ITERA

DOKUMEN SPESIFIKASI PROYEK *BIG DATA*

Implementasi Ekosistem Hadoop untuk Analisis Big Data Lalu Lintas Kota Medan: Prediksi Kemacetan Jalan Menggunakan Apache Spark dan Random Forest Berbasis Data GPS

Kelompok 16

Dwi Ratna Anggraeni	122450008
Febiya Jomy Pratiwi	122450074
Residen Nusantara R M	122450080
Fayyaza Aqila S A	122450131

PROGRAM STUDI SAINS DATA

FAKULTAS SAINS

INSTITUT TEKNOLOGI SUMATERA

Mei 2025

DAFTAR ISI

1. Pendahuluan

1.1 Latar Belakang

Kemacetan lalu lintas merupakan salah satu tantangan utama yang dihadapi oleh kota-kota besar, termasuk Kota Medan. Pertumbuhan jumlah kendaraan pribadi yang tidak diiringi oleh peningkatan kapasitas dan kualitas infrastruktur jalan, serta belum optimalnya sistem manajemen lalu lintas, menyebabkan penumpukan kendaraan pada waktu-waktu tertentu. Dampaknya tidak hanya dirasakan dalam bentuk keterlambatan mobilitas, tetapi juga mempengaruhi efisiensi ekonomi, meningkatnya konsumsi bahan bakar, polusi udara, hingga menurunnya kualitas hidup masyarakat.

Dalam konteks perkembangan teknologi digital, data GPS dari kendaraan kini menjadi sumber data spasial-temporal yang sangat kaya dan berpotensi besar untuk dianalisis. Data ini dapat digunakan untuk memantau dan memetakan kondisi lalu lintas secara real-time serta memberikan informasi historis mengenai pola kemacetan di berbagai ruas jalan. Namun, besarnya volume data GPS yang dihasilkan secara kontinu menjadikan pemrosesan dan analisisnya tidak dapat ditangani secara efisien oleh sistem konvensional.

Oleh karena itu, pemanfaatan pendekatan Big Data menjadi sangat relevan dalam permasalahan ini. Teknologi ekosistem Hadoop memungkinkan penyimpanan dan pemrosesan data besar secara terdistribusi, sementara Apache Spark memberikan kemampuan analitik cepat, termasuk dalam menerapkan algoritma pembelajaran mesin. Dengan mengintegrasikan algoritma Random Forest, sistem ini dapat membangun model prediktif yang mampu mengklasifikasikan tingkat kemacetan berdasarkan waktu, lokasi GPS, dan kecepatan kendaraan. Pendekatan ini memberikan solusi berbasis data untuk mendukung manajemen lalu lintas yang lebih cerdas dan adaptif di Kota Medan.

1.2 Tujuan Dokumen

Dokumen ini bertujuan untuk menjelaskan perancangan dan implementasi sistem prediksi kemacetan lalu lintas di Kota Medan dengan memanfaatkan data GPS kendaraan, ekosistem Hadoop, dan algoritma Random Forest. Fokus dari sistem ini meliputi tiga aspek utama, yaitu perancangan arsitektur Big Data berbasis Hadoop dan Apache Spark, pembangunan pipeline data untuk ingestion dan transformasi data spasial-temporal, serta penerapan model Random Forest untuk mengklasifikasikan tingkat kemacetan berdasarkan waktu, lokasi, dan kecepatan kendaraan. Dokumen ini disusun sebagai panduan teknis dalam pengembangan solusi prediktif lalu lintas berbasis Big Data dan Machine Learning.

1.3 Lingkup Sistem

Sistem ini dirancang untuk memproses dan menganalisis data GPS kendaraan dalam skala besar guna memprediksi tingkat kemacetan di ruas-ruas jalan Kota Medan. Lingkupnya mencakup proses pengambilan data GPS dari sumber eksternal, penyimpanan data mentah dalam HDFS (bronze layer), pembersihan dan transformasi data menggunakan Apache Spark (silver layer), serta agregasi dan analisis data pada gold layer. Sistem juga mencakup pelatihan model Random Forest untuk klasifikasi kemacetan, integrasi hasil ke dalam Hive untuk query analitik, serta penyajian visualisasi prediksi melalui dashboard interaktif menggunakan Apache Superset. Seluruh komponen berjalan dalam ekosistem Hadoop yang dikembangkan secara modular dan terdistribusi, serta dapat dijalankan secara lokal menggunakan Docker sebagai simulasi cluster produksi.

1.4 Signifikansi Dokumen

Dokumen ini memiliki peran penting sebagai panduan teknis dalam pengembangan sistem prediksi kemacetan berbasis Big Data dan Machine Learning. Secara akademik, dokumen ini memberikan wawasan praktis bagi mahasiswa dalam menerapkan ekosistem Hadoop dan algoritma Random Forest pada data spasial-temporal.

Dari sisi fungsional, sistem yang dibangun dapat dimanfaatkan oleh instansi seperti Dinas Perhubungan Kota Medan untuk memantau kondisi lalu lintas dan mengambil keputusan berbasis data. Selain itu, sistem ini juga mendukung pengembangan kota cerdas (smart city) dengan menyediakan informasi kemacetan secara prediktif dan visual, yang berguna bagi masyarakat dan perencana kebijakan transportasi.

2. Deskripsi Umum

2.1 Perspektif Sistem

Sistem ini merupakan platform prediksi kemacetan lalu lintas Kota Medan berbasis Big Data yang dirancang menggunakan ekosistem Hadoop. Dengan pendekatan *distributed computing*, sistem mampu memproses data GPS kendaraan dalam jumlah besar secara batch untuk menghasilkan analisis dan prediksi kemacetan secara efisien. Komponen utama yang digunakan meliputi HDFS untuk penyimpanan data, Apache Spark untuk pemrosesan dan pelatihan model Random Forest, Hive untuk query analitik, serta Superset untuk visualisasi prediktif dalam bentuk peta interaktif. Orkestrasi pipeline dilakukan menggunakan Apache Airflow. Sistem dirancang modular dan skalabel, serta dikembangkan dalam lingkungan lokal berbasis Docker yang dapat disesuaikan untuk skala produksi atau cloud. Arsitektur ini mendukung integrasi fitur tambahan di masa depan, seperti data cuaca atau sensor lalu lintas, guna memperkuat fungsi sistem dalam mendukung smart city.

2.2 Fungsi Sistem Utama

Pada proyek ini terdapat beberapa fungsi utama, yaitu:

- **Hadoop Distributed File System (HDFS)**
  Berfungsi sebagai sistem penyimpanan data terdistribusi dalam skala besar dengan mekanisme replikasi blok data antar node, serta mendukung eksekusi job secara paralel pada beberapa node pekerja.

- **Apache Spark**
  Digunakan untuk pemrosesan batch data, meliputi proses ETL (Extract, Transform, Load) serta analisis data seperti klasifikasi menggunakan algoritma Random Forest, regresi, dan clustering.

- **Apache Hive**
  Menyediakan layanan query SQL untuk mengakses dan menganalisis data yang tersimpan dalam HDFS melalui antarmuka HiveQL.

- **Apache HBase**
  Berperan sebagai penyimpanan data semi terstruktur hasil proses transformasi, memungkinkan akses data dengan kecepatan tinggi dan fleksibilitas dalam pengolahan data real-time.

- **Apache Airflow**
  Mengatur orkestrasi dan penjadwalan pipeline data dari proses ingestion hingga analitik dan visualisasi.

- **Apache Superset**
  Menyediakan visualisasi data interaktif berupa peta dan grafik untuk memudahkan pemantauan dan pengambilan keputusan terkait kemacetan lalu lintas.

- **Apache Ambari**
  Memantau kesehatan dan performa cluster Hadoop serta komponennya secara real-time.

2.3 Karakteristik Pengguna

Sistem prediksi kemacetan berbasis Big Data dan Machine Learning ini dirancang untuk melayani berbagai pemangku kepentingan dengan kebutuhan dan tingkat pemahaman teknologi yang berbeda. Analis data dan ilmuwan transportasi menggunakan data GPS yang terstruktur untuk mengevaluasi pola kemacetan, membangun laporan, serta melakukan analisis mendalam melalui query Hive dan visualisasi data. Mereka membutuhkan akses terhadap data historis, hasil model prediksi, dan kemampuan ekspor hasil analisis guna mendukung riset dan pengembangan. Dinas Perhubungan dan pengambil kebijakan kota memanfaatkan visualisasi kemacetan berupa peta dan grafik interaktif untuk mengambil keputusan berbasis data secara cepat. Informasi ini sangat berguna dalam perencanaan rekayasa lalu lintas, pengaturan jam operasional kendaraan berat, serta optimalisasi rute transportasi umum.

Sementara itu, administrator sistem dan teknologi informasi bertanggung jawab atas pengelolaan infrastruktur cluster Hadoop, termasuk instalasi, konfigurasi, pemantauan layanan, pemeliharaan pipeline data, dan pengelolaan akses pengguna. Mereka juga menangani logging serta backup data GPS untuk memastikan kestabilan dan keandalan sistem. Selain itu, mahasiswa dan pengajar bidang data science dan transportasi menggunakan sistem ini sebagai media pembelajaran praktis untuk memahami penerapan teknologi Big Data dan Machine Learning pada data spasial-temporal. Sistem ini sangat cocok digunakan dalam praktikum mata kuliah seperti Big Data, Analitik Spasial, dan Sistem Informasi Geografis.

3. Spesifikasi Proyek

3.1 Ringkasan

Proyek ini merupakan bagian dari implementasi pembelajaran pada mata kuliah Big Data di Program Studi Sains Data, Institut Teknologi Sumatera, dengan fokus pada permasalahan kemacetan lalu lintas di Kota Medan. Sistem yang dikembangkan memanfaatkan ekosistem Hadoop dan teknologi pendukung seperti Apache Spark, Hive, Airflow, dan Superset untuk mengelola dan menganalisis data GPS dari ribuan kendaraan secara kontinu. Data diproses melalui arsitektur Data Lake berlapis (Bronze, Silver, Gold) yang memungkinkan pembersihan, transformasi, dan agregasi data secara efisien. Proyek ini membangun pipeline otomatis yang mengumpulkan data GPS, mengolahnya menjadi format optimal, dan melatih model Machine Learning, khususnya Random Forest, untuk memprediksi tingkat kemacetan berdasarkan waktu dan lokasi. Selain menghasilkan insight

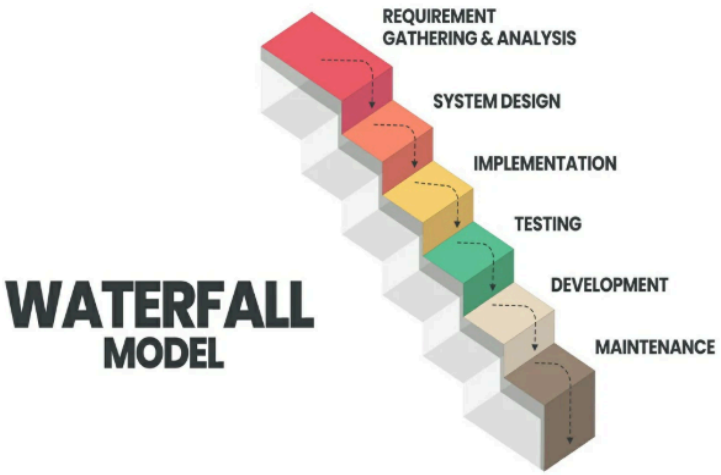
prediktif, sistem juga menyediakan dashboard visualisasi interaktif berbasis peta yang dapat digunakan oleh Dinas Perhubungan, operator transportasi, maupun masyarakat umum. Proyek ini sekaligus menjadi portofolio praktikal bagi mahasiswa dalam penerapan teknologi Big Data modern pada data spasial-temporal nyata.

3.2 Metode Proyek

Proyek ini menggunakan model pengembangan sistem Waterfall, yang dilakukan secara berurutan mulai dari analisis kebutuhan hingga deployment dan pemeliharaan. Tahap pertama adalah analisis kebutuhan, di mana dilakukan identifikasi jenis data GPS yang dibutuhkan serta spesifikasi teknis sistem untuk pengumpulan, penyimpanan, dan pemrosesan data berskala besar. Selanjutnya, dilakukan perancangan arsitektur Big Data berbasis Hadoop yang mencakup pipeline ingestion data, pemrosesan batch menggunakan Apache Spark, serta integrasi dengan Hive untuk analitik dan Superset untuk visualisasi.

Setelah itu, tahap desain sistem mencakup pembuatan struktur data, skema tabel Hive, penjadwalan workflow menggunakan Apache Airflow, dan pengaturan lingkungan pengembangan berbasis Docker. Pada tahap implementasi, pipeline data dibangun mulai dari proses ingestion, ETL, pelatihan model Random Forest, hingga deployment dashboard visualisasi. Sistem kemudian diuji melalui pengujian unit, integrasi, serta evaluasi performa model prediksi dengan menggunakan metrik seperti akurasi dan confusion matrix.

Terakhir, sistem dideploy pada lingkungan lokal menggunakan Docker dengan pipeline yang berjalan otomatis. Kinerja dan stabilitas sistem dipantau secara kontinu menggunakan Apache Ambari. Pendekatan ini memastikan setiap tahap berjalan sistematis sehingga menghasilkan sistem prediksi kemacetan yang handal dan dapat dikembangkan lebih lanjut.



Gambar 1. Metode Waterfall

3.3 Studi Kasus

Studi kasus proyek ini berfokus pada pengelolaan dan prediksi kemacetan lalu lintas di Kota Medan menggunakan data GPS kendaraan. Kota Medan, sebagai salah satu kota metropolitan di Indonesia, menghadapi peningkatan signifikan jumlah kendaraan yang berdampak pada kemacetan di beberapa ruas jalan utama seperti Jalan Gatot Subroto, Jalan Sisingamangaraja, dan Jalan Jenderal Sudirman. Sistem ini mengumpulkan data spasial dan temporal dari GPS kendaraan secara kontinu untuk memantau kondisi lalu lintas. Data yang diperoleh diolah melalui pipeline Big Data untuk mengidentifikasi pola kemacetan berdasarkan waktu dan lokasi. Dengan menggunakan model Random Forest, sistem mampu memprediksi tingkat kemacetan dalam interval waktu tertentu, sehingga memberikan informasi yang berguna bagi Dinas Perhubungan Kota Medan untuk pengambilan keputusan dan perencanaan lalu lintas yang lebih efektif. Selain itu, hasil prediksi disajikan dalam bentuk dashboard interaktif yang dapat diakses oleh masyarakat umum dan operator transportasi untuk memilih rute terbaik dan menghindari kemacetan.

4. Metode Umum

4.1 Analisis Kebutuhan (Requirement Analysis)

Tujuan Analisis kebutuhan dilakukan untuk memahami permasalahan utama yang ingin diatasi melalui implementasi ekosistem Hadoop dalam sistem prediksi kemacetan lalu lintas Kota Medan. Permasalahan utama adalah ketiadaan sistem prediksi kemacetan secara real-time yang menghambat respons cepat dan pengambilan keputusan efektif dalam pengelolaan lalu lintas. Dengan memanfaatkan teknologi Big Data seperti Apache Spark dan algoritma Random Forest, sistem ini diharapkan mampu menghasilkan prediksi kemacetan yang akurat serta memberikan rekomendasi rute alternatif yang lebih efisien.

Stakeholder utama yang terlibat antara lain Dinas Perhubungan Kota Medan sebagai pengambil kebijakan dan pengelola lalu lintas, operator transportasi daring seperti ojek online dan taksi digital yang membutuhkan informasi lalu lintas real-time, peneliti serta akademisi di bidang transportasi dan data science untuk pengembangan model prediktif lebih lanjut, serta masyarakat pengguna jalan sebagai end-user yang memperoleh informasi prediksi kemacetan dan rekomendasi rute.

Output yang diharapkan dari sistem ini meliputi peta interaktif prediksi kemacetan berdasarkan lokasi dan waktu, visualisasi grafik tren lalu lintas harian dan mingguan, serta rekomendasi rute alternatif yang dihitung berdasarkan tingkat kemacetan dan estimasi waktu tempuh dari model.

Untuk membangun model prediksi yang akurat, data yang diperlukan mencakup data GPS kendaraan berupa koordinat latitude dan longitude, informasi waktu dan tanggal pergerakan kendaraan untuk mengetahui pola kemacetan berdasarkan waktu, kecepatan kendaraan sebagai indikator kepadatan lalu lintas, identifikasi ruas jalan atau ID segmen jalan untuk pemetaan spasial, serta volume kendaraan per segmen waktu guna menghitung densitas lalu lintas.

4.1.1 Tabel Kebutuhan Fungsional

Tabel 1. Kebutuhan Fungsional

No	Kebutuhan	Prioritas
1	Mengumpulkan data GPS kendaraan dari sumber terpercaya (provider GPS/API lokal)	Tinggi
2	Menyimpan data GPS mentah dalam sistem file terdistribusi (HDFS)	Tinggi
3	Membersihkan dan memvalidasi data GPS (koordinat, waktu, nilai null, kecepatan tidak wajar)	Tinggi
4	Mengubah format data ke Parquet/ORC untuk efisiensi komputasi dan storage	Sedang
5	Mengelompokkan data berdasarkan waktu dan lokasi (segmen jalan, jam)	Tinggi
6	Menghitung kecepatan rata-rata dan tingkat kemacetan per ruas dan waktu tertentu	Tinggi
7	Menyediakan query analitik melalui Hive untuk analisis lalu lintas	Tinggi
8	Menyediakan visualisasi interaktif (peta dan grafik tren kemacetan)	Sedang
9	Menyusun pipeline batch dan/atau near real-time processing secara periodik	Sedang
10	Monitoring dan pengelolaan sistem melalui Ambari atau tool setara	Rendah
11	Logging dan backup proses ingestion, transformasi, dan prediksi	Sedang

4.1.2 Tabel Kebutuhan Non-Fungsional

Tabel 2. Kebutuhan Non-Fungsional

No	Kebutuhan	Prioritas
1	Sistem dapat diskalakan secara horizontal untuk menampung pertambahan data GPS (Skalabilitas)	Tinggi
2	Sistem dirancang dengan ketersediaan tinggi agar dapat berjalan 24/7 (Ketersediaan)	Tinggi
3	Pengaturan akses data berdasarkan role (admin, analis, publik) (Keamanan Data)	Sedang

4	Pipeline prediksi harus mampu recovery otomatis saat terjadi error/crash (Reliabilitas)	Tinggi
5	Sistem berjalan di lingkungan Docker agar mudah dipindahkan dan direplikasi (Portabilitas)	Sedang
6	Penggunaan format Parquet/ORC untuk efisiensi penyimpanan dan akselerasi query	Sedang
7	Dokumentasi menyeluruh mencakup arsitektur, konfigurasi, dan pipeline data (Dokumentasi)	Tinggi

4.2 Perancangan Sistem: Arsitektur dan Desain (System Design)

4.2.1 Arsitektur Data

Sistem ini menggunakan pendekatan *Medallion Architecture* dengan skema *batch processing* untuk mengelola data GPS dari kendaraan yang digunakan dalam analisis kemacetan. Setiap lapisan dalam arsitektur memiliki fungsi spesifik yang mendukung *pipeline* data dari mentah hingga siap analitik.

Tabel 3. Arsitektur Medallion

Lapisan	Deskripsi	Komponen Utama (Tools)	Format Data	Tujuan
Bronze	Menyimpan data mentah GPS dari API atau file CSV/JSON tanpa transformasi.	HDFS, curl, bash	CSV / JSON / Raw	Arsip permanen data GPS mentah dari sumber eksternal
Silver	Menyimpan data hasil pembersihan dan transformasi, validasi timestamp, parsing koordinat, dan deteksi outlier.	Apache Spark (ETL), Apache Hive (DDL), HDFS	Parquet / Avro	Data GPS terstruktur, bersih, dan siap digunakan untuk analitik
Gold	Menyimpan hasil agregasi analitik, seperti kecepatan rata-rata, volume kendaraan, dan klasifikasi kemacetan.	Apache Hive, Apache Spark (Aggregation), Apache Superset	Parquet / ORC	Dataset akhir untuk query cepat dan visualisasi prediksi kemacetan

4.2.2 Desain Infrastruktur

Sistem dikembangkan dalam lingkungan cluster lokal berbasis Docker, yang terdiri dari sejumlah komponen utama dari Hadoop ecosystem. Berikut adalah spesifikasi infrastrukturnya:

Tabel 4. Arsitektur Cluster Lokal (Hadoop Cluster)

Komponen	Jumlah Node	Peran dan Deskripsi
Hadoop Namenode	1	Master node untuk HDFS, menyimpan metadata file
Hadoop Datanode	2	Penyimpanan blok data GPS, direplikasi untuk toleransi kesalahan
ResourceManager (YARN)	1	Penjadwalan job Spark dan Hive
NodeManager	2	Menjalankan task Spark pada setiap worker node
Apache HiveServer2	1	Engine untuk eksekusi query HiveQL dan koneksi BI tools
Apache Spark Master	1	Koordinator job Spark batch
Apache Spark Worker	2	Menjalankan job transformasi dan analitik Spark
Hive Metastore	1	Metadata store untuk skema tabel Hive
Apache Ambari Server	1	Monitoring layanan Hadoop secara visual
Apache Superset	1	Visualisasi prediksi kemacetan pada layer Gold

Tabel 5. Daftar Teknologi Apache Projects yang Digunakan

No	Teknologi	Kategori	Fungsi Utama
1	Hadoop HDFS	Storage	Menyimpan data GPS mentah dan hasil transformasi secara terdistribusi
2	Hadoop YARN	Resource Management	Mengatur resource CPU/RAM dan menjadwalkan job Spark
3	Apache Hive	Query Engine/Metadata	Query SQL terhadap data agregat GPS

4	Apache Spark	ETL / Analytics Engine	Membersihkan data GPS, menghitung agregat, dan membangun model prediktif
5	Apache Ambari	Cluster Management	Monitoring performa cluster Hadoop dan komponennya
6	Hive Metastore	Metadata Store	Menyimpan skema dan metadata tabel analitik
7	Apache Superset	BI / Visualisasi	Menyajikan hasil prediksi kemacetan dalam bentuk grafik dan peta interaktif
8	Apache Airflow	Workflow Orchestration	Menjadwalkan pipeline: ingest → transform → aggregate → export

4.2.3 Orkestrasi Directed Acyclic Graphs (DAG) di Airflow

```
dag_traffic_pipeline:
├── task_1: fetch_gps_from_api
├── task_2: load_to_hdfs_bronze
├── task_3: spark_transform_to_silver
├── task_4: spark_aggregate_to_gold
├── task_5: hive_refresh_traffic_tables
└── task_6: export_prediction_to_dashboard
```

Tabel 6. Orkestrasi Alur Kegiatan Sistem

Urutan	Aktivitas	Layer	Tools
1	Ambil data GPS dari API dan unggah ke HDFS	Bronze	Bash, curl, HDFS CLI
2	Simpan data GPS mentah tanpa perubahan	Bronze	HDFS
3	Validasi format dan pembersihan data GPS (null, duplikat)	Silver	Apache Spark, Hive Metastore
4	Ubah format ke Parquet dan simpan sebagai data bersih	Silver	Spark + HDFS
5	Hitung metrik lalu lintas: kecepatan, volume, kemacetan	Gold	Spark SQL / Hive
6	Simpan agregat prediksi ke tabel Hive	Gold	HDFS, Hive
7	Visualisasi peta dan grafik kemacetan	Gold	Superset / Jupyter
8	Monitoring dan manajemen sistem	Semua	Apache Ambari

4.3 Implementasi

Implementasi sistem dilakukan untuk membangun arsitektur dan perancangan sistem menjadi sistem nyata (*real deployment*) dalam lingkungan lokal menggunakan Docker Desktop VM berbasis Linux. Sistem ini memanfaatkan tools utama dari *Hadoop ecosystem* dan Apache Foundation untuk menangani big data GPS lalu lintas Kota Medan dalam rangka memprediksi kemacetan jalan.

4.3.1 Lingkungan Implementasi

Tabel 7. Spesifikasi Lingkungan

Komponen	Spesifikasi
OS	Ubuntu Server 22.04 (via Docker VM), host OS: Windows 11
Cluster	Hadoop Cluster Pseudo-distributed dengan 2 Node
Orkestrasi	Shell Script + Crontab (opsional: Apache Nifi/Airflow)
Core Tools	Hadoop, HDFS, Hive, Spark, Ambari, Superset, Airflow
Penyimpanan	HDFS sebagai distributed storage
Format Data	CSV (bronze), Parquet (silver & gold)

4.3.2 Instalasi dan Setup Cluster

a. Persiapan Cluster Lokal

1. Install Docker Desktop dan aktifkan WSL2 di Windows 11.
2. Jalankan container berbasis image Hadoop.
3. Gunakan docker-compose untuk spin-up:
    - 2 NameNode
    - 2 DataNode
    - 1 Spark Master
    - 1 Spark Worker
    - 1 Hive Metastore + HiveServer2
    - Ambari
    - Superset
    - Airflow

b. Struktur docker-compose.yml

Buat file docker-compose.yml untuk mendefinisikan layanan, jaringan, dan volume yang dibutuhkan setiap container.

4.3.3 Struktur Folder HDFS

Tabel 8. Struktur Folder dan Format

Layer	Path HDFS	Format
Bronze	/data/bronze/	CSV
Silver	/data/silver/	Parquet
Gold	/data/gold/	Parquet
Temp	/data/tmp/	-

4.3.4 Implementasi Pipeline Batch

1. Bronze Layer (Ingestion)

```
curl -o gps_raw.csv https://data.medan.go.id/api/gps
hdfs dfs -put -f gps_raw.csv /data/bronze/gps_$(date +%F).csv
```

Automasi via crontab untuk ingest data GPS secara berkala.

2. Silver Layer (Cleansing & Normalization)

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("CleanGPS").getOrCreate()
df = spark.read.csv("/data/bronze/*.csv", header=True)
df_clean = df.dropDuplicates().na.fill("Unknown")
df_clean.write.parquet("/data/silver/gps_clean.parquet",
mode="overwrite")
```

3. Gold Layer (Aggregation & Prediction Feature Preparation)

```
df = spark.read.parquet("/data/silver/gps_clean.parquet")
agg = df.groupBy("jalan", "jam").agg({"kecepatan": "avg"})
agg.write.parquet("/data/gold/gps_summary.parquet",
mode="overwrite")
```

4.3.5 Implementasi Query dan Visualisasi

Hive Table (DDL)

```
CREATE EXTERNAL TABLE gps_summary (
  jalan STRING,
  jam STRING,
  rata_rata_kecepatan DOUBLE
)
STORED AS PARQUET
LOCATION '/data/gold/gps_summary.parquet';
```

4.3.6 Pengembangan ETL

- Upload data GPS mentah → Bronze (HDFS)
- ETL Spark job → Silver (cleaning & transformasi)
- Agregasi & fitur untuk model ML → Gold (ringkasan kecepatan rata-rata)
- Hive table dari gold layer

4.3.7 Dashboard

- Import dataset `gps_summary` ke Apache Superset.
- Visualisasi peta panas kemacetan, tren waktu, dan analisis per ruas jalan.

4.3.8 Monitoring dan Logging

- Spark UI: `http://localhost:8080`
- Ambari UI: Monitoring service dan resource
- Log file otomatis di `/logs/` jika dikonfigurasi

4.3.9 Output Implementasi

Tabel 9. Output dan Lokasi Implementasi

Output	Lokasi
Data GPS mentah	/data/bronze/
Data bersih GPS	/data/silver/
Ringkasan kecepatan rata-rata jalan	/data/gold/
Tabel Hive (gps_summary)	Hive Metastore
Visualisasi lalu lintas	Apache Superset / Jupyter Dashboard

4.4 Penerapan (Deployment)

Penerapan sistem dilakukan pada lingkungan lokal berbasis Docker VM yang di simulasikan sebagai cluster Hadoop pseudo-distributed untuk mendukung kebutuhan batch processing berbasis data lake. Tahapan deployment mencakup penyusunan lingkungan kontainer, ingest data mentah, transformasi data menggunakan Spark, serta integrasi dengan Hive dan visualisasi menggunakan Superset.

4.4.1 Strategi Deployment

Tabel 10. Tahap dan Deskripsi Strategi

Tahapan	Deskripsi
1. Setup Environment	Instalasi Docker Desktop dan Ubuntu Server (via WSL2) serta setup docker-compose cluster.

2. Build & Configure	Jalankan dan konfigurasikan layanan Hadoop, Spark, Hive, HDFS, Ambari, Superset.
3. Data Ingestion	Gunakan curl dan bash script dalam cron job untuk mengambil data GPS mentah ke layer **bronze**.
4. Spark Transformation	Deploy batch job Spark untuk cleansing (**silver layer**) dan agregasi OLAP (**gold layer**).
5. Hive Integration	Definisikan tabel eksternal Hive dari hasil transformasi di **gold layer**.
6. Visualisasi	Deploy Apache Superset dan hubungkan ke Hive metastore untuk eksplorasi dan visualisasi data.

4.4.2 Teknologi Deployment

Tabel 11. Alat dan Peran Teknologi

Tools	Peran
Docker	Virtualisasi container untuk seluruh layanan di Hadoop ecosystem.
Docker Compose	Orkestrasi dan manajemen layanan multi-container (Hadoop, Spark, Hive, dll).
Bash + Crontab	Automasi proses pengambilan data secara berkala ke dalam HDFS.
Spark Submit	Menjalankan job batch ETL (cleansing & agregasi).
Hive Metastore	Menyimpan metadata tabel, biasanya menggunakan MySQL/PostgreSQL.
Apache Superset	Visualisasi data lalu lintas berdasarkan hasil analisis.

4.4.3 Struktur File di Server

Struktur direktori sistem yang disimpan di dalam server lokal:

```
/opt/bigdata/
├── docker-compose.yml        # Definisi semua layanan (Hadoop, Hive, Spark, Superset)
├── data/                     # Folder volume HDFS lokal
│   ├── bronze/               # Data GPS mentah
│   ├── silver/               # Data hasil cleansing dan normalisasi
│   └── gold/                 # Data hasil agregasi dan fitur model
├── scripts/
│   ├── ingest.sh             # Script untuk mengunduh dan upload data GPS ke HDFS
```

```
(bronze)
│   └── etl_spark.py          # Script Spark untuk ETL: cleansing dan agregasi
└── logs/                     # Direktori penyimpanan log (Spark, ingestion, job)
```

4.5 Pengujian (Testing)

Pengujian dilakukan untuk memastikan bahwa seluruh proses dalam pipeline batch berjalan dengan benar, mulai dari proses ingestion data GPS hingga penyajian data dalam bentuk visualisasi di dashboard. Pengujian ini mencakup berbagai jenis pengujian dari tingkat unit hingga end-to-end, dengan pendekatan yang terstruktur dan dapat direproduksi.

4.5.1 Jenis Pengujian

Tabel 12. Jenis dan Tujuan Pengujian

Jenis Pengujian	Tujuan
Unit Test	Memastikan setiap script berjalan secara mandiri, seperti etl_spark.py.
Integration Test	Memastikan konektivitas dan alur antar komponen: ingestion → HDFS → Spark → Hive.
Data Quality Test	Memastikan tidak ada missing values, duplikasi, dan format data konsisten.
Performance Test	Mengukur waktu eksekusi ingestion dan job Spark batch.
End-to-End Test	Menjalankan seluruh pipeline dari pengambilan data sampai muncul di dashboard Superset.

4.5.2 Kasus Uji (Test Cases)

Tabel 13. Kasus Uji

No	Kasus Uji	Deskripsi	Status
1	Data berhasil diambil dari API	File CSV berhasil diunduh dan tersimpan di /data/bronze/.	✅
2	Spark berhasil membersihkan data	Dataset di silver layer tidak mengandung missing/null.	✅
3	Spark menghasilkan ringkasan agregat	File output agregat tersedia di /data/gold/.	✅
4	Hive membaca tabel gold layer	Query SELECT berhasil dijalankan di Hive.	✅

5	Superset menampilkan grafik lalu lintas	Dashboard memuat dan menampilkan data agregat dengan benar.	✅
6	Log file tertulis saat job dijalankan	File log tersedia di /logs/job.log setelah job Spark dijalankan.	✅

4.5.3 Tools Pengujian

Tabel 14. Alat Pengujian

Tools	Fungsi
Jupyter Notebook	Validasi manual output data pada Spark dan query Hive.
Bash + Log File	Logging proses ingestion dan Spark jobs.
Hive CLI	Menjalankan query SQL untuk uji keterbacaan data di Hive.
Superset	Validasi visualisasi data oleh pengguna akhir (end-user validation).

Pengujian ini memastikan bahwa sistem berjalan stabil, data valid, serta pipeline batch berjalan sesuai harapan. Semua komponen saling terintegrasi dengan baik dan dapat ditelusuri melalui log dan visualisasi akhir.

4.6 Analitik

Analisis lanjutan dilakukan terhadap data lalu lintas dan cuaca yang telah diproses hingga Gold Layer untuk menghasilkan model prediktif dan wawasan berbasis machine learning. Proses ini menggunakan Apache Spark MLlib untuk pelatihan model, Hive untuk menyimpan hasil analitik, Jupyter/VSCode untuk eksperimen awal, dan Superset sebagai platform visualisasi interaktif.

4.6.1 Tujuan Analitik Machine Learning

- **Memprediksi tingkat kemacetan jalan** berdasarkan data historis GPS kendaraan (misalnya, kecepatan rata-rata, curah hujan, lokasi).

4.6.2 Data Input untuk MLlib

1. **Sumber**: Gold Layer dari data agregat lalu lintas.
2. **Format**: Parquet atau ORC.
3. **Kolom Data Utama**:
    - Road_name
    - Hour
    - Avg_speed_kmh
    - Count_vehicles

- Rainfall_mm
- Temperature_c
- Visibility_km
- Latitude
- Longitude
- congestion_level

4.6.3 Metodologi Analitik

Tabel 15. Metodologi

No	Analisis	Algoritma Spark MLlib	Tujuan
1	Prediksi tingkat kemacetan	Random Forest	Memprediksi kemacetan berdasarkan waktu, cuaca, jumlah kendaraan, dll

4.6.4 Tahapan Analitik

1. **Load Data**: Baca data Gold Layer menggunakan Spark SQL.
2. **Preprocessing:**
   - *Isi nilai kosong*: Kolom road_name yang kosong diisi dengan "Unknown". Kemudian untuk nilai numerik (misalnya, avg_speed_kmh, rainfall_mm) diisi dengan 0 jika kosong.
   - *Encoding kategorikal*: Encode road_name menjadi road_index.
   - *Normalisasi*: Normalisasi fitur numerik menggunakan MinMaxScaler.
3. **Splitting Data:**
   - Bagi data menjadi 80% train dan 20% test.
4. **Modeling:**
   - Menggunakan Random Forest Regressor untuk memprediksi.
5. **Evaluasi:**
   - Menggunakan RMSE dan MAE untuk mengukur akurasi prediksi.
6. **Saving Model:**
   - Simpan model ke HDFS.
7. **Inference:**
   - Jalankan prediksi dan simpan hasil di tabel analitik Hive atau folder Gold/Insight.

4.6.5 Tools Analitik

Tabel 16. Alat dan Fungsi

Tools	Fungsi
Apache Spark MLlib	Pemodelan prediksi (Random Forest) untuk tingkat kemacetan berdasarkan data trafik dan cuaca.

Hive Metastore	Penyimpanan hasil inferensi dan metadata.
Jupyter / VSCode	Eksplorasi awal dan eksperimen pipeline analitik.
Apache Superset	Visualisasi pola kemacetan, prediksi dan segmentasi jalan.

4.7 Dataset

Data dari dua sumber (data simulasi trafik taksi dan data cuaca dummy) dikumpulkan menjadi satu dalam data lake melalui batch processing untuk analisis prediksi kemacetan di kota Medan. Data ini digunakan untuk memahami hubungan antara kondisi cuaca (misalnya, curah hujan) dan kecepatan kendaraan, yang dapat mengindikasikan kemacetan.

4.7.1 Deskripsi Dataset

Dataset 1: simulasi-trafik_medan.csv

Tabel 17. Contoh Data 1

timestamp	latitude	longitude	speed_kmh	taxi_id	index_right	road_name
2013-07-01, 00:01:13	3.578676	98.682801	3.953903	20000589	25166	
2013-07-01, 00:01:28	3.57981	98.680974	57.304474	20000589	25166	Jalan Teratai
2013-07-01, 06:39:43	3.566751	98.681199	0.759373	20000272	43848	Jalan Komodor Muda Adi Sucipto
2013-07-01, 06:40:28	3.566760	8.681217	0.239716	20000272	77991	Jalan Komodor Muda Adi Sucipto

Deskripsi: Data simulasi pergerakan taksi di Medan pada 1 Juli 2013, mencakup lokasi, kecepatan, dan nama jalan. Data ini digunakan untuk menganalisis pola kemacetan berdasarkan kecepatan kendaraan.

Tabel 18. Deskripsi Data 1

Kolom	Tipe Data	Deskripsi
timestamp	String	Waktu pengukuran pergerakan taksi (format: YYYY-MM-DD HH:mm:ss)
latitude	Float	Koordinat lintang lokasi taksi
longitude	Float	Koordinat bujur lokasi taksi
speed_kmh	Float	Kecepatan taksi dalam kilometer per jam

taxi_id	Integer	ID unik taksi
index_right	Integer	Indeks internal untuk pelacakan data
road_name	String	Nama jalan tempat taksi berada

Dataset 2: cuaca_medan_dummy.csv

Tabel 18. Contoh Data 2

date_time	**location**	**rainfall_mm**	**temperature_c**	**humidity_percent**	**visibility_km**
2013-07-01 00:00:00	Medan	0.00	24.30	83.40	8.98
2013-07-01 01:00:00	Medan	0.00	26.93	81.05	8.93
2013-07-01 06:00:00	Medan	10.29	26.36	91.29	5.11
2013-07-01 07:00:00	Medan	0.00	28.02	84.47	8.92

Deskripsi: Data cuaca simulasi per jam untuk Medan pada 1 Juli 2013, dibuat berdasarkan iklim tropis Medan (musim kemarau, hujan sesekali) dan dikorelasikan dengan kecepatan rata-rata taksi. Data ini penting untuk analisis dampak cuaca (misalnya, hujan) terhadap kemacetan.

Tabel 19. Deskripsi Data 2

Kolom	**Tipe Data**	**Deskripsi**
date_time	String	Waktu pengukuran cuaca (format: YYYY-MM-DD HH:mm:ss)
location	String	Lokasi pengukuran cuaca (selalu "Medan")
rainfall_mm	Float	Curah hujan dalam milimeter per jam (0 = tidak hujan, >5 = hujan sedang/lebat)
temperature_c	Float	Suhu dalam derajat Celcius
humidity_percent	Float	Kelembaban relatif dalam persen
visibility_km	Float	Jarak pandang dalam kilometer (rendah saat hujan, tinggi saat cerah)

4.7.2 Format Penyimpanan

Data diproses dan disimpan dalam data lake di Hadoop dengan tiga lapisan:

1. **Bronze Layer (Raw):**
   Format: CSV
   Path:
   - Trafik: /bronze/trafik/2013-07-01.csv
   - Cuaca: /bronze/cuaca/2013-07-01.csv

   Deskripsi: Data mentah dari sumber asli (simulasi trafik dan cuaca dummy) tanpa pembersihan, disimpan langsung setelah ingestion.

2. **Silver Layer (Cleaned):**
   Format: Parquet
   Path: Gabungan: /silver/trafik_cuaca/2013-07.parquet
   Deskripsi: Data yang telah dibersihkan (misalnya, format timestamp diseragamkan, nilai kosong di road_name ditangani) dan digabungkan berdasarkan timestamp (trafik dibulatkan ke jam terdekat). Disimpan dalam format Parquet untuk efisiensi penyimpanan dan query.

3. **Gold Layer (Aggregated):**
   Format: ORC
   Path: /gold/trafik_cuaca_agg/2013-07.orc
   Deskripsi: Data agregasi untuk analisis prediktif kemacetan, misalnya, kecepatan rata-rata, total curah hujan, dan jumlah pengukuran per jam, jalan, atau koordinat. Digunakan untuk laporan atau model machine learning.

4.8 Aliran Data (Pipeline)

```
[1] Data Source (CSV, JSON, XML, API, FTP, DB)
├── simulasi-trafik_medan.csv
└── cuaca_medan_dummy.csv

[2] Bronze Layer (HDFS - Raw Zone)
├── Simpan file mentah ke HDFS (/datalake/bronze/)
├── Format tetap CSV/JSON (belum diproses)
└── Metadata tracking dicatat (timestamp, sumber, ukuran) via Hive
Metastore

[3] Ingestion &amp;amp; Staging (Apache Spark + Hive) – Bronze ke
Silver
├── Gunakan Spark untuk baca data dari HDFS (bronze)
├── Lakukan parsing, validasi skema, handling missing values
├── Standardisasi waktu, format tanggal, dan satuan
└── Tulis ke Silver Layer dalam format kolumnar (Parquet/ORC)
```

```
[4] Silver Layer (HDFS - Clean Zone)
├── Path: /datalake/silver/
├── Data sudah bersih dan distandardisasi
├── Join antar dataset (ekspor + cuaca + biaya + kurs)
└── Simpan hasil ke format Parquet untuk efisiensi query


[5] Enrichment &amp;amp; Transformation (Apache Spark) - Silver ke Gold
├── Hitung agregasi bulanan, nilai ekspor (dalam USD/IDR)
├── Integrasi permintaan global
└── Siapkan data untuk analitik &amp;amp; ML

[6] Gold Layer (HDFS - Curated Zone)
├── Path: /datalake/gold/
├── Dataset siap pakai untuk analitik dan machine learning
├── Format: Parquet atau Hive Tables
└── Gunakan Hive untuk query BI,dan Spark MLlib untuk model
prediksi

[7] Analytics Layer
├── Apache Hive (OLAP + BI)
├── Apache Superset (dashboard)
└── Apache Spark MLlib (Prediksi tingkat kemacetan)

[8] Monitoring &amp;amp; Orchestration
├── Apache Airflow → Workflow terjadwal
└── Apache Ambari → Monitoring cluster &amp;amp; resource
```

5. Lampiran

5.1 Repositori Portofolio

Link Repositori : SQUAD MACET_KELOMPOK 16

5.2 Lampiran Code

Coming Sooonnn