, lon1, lat2, lon2):
    """
    Menghitung jarak menggunakan rumus Haversine secara
        vectorized,
    dapat menerima scalar atau array.
    """
    R = 6371.0  # radius bumi (km)

    lat1 = np.radians(lat1)
    lon1 = np.radians(lon1)
    lat2 = np.radians(lat2)
    lon2 = np.radians(lon2)

    dlat = lat2 - lat1
    dlon = lon2 - lon1

    a = np.sin(dlat/2)**2 + np.cos(lat1) * np.cos(lat2) * np
        .sin(dlon/2)**2
    return 2 * R * np.arcsin(np.sqrt(a))

def haversine_matrix(df):
    """
    Membuat matrix NxN berisi jarak antar item pada
```

```
        DataFrame.
    Tanpa loop, tanpa apply, full vectorized (sangat cepat).
    """
    R = 6371.0

    lat = np.radians(df['latitude'].values)
    lon = np.radians(df['longitude'].values)

    lat1 = lat[:, None]
    lon1 = lon[:, None]
    lat2 = lat[None, :]
    lon2 = lon[None, :]

    dlat = lat2 - lat1
    dlon = lon2 - lon1

    a = np.sin(dlat/2)**2 + np.cos(lat1)*np.cos(lat2)*np.sin
        (dlon/2)**2
    dist_matrix = 2 * R * np.arcsin(np.sqrt(a))

    return dist_matrix

def lbs_similarity(df):
    """
    similarity = 1 / (1 + distance)
    Nilai mendekati 1 jika jarak sangat dekat.
    """
    dist_matrix = haversine_matrix(df)
    sim_lbs = 1 / (1 + dist_matrix)

    # Similarity diri sendiri = 0
    np.fill_diagonal(sim_lbs, 0)

    return sim_lbs

def recommend_lbs(df, target, top_n=10, by_name=True):
    """
    Mengembalikan rekomendasi wisata berdasarkan kedekatan
        lokasi.
    """
```

```
    sim_lbs = lbs_similarity(df)

    # Tentukan index target
    if by_name:
        idx = df.index[df['Nama tempat'].str.lower() ==
           target.lower()]
        if len(idx) == 0:
            raise ValueError(f"Nama tempat '{target}' tidak
               ditemukan.")
        idx = idx[0]
    else:
        idx = target

    sims = sim_lbs[idx]

    # Ambil top-N similarity terbesar
    top_idx = np.argsort(-sims)[:top_n]

    hasil = df.iloc[top_idx].copy()
    hasil['lbs_score'] = sims[top_idx]

    return hasil[['Nama tempat', 'kabupaten kota', 'latitude
       ', 'longitude', 'lbs_score']]

def minmax_normalize(arr):
    a = np.array(arr, dtype=float)
    if a.ndim == 1:
        if np.all(np.isclose(a, a[0])):
            return np.zeros_like(a)
        lo = np.nanmin(a)
        hi = np.nanmax(a)
        if hi - lo <= 1e-12:
            return np.zeros_like(a)
        return (a - lo) / (hi - lo)
    else:
        # 2D matrix: normalize per entire matrix (flatten)
        flat = a.flatten()
        valid = ~np.isnan(flat)
        if valid.sum() == 0:
            return np.zeros_like(a)
```

```
        lo = np.nanmin(flat)
        hi = np.nanmax(flat)
        if np.isclose(hi, lo):
            return np.zeros_like(a)
        norm_flat = np.full_like(flat, np.nan, dtype=float)
        norm_flat[valid] = (flat[valid] - lo) / (hi - lo)
        return norm_flat.reshape(a.shape)

def build_hybrid_similarity(df, alpha1=0.5, alpha2=0.3, beta
   =0.2):

    # 1. compute components
    sim_k, sim_f, _, _ = compute_similarity_matrices(df)
    sim_lbs = lbs_similarity(df)

    # If any sim_j or sim_f is None (e.g., no J columns),
       create zeros
    n = len(df)
    if sim_k is None:
        sim_k = np.zeros((n, n), dtype=float)
    if sim_f is None:
        sim_f = np.zeros((n, n), dtype=float)

    # 2. normalize each component to 0..1 (handle NaN in
       sim_lbs)
    sim_k_n = minmax_normalize(sim_k)
    sim_f_n = minmax_normalize(sim_f)
    sim_lbs_n = minmax_normalize(sim_lbs)

    # 3. combine
    hybrid = (alpha1 * sim_k_n) + (alpha2 * sim_f_n) + (beta
        * sim_lbs_n)

    # ensure diagonal 0 (no self-recommendation)
    np.fill_diagonal(hybrid, 0.0)

    return hybrid, sim_k_n, sim_f_n, sim_lbs_n

# REKOMENDASI HYBRID
def recommend_hybrid(df, target, top_n=10, by_name=True,
```

```
alpha1=0.5, alpha2=0.3, beta=0.2):

 hybrid, sim_k_n, sim_f_n, sim_lbs_n =
    build_hybrid_similarity(df, alpha1=alpha1, alpha2=
    alpha2, beta=beta)

 # resolve target index
 if by_name:
     if 'Nama tempat' not in df.columns:
         raise ValueError("Kolom 'Nama tempat' tidak
            ditemukan.")
     idx_list = df.index[df['Nama tempat'].str.lower() ==
         str(target).lower()].tolist()
     if not idx_list:
         raise ValueError(f"Nama tempat '{target}' tidak
            ditemukan.")
     t = idx_list[0]
 else:
     t = int(target)
     if t < 0 or t >= len(df):
         raise IndexError("Index target di luar jangkauan
            .")

 scores = hybrid[t]
 # get top indices
 top_idx = np.argsort(-scores)[:top_n]
 res = df.iloc[top_idx].copy().reset_index(drop=True)

 # attach component scores
 res['score_hybrid'] = scores[top_idx]
 res['score_cat'] = sim_k_n[t, top_idx]
 res['score_fas'] = sim_f_n[t, top_idx]
 res['score_loc'] = sim_lbs_n[t, top_idx]

 # compute distance_km if lat/lon present (for display)
 if ('latitude' in df.columns) and ('longitude' in df.
    columns):
     # compute distances for target->top_idx using
        haversine_matrix (or vectorized haversine)
     # simple vectorized: haversine_vectorized(target_lat
```

```
            , target_lon, array_lats, array_lons)
        tlat = df.loc[t, 'latitude']
        tlon = df.loc[t, 'longitude']
        # if target has NaN coords, distances become NaN
        if not (np.isnan(tlat) or np.isnan(tlon)):
            # vectorized haversine for arrays
            lat_arr = res['latitude'].to_numpy(dtype=float)
            lon_arr = res['longitude'].to_numpy(dtype=float)
            # reuse haversine_vectorized logic:
            lat1 = np.radians(tlat)
            lon1 = np.radians(tlon)
            lat2 = np.radians(lat_arr)
            lon2 = np.radians(lon_arr)
            dlat = lat2 - lat1
            dlon = lon2 - lon1
            a = np.sin(dlat/2)**2 + np.cos(lat1)*np.cos(lat2
               )*np.sin(dlon/2)**2
            dist_km = 2 * 6371.0 * np.arcsin(np.sqrt(a))
            res['distance_km'] = dist_km
        else:
            res['distance_km'] = np.nan

    return res
```

A.3 Evaluasi

```
def evaluate_similarity(df, sim_k, sim_f):
    results = {}
    kategori_unique = df['kategori'].unique()
    intra_list = []
    inter_list = []

    for kat in kategori_unique:
        idx_same = df.index[df['kategori'] == kat].tolist()
        idx_other = df.index[df['kategori'] != kat].tolist()

        if len(idx_same) > 1:
            intra_scores = sim_k[np.ix_(idx_same, idx_same)]
            upper_triangle = intra_scores[np.triu_indices(
               len(idx_same), k=1)]
```

```
            intra_list.extend(upper_triangle)

        if len(idx_same) > 0 and len(idx_other) > 0:
            inter_scores = sim_k[np.ix_(idx_same, idx_other)
               ]
            inter_list.extend(inter_scores.flatten())

    results['intra_mean'] = np.mean(intra_list) if len(
       intra_list) else 0
    results['inter_mean'] = np.mean(inter_list) if len(
       inter_list) else 0
    results['separation'] = results['intra_mean'] - results[
       'inter_mean']

    return results

# COVERAGE
def evaluate_coverage(sim_matrix, threshold=0.1):
    n = len(sim_matrix)
    count = 0
    for i in range(n):
        if np.sum(sim_matrix[i] > threshold) > 1:
            count += 1
    return count / n

# DIVERSITY
def evaluate_diversity(sim_matrix, top_n=5):
    n = len(sim_matrix)
    div_list = []

    for i in range(n):
        sims = sim_matrix[i].copy()
        sims[i] = -1
        top_idx = np.argsort(-sims)[:top_n]

        sub_sim = sim_matrix[np.ix_(top_idx, top_idx)]
        upper = sub_sim[np.triu_indices(top_n, k=1)]
        if len(upper) > 0:
            diversity = 1 - np.mean(upper)
            div_list.append(diversity)
```

```
    return np.mean(div_list) if div_list else 0

# EVALUASI CBF
def evaluate_cbf(df):
    sim_k, sim_f = cbf_similarity(df)

    print("\nEvaluasi Similarity (Kategori)")
    sim_eval = evaluate_similarity(df, sim_k, sim_f)
    print(sim_eval)

    print("\nCoverage")
    coverage = evaluate_coverage(sim_k)
    print(f"Coverage: {coverage:.3f}")

    print("\nDiversity")
    diversity = evaluate_diversity(sim_k, top_n=5)
    print(f"Diversity: {diversity:.3f}")

    return {
        "similarity_eval": sim_eval,
        "coverage": coverage,
        "diversity": diversity
    }

result = evaluate_cbf(df)
print(result)

def evaluate_hybrid(df, alpha1=0.5, alpha2=0.3, beta=0.2):
    hybrid, _, _, _ = build_hybrid_similarity(df, alpha1=
       alpha1, alpha2=alpha2, beta=beta)

    # similarity by categories
    cats = df['kategori'].unique()
    intra = []
    inter = []
    for cat in cats:
        idx_same = df.index[df['kategori'] == cat].tolist()
        idx_other = df.index[df['kategori'] != cat].tolist()
        if len(idx_same) > 1:
```

```
            mat = hybrid[np.ix_(idx_same, idx_same)]
            intra.extend(mat[np.triu_indices(len(idx_same),
               k=1)].tolist())
        if len(idx_same) > 0 and len(idx_other) > 0:
            mat2 = hybrid[np.ix_(idx_same, idx_other)]
            inter.extend(mat2.flatten().tolist())
    intra_mean = np.mean(intra) if len(intra)>0 else 0.0
    inter_mean = np.mean(inter) if len(inter)>0 else 0.0

    # coverage: at least one recommendation above small
       threshold
    n = len(hybrid)
    thresh = 0.01
    count = 0
    for i in range(n):
        if np.sum(hybrid[i] > thresh) >= 1:
            count += 1
    coverage = count / n

    # diversity (top-5)
    divs = []
    top_k = 5
    for i in range(n):
        sims = hybrid[i].copy()
        sims[i] = -1
        top_idx = np.argsort(-sims)[:top_k]
        sub = hybrid[np.ix_(top_idx, top_idx)]
        upper = sub[np.triu_indices(len(top_idx), k=1)]
        if len(upper) > 0:
            divs.append(1 - np.mean(upper))
    diversity = np.mean(divs) if len(divs)>0 else 0.0

    return {
        "intra_mean": intra_mean,
        "inter_mean": inter_mean,
        "separation": intra_mean - inter_mean,
        "coverage": coverage,
        "diversity": diversity
    }
```

LAMPIRAN B

Hasi Lengkap Eksperimen

```
# Penggunaan CBF
top5 = recommend_cbf(df, target='Air Terjun Cengkaan', top_n
   =5, alpha_k=0.6, alpha_f=0.4, by_name=True)

print(top5[['Nama tempat', 'kabupaten kota', 'cbf_score']])

# Penggunaan LBS
rekom = recommend_lbs(df, "Curug Gimo", top_n=5)
print(rekom)

# Penggunaan Hybrid Recommendation CBF + LBS

df_fixed = fix_unhashable(df)
df_clean = clean_coordinates(df_fixed)

hybrid_sim, _, _, _ = build_hybrid_similarity(df_clean,
   alpha1=0.5, alpha2=0.3, beta=0.2)
rec = recommend_hybrid(df_clean, target="Air Terjun Cengkaan
   ", top_n=10, by_name=True, alpha1=0.5, alpha2=0.3, beta
   =0.2)
print(rec[['Nama tempat','kategori','score_hybrid','
   score_cat','score_fas','score_loc','distance_km']])
eval_result = evaluate_hybrid(df_clean, alpha1=0.5, alpha2
   =0.3, beta=0.2)
print(eval_result)
```

LAMPIRAN C

Dokumentasi

Seluruh dokumentasi proyek, dapat diakses melalui repositori GitHub berikut:

https://github.com/sains-data/team-2-projek-sains-data-_RA