Project & Tools Setup

Big Data Analytics

Professor Anis Koubaa

SE 446
Alfaisal University
https://github.com/aniskoubaa/big_data_course

Spring 2026

Today's Agenda

1. Semester Project Overview
2. 5 Milestones Explained
3. Datasets Overview
4. GitHub Setup & Workflow
5. Google Colab & Databricks
6. ExamGPT for Submissions
7. Setup Checklist & Q&A

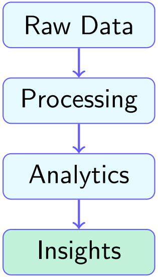
Smart City Data Platform

Your Mission

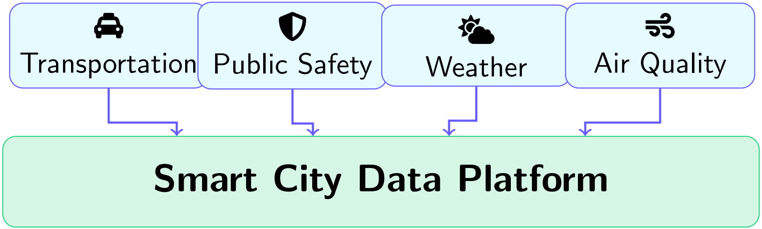
Build an end-to-end **Big Data analytics platform** using real urban datasets.

What You'll Build:

- Data ingestion pipelines
- MapReduce processing jobs
- SQL analytics with Hive
- Spark-based transformations
- Real-time streaming dashboards



Why a Smart City Project?



Transportation

Public Safety

Weather

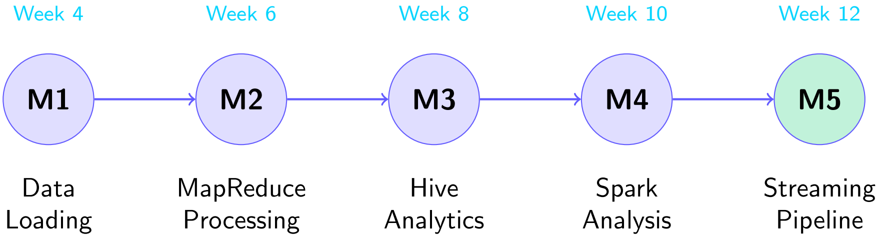
Air Quality

Smart City Data Platform

Real-World Relevance

Smart cities generate **massive amounts of data**. Learn to process it!

Project Milestones Overview



Week 4 — M1 — Data Loading

Week 6 — M2 — MapReduce Processing

Week 8 — M3 — Hive Analytics

Week 10 — M4 — Spark Analysis

Week 12 — M5 — Streaming Pipeline

Each Milestone	Weight	Total
4%	× 5 milestones	= 20% of course grade

M1: Data Loading (Week 4)

Objective

Load and explore datasets in a distributed environment

Key Tasks:

- Understand HDFS concepts
- Load CSV/JSON files
- Explore data schemas
- Basic data profiling
- Handle missing values

Technologies

- Google Colab
- Pandas basics
- HDFS concepts
- Data formats

Deliverable:

Jupyter notebook with data loaded and explored

M1: Smart City Context

Why This Matters for Smart Cities

Cities collect **millions of records daily** from taxis, sensors, and systems. Before any analysis, data must be loaded and understood.

Real-World Problem:

- NYC Taxi: 400K+ trips/day
- Data arrives in various formats
- Missing values common
- Schema changes over time

Why Big Data Tools?

- Excel can't handle 1M+ rows
- Need distributed storage (HDFS)
- Automated profiling essential
- Scalable from day one

Learning Outcome

Understand how to prepare urban data for analysis at scale.

M1: Sample Code Preview

Loading and Exploring Data with Pandas

```python
import pandas as pd

# Load the NYC Taxi dataset
df = pd.read_csv(’nyc_taxi_data.csv’)

# Display basic info
print(f"Shape: {df.shape}")
print(df.columns.tolist())

# Check for missing values
print(df.isnull().sum())

# Basic statistics
print(df.describe())
```

What You’ll Learn

- Load CSV/JSON data
- Explore data schemas
- Handle missing values
- Basic data profiling
- Pandas DataFrame ops

Key Concepts

`read_csv()`, `shape`, `isnull()`, `describe()`

M2: MapReduce Processing (Week 6)

Objective

Implement batch processing with MapReduce paradigm

Key Tasks:

- Write custom mappers
- Write custom reducers
- Aggregation operations
- Word count patterns
- Data transformations

Technologies

- MapReduce concepts
- mrjob (Python)
- Databricks

Deliverable:

MapReduce jobs processing taxi/crime data

M2: Smart City Context

Why This Matters for Smart Cities

Crime analysis requires processing millions of incident records to identify patterns, hotspots, and trends.

Real-World Problem:

- Chicago: 500K+ crime records/year
- Need to count by type, location
- Traditional SQL too slow
- Single machine can't scale

Why MapReduce?

- Parallel processing across nodes
- Fault-tolerant execution
- Scales horizontally
- Perfect for batch aggregation

Learning Outcome

Learn to parallelize computations when data is too large for one machine.

M2: Sample Code Preview

MapReduce: Count Crimes by Type

```
from mrjob.job import MRJob

class CrimeTypeCount(MRJob):

    def mapper(self, _, line):
        if not line.startswith(’ID’):
            fields = line.split(’,’)
            crime_type = fields[5]
            yield crime_type, 1

    def reducer(self, key, counts):
        yield key, sum(counts)

if __name__ == ’__main__’:
    CrimeTypeCount.run()
```

What You’ll Learn

- Map function (key-value)
- Reduce function (aggregate)
- Distributed processing
- Data transformations

Key Concepts

mapper(), reducer(), yield, word count pattern

M3: Hive Analytics (Week 8)

Objective

Perform SQL analytics on Big Data using Hive

Key Tasks:

- Create Hive tables
- Write HiveQL queries
- Joins and aggregations
- Partitioning strategies
- Performance optimization

Technologies

- Apache Hive
- HiveQL (SQL-like)
- Databricks SQL

Deliverable:

Analytics queries answering business questions

M3: Smart City Context

🚗 Why This Matters for Smart Cities

City planners need **SQL-like analytics** on transportation data to optimize routes, pricing, and schedules.

Real-World Problem:

- Find peak hours for taxi demand
- Calculate average fares by zone
- Identify underserved areas
- Optimize fleet allocation

Why Hive/SQL?

- Familiar SQL syntax
- Runs on distributed data
- Business users can query
- No coding required

Learning Outcome

Enable business analysts to query Big Data without programming.

M3: Sample Code Preview

HiveQL: Analyze Taxi Trip Patterns

```sql
-- Create external table
CREATE EXTERNAL TABLE taxi_trips (
    pickup_datetime STRING,
    passenger_count INT,
    trip_distance DOUBLE,
    fare_amount DOUBLE
) ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ’,’;

-- Average fare by hour
SELECT HOUR(pickup_datetime) as hr,
       AVG(fare_amount) as avg_fare
FROM taxi_trips
GROUP BY HOUR(pickup_datetime);
```

What You’ll Learn

- Create Hive tables
- SQL on Big Data
- Aggregations (AVG, COUNT)
- Time-based analysis

Key Concepts

`CREATE TABLE`, `GROUP BY`, `HOUR()`, external tables

M4: Spark Analysis (Week 10)

Objective

Leverage Spark for advanced data processing

Key Tasks:

- PySpark DataFrames
- Spark SQL queries
- Data transformations
- Joining multiple datasets
- Performance tuning

Technologies

- Apache Spark
- PySpark
- Spark SQL
- Databricks

Deliverable:

Spark notebook with multi-dataset analysis

M4: Smart City Context

⛅ Why This Matters for Smart Cities

Understanding **cross-domain correlations** (weather × transportation) improves city services.

Real-World Problem:

- How does rain affect taxi tips?
- Do snow days increase demand?
- Air quality impact on ridership?
- Need to join multiple datasets

Why Apache Spark?

- 100x faster than MapReduce
- In-memory processing
- Easy joins across datasets
- Python-friendly (PySpark)

Learning Outcome

Combine multiple data sources to discover urban insights.

M4: Sample Code Preview

PySpark: Join Taxi with Weather

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *

spark = SparkSession.builder \
    .appName("TaxiWeather").getOrCreate()

# Load datasets
taxi = spark.read.csv("taxi.csv", header=True)
weather = spark.read.csv("weather.csv", header=
    True)

# Join on date
joined = taxi.join(weather,
    to_date(taxi.pickup_datetime) == weather.date
        )

# Analyze by precipitation
```

What You'll Learn

- PySpark DataFrames
- Multi-dataset joins
- Spark SQL functions
- Data transformations

Key Concepts

SparkSession, join(), groupBy(), agg()

M5: Streaming Pipeline (Week 12)

Objective

Build a real-time data streaming pipeline

Key Tasks:

- Kafka producer/consumer
- Spark Structured Streaming
- Real-time aggregations
- Window operations
- Dashboard visualization

Technologies

- Apache Kafka
- Spark Streaming
- Delta Lake
- Databricks

Deliverable:

Complete streaming pipeline with visualization

M5: Smart City Context

Why This Matters for Smart Cities

Real-time monitoring enables **immediate response** to traffic, emergencies, and service disruptions.

Real-World Problem:

- Live taxi demand tracking
- Traffic congestion alerts
- Emergency response dispatch
- Surge pricing triggers

Why Streaming?

- Batch processing too slow
- Decisions in seconds, not hours
- Continuous data ingestion
- Real-time dashboards

Learning Outcome

Build systems that react to city events as they happen.

M5: Sample Code Preview

Spark Streaming: Real-time Monitoring

```
from pyspark.sql.functions import *

# Read streaming data from Kafka
stream = spark.readStream \
    .format("kafka") \
    .option("subscribe", "taxi_events") \
    .load()

# Count trips per 5-min window
counts = stream \
    .withWatermark("ts", "10 min") \
    .groupBy(window("ts", "5 min")) \
    .count()

# Output to dashboard
counts.writeStream \
    .format("console").start()
```

What You'll Learn

- Kafka integration
- Structured Streaming
- Window operations
- Real-time aggregations

Key Concepts

`readStream`, `window()`, `watermark`, `writeStream`

Datasets We'll Use

Dataset	Size	Records	Description
NYC Yellow Taxi	∼50 MB	1M+ trips	Trip records, fares, locations
Chicago Crimes	∼30 MB	500K+	Crime types, dates, GPS coords
NYC Weather	∼5 MB	10K+	Daily temp, precipitation
Air Quality Index	∼3 MB	5K+	Daily AQI by city

Pre-Hosted

All datasets are already hosted. No downloading needed!

Real Data

Based on actual open government data sources.

Where Are Datasets Hosted?

Original Source: Kaggle

Full datasets available on Kaggle (free account required)

Dataset	Kaggle Link
NYC Yellow Taxi	`kaggle.com/datasets/elemento/nyc-yellow-taxi-trip-data`
Chicago Crimes	`kaggle.com/datasets/chicago/chicago-crime`
NYC Weather	`kaggle.com/datasets/danbraswell/new-york-city-weather-data-2019`
Air Quality Index	`kaggle.com/datasets/programmerrdai/air-quality-data-2012-to-2024`

Course Samples

Small CSV samples provided in:

`github.com/aniskoubaa/big_data_course/data/`

For Class Work

- Use GitHub samples for testing
- Use Kaggle for full analysis
- Both work in Colab/Databricks

NYC Yellow Taxi Dataset

Key Fields:

- `pickup_datetime` - When trip started
- `dropoff_datetime` - When trip ended
- `passenger_count` - Number of passengers
- `trip_distance` - Miles traveled
- `fare_amount` - Base fare
- `tip_amount` - Tip given
- `pickup_location` - Start zone
- `dropoff_location` - End zone

Sample Questions

- What's the average trip distance?
- Which hours are busiest?
- How do tips vary by time of day?
- Most popular pickup locations?

NYC Yellow Taxi - Sample Data

Sample Records from `nyc_taxi_data.csv`

pickup_datetime	dropoff_datetime	passengers	distance	fare	tip
2024-01-15 08:23:00	2024-01-15 08:45:00	2	3.5	18.50	4.00
2024-01-15 09:10:00	2024-01-15 09:25:00	1	1.8	12.00	2.50
2024-01-15 12:45:00	2024-01-15 13:30:00	3	8.2	32.00	6.40
2024-01-15 18:00:00	2024-01-15 18:15:00	1	2.1	14.50	3.00
2024-01-15 22:30:00	2024-01-15 23:00:00	4	5.6	25.00	5.00

Data Insights

- **1M+ records** spanning several months
- Timestamps allow time-based analysis (rush hours, weekends)
- Fare and tip data enable financial analysis

Chicago Crimes Dataset

Key Fields:

- `date` - When crime occurred
- `primary_type` - Crime category
- `description` - Detailed type
- `location_description` - Where
- `arrest` - Was arrest made?
- `latitude`, `longitude` - GPS
- `district` - Police district

Sample Questions

- Most common crime types?
- Crime trends by month?
- Arrest rate by crime type?
- Hotspot locations?

Chicago Crimes - Sample Data

Sample Records from `chicago_crimes.csv`

date	primary_type	location	district	arrest
2024-01-10 14:30	THEFT	STREET	11	False
2024-01-10 22:15	BATTERY	APARTMENT	7	True
2024-01-11 03:45	BURGLARY	RESIDENCE	3	False
2024-01-11 16:20	ASSAULT	SIDEWALK	11	True
2024-01-12 11:00	THEFT	RETAIL STORE	1	True

Data Insights

- **500K+ records** with GPS coordinates
- 30+ crime types for categorical analysis
- Arrest boolean enables outcome analysis

NYC Weather Dataset

Key Fields:

- `date` - Calendar date
- `temp_max` - Maximum temperature (°F)
- `temp_min` - Minimum temperature (°F)
- `temp_avg` - Average temperature
- `precipitation` - Rainfall (inches)
- `snow` - Snowfall (inches)
- `wind_speed` - Avg wind speed

Sample Questions

- How does weather affect taxi usage?
- Correlation between rain and trips?
- Seasonal patterns in ridership?
- Snow days vs. normal days?

NYC Weather - Sample Data

Sample Records from `nyc_weather.csv`

date	temp_max	temp_min	temp_avg	precip	snow	wind
2024-01-15	42	28	35	0.00	0.0	8.5
2024-01-16	38	25	31	0.25	2.1	12.3
2024-01-17	35	22	28	0.00	0.0	6.2
2024-01-18	45	32	38	0.10	0.0	9.1
2024-01-19	52	40	46	0.50	0.0	15.4

Data Insights

- **10K+ records** spanning multiple years
- Perfect for joining with taxi/crime data by date
- Enables weather-impact analysis on urban activities

Air Quality Index Dataset

Key Fields:

- date - Measurement date
- city - City name
- aqi_value - Air Quality Index (0-500)
- aqi_category - Good/Moderate/Unhealthy
- main_pollutant - PM2.5, Ozone, etc.
- pm25 - Fine particulate matter
- ozone - Ozone level

Sample Questions

- Which cities have worst air quality?
- Seasonal AQI patterns?
- Correlation with weather?
- Days exceeding safety thresholds?

Air Quality Index - Sample Data

Sample Records from `air_quality.csv`

date	city	aqi_value	category	pollutant	pm25
2024-01-15	New York	45	Good	PM2.5	12.3
2024-01-15	Chicago	68	Moderate	Ozone	18.5
2024-01-16	New York	82	Moderate	PM2.5	28.1
2024-01-16	Chicago	55	Moderate	PM2.5	15.2
2024-01-17	New York	38	Good	Ozone	8.7

Data Insights

- **5K+ records** for multiple cities
- AQI categories enable health-impact analysis
- Can be joined with weather data for correlation studies

Why GitHub?

For You

- Version control for your code
- Backup in the cloud
- Portfolio for future jobs
- Industry-standard skill

For the Course

- Track individual contributions
- Team collaboration
- Code review capability
- Submission timestamps

⚠ **Important:** Your GitHub commits are part of your grade!

GitHub Account Setup

1. Go to `https://github.com`
2. Click "Sign Up" and create account
3. **Username Tips:**
   - Use a professional username
   - Example: `ahmed-alsaud` or `fatima_alfarsi`
   - Avoid: `coolhacker123`, `xXx_gamer`
4. Verify your email address
5. Apply for **GitHub Student Developer Pack**:
   - `https://education.github.com/pack`
   - Free Pro features with your `@alfaisal.edu` email

Team Repository Structure

Repository Organization:

Each team gets **ONE shared repository**

```
se446-team-01/
  milestone_1/
    student_ahmed/
    student_fatima/
  milestone_2/
    student_ahmed/
    student_fatima/
  ...
```

Important Rules:

- Each student works in their **own folder**
- Individual commits are tracked separately
- Work on your assigned tasks only
- Quality matters more than quantity

Note

Your individual contributions will be evaluated based on your folder's commits

Essential Git Commands

Command	Purpose
`git clone <url>`	Download repository
`git pull`	Get latest changes
`git status`	Check what's changed
`git add .`	Stage all changes
`git commit -m "message"`	Save changes locally
`git push`	Upload to GitHub

Daily Workflow

`git pull` → Do work → `git add .` → `git commit -m "M1:  ..."` → `git push`

Commit Message Standards

Format

`<MILESTONE>: <Short description of what you did>`

✓ Good Examples:

- `M1: Load taxi data and check schema`
- `M2: Implement crime count mapper`
- `M3: Add average fare HiveQL query`
- `M4: Join weather with taxi data`

✗ Bad Examples:

- `update` ← Too vague
- `asdfasdf` ← Meaningless
- `done` ← What's done?
- `fixed stuff` ← What stuff?

Google Colab Setup

What is Google Colab?

Free cloud-based Jupyter notebooks with Python pre-installed.

Features:

- ✓ No installation needed
- ✓ Python + libraries ready
- ✓ Free GPU access
- ✓ Google Drive integration
- ✓ Easy sharing

Setup Steps:

1. Go to `https://colab.google.com`
2. Sign in with Google account
3. Click “New Notebook”
4. You’re ready!

We’ll Use Colab For:

- Weeks 1-4
- Python & Pandas basics
- M1: Data Loading

Databricks Community Edition

What is Databricks?

Industry-standard cloud platform for Big Data processing with Apache Spark.

Features:

- ✓ Apache Spark pre-configured
- ✓ Notebooks with clusters
- ✓ Hive, SQL, Streaming
- ✓ Free Community Edition
- ✓ Industry-used platform

Setup Steps:

1. Go to `https://databricks.com/try`
2. Select "Community Edition"
3. Create account (use `@alfaisal.edu`)
4. Verify email
5. Create a cluster

We'll Use Databricks For:

- Weeks 5-12
- M2-M5: MapReduce, Hive, Spark, Streaming

Colab vs Databricks

Feature	Google Colab	Databricks
Primary Use	Python/Pandas	Big Data (Spark)
Spark Support	Limited	Native
Setup Difficulty	Easy	Moderate
Cost	Free	Free (Community)
Industry Use	Learning	Production
When We Use	Weeks 1-4	Weeks 5-12

Bottom Line

Start with **Colab** for basics, graduate to **Databricks** for real Big Data.

What is ExamGPT?

Definition

ExamGPT is an AI-powered platform for in-class quizzes and submissions.

How We'll Use It:

- Last 15 minutes of each class
- Quick comprehension checks
- Attendance verification
- Instant feedback on answers

URL: `https://examgpt.aniskoubaa.org`

Benefits

- ✓ Instant grading
- ✓ AI-powered feedback
- ✓ Tracks your progress
- ✓ Accessible anywhere

Counts toward attendance!

ExamGPT Setup

1. Go to `https://examgpt.aniskoubaa.org`
2. Click “Register” or “Sign Up”
3. Use your `@alfaisal.edu` email
4. Enter your:
   - Full name (as in university records)
   - Student ID
5. Verify your email
6. Join the **SE446** course when prompted

Important

Use the **same name** as your official university registration!

Today's Setup Checklist

□	Create GitHub account	`github.com`
□	Create Google account (for Colab)	`google.com`
□	Sign up for ExamGPT	`examgpt.aniskoubaa.org`
□	Clone the course repository	See next slide
□	Request to join course GitHub org	Announced in class

Clone the Course Repository

Course Repository

`https://github.com/aniskoubaa/big_data_course`

Option 1: Command Line

- Open Terminal (Mac/Linux) or Git Bash (Windows)
- Run: `git clone https://github.com/aniskoubaa/big_data_course.git`

Option 2: GitHub Desktop

- Download from `https://desktop.github.com`
- Click "Clone Repository"
- Paste the URL

Option 3: Web Download

- Click green "Code" button on GitHub
- Select "Download ZIP"

Getting Help

During Class

- Raise your hand
- Ask the TA
- Help your neighbor

Office Hours

- Prof. Koubaa: Office SG-10
- By appointment
- `akoubaa@alfaisal.edu`

Online Resources

- Course GitHub Issues
- Moodle Discussion Forum
- Google & Stack Overflow
- AI Assistants (for learning)

Important Links

- Course: `aniskoubaa.org/se446`
- Moodle: Alfaisal LMS

Questions?

Let's do the setup together!

Prof. Anis Koubaa
akoubaa@alfaisal.edu

https://github.com/aniskoubaa/big_data_course

Hands-On Time: Set up your accounts now!