Optimizing High-Performance Data Processing for Web Crawling on PetFinder.my

Presented by Group 5

Neo Zheng Weng A22EC0093
Ng Shu Yu A22EC0228
Muhammad Safwan Bin Mohd Azmi A22EC0221
Navasarathy A/L S.Ganeswaran A22EC0091

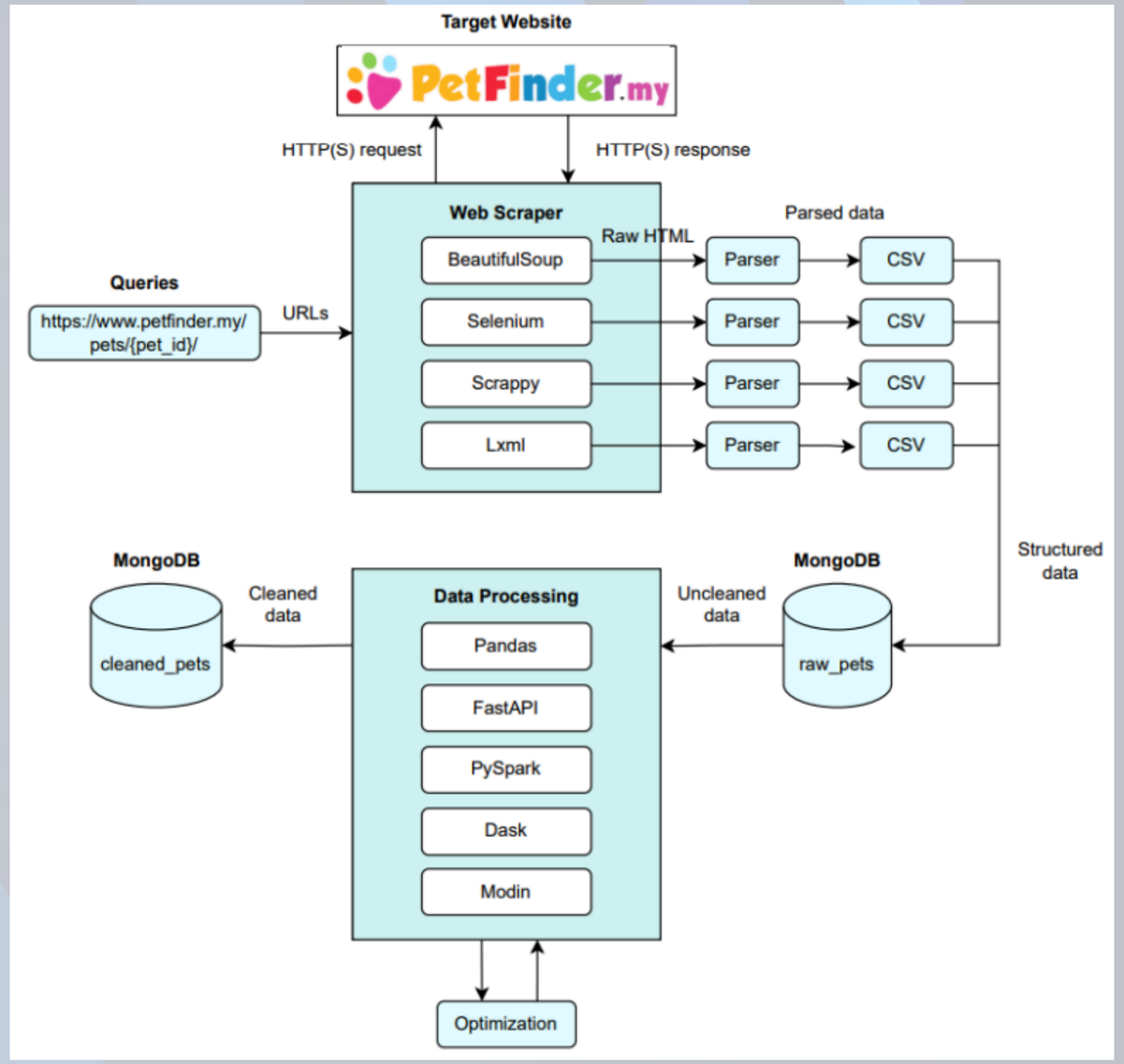
INTRODUCTION

This project builds a high-performance pipeline to scrape and process over 100,000 structured pet listings from PetFinder.my using tools like Scrapy, BeautifulSoup, Selenium, and lxml; stores the data in MongoDB; and benchmarks initial processing with Pandas before applying and comparing optimized techniques—including PySpark, Dask, Modin, and asynchronous processing—by measuring execution time, resource usage, and throughput to identify the most efficient approach for large-scale data analysis.

OBJECTIVES

- Scrape at least 100,000 pet records from PetFinder Malaysia.
- Clean and transform data using Python libraries.
- Apply optimization methods like multithreading, multiprocessing, and distributed computing.
- Compare performance before and after optimization using execution time, memory, CPU usage, and throughput.

SYSTEM ARCHITECTURE



WEB CRAWLING METHOD

Libraries used:

- BeautifulSoup
- Selenium
- Scrapy
- lxml

Crawling methods:

- Direct URL-based crawling
- Rate-limiting
- Asynchronous concurrent crawling
- Error handling
- User-Agent spoofing
- Save data progressively

RECORDS COLLECTED

A total of 101730 pets were collected from the website

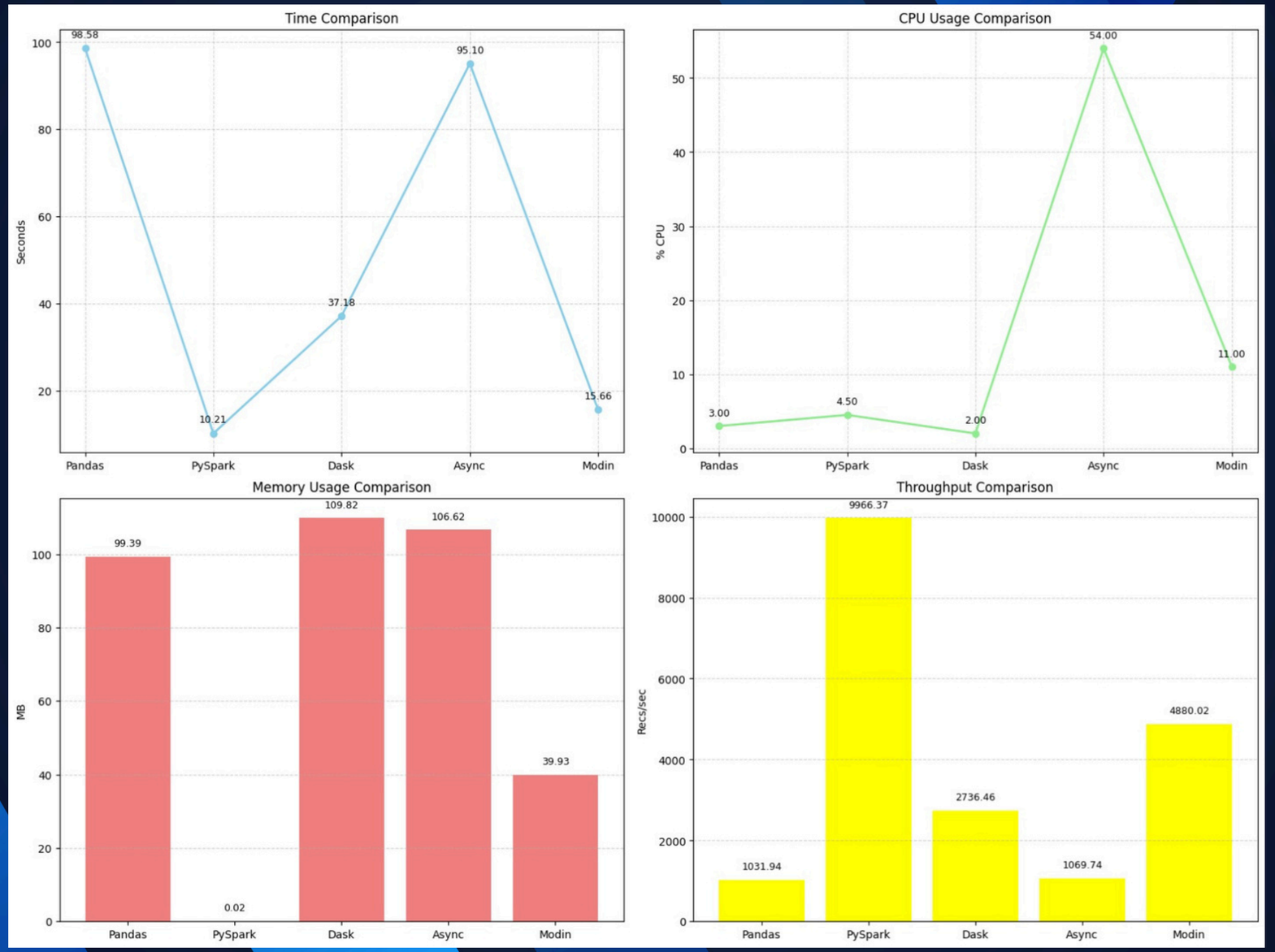
With 18 features:

Pet ID, Name, Type, Species, Profile, Amount, Vaccinated, Dewormed, Spayed, Condition, Body, Color, Location, Posted Date, Price, Uploader Type, Uploader Name, Status

DATA PROCESSING & OPTIMIZATION

Libraries	Optimization Techniques
FastAPI (Async)	Asynchronous processing
PySpark	Distributed processing
Dask	Distributed / Parallel processing
Modin	Parallel processing

PERFORMANCE CHARTS & GRAPHS



PERFORMANCE EVALUATION

Libraries	Time (s)	Memory (MB)	CPU Usage (%)	Throughput (No of Records/s)
Pandas	98.58	99.39	3.00	1031.94
PySpark	10.21	0.02	4.50	9966.37
Dask	37.1	109.82	2.00	2736.46
Modin	15.66	39.93	11.00	4880.02
FastAPI (Async)	95.10	106.2	54.00	1069.74

PERFORMANCE EVALUATION

Libraries	Evaluation
Pandas	Most user-friendly but slowest, best for small datasets and quick analysis.
FastAPI (Async)	Suitable for I/O-bound tasks but less effective for CPU-heavy data processing.
PySpark	Best overall performer with the highest throughput and fastest execution for large-scale data processing.
Dask	Delivers balanced performance for parallel processing on a single machine or small cluster.
Modin	Offers near-PySpark performance with minimal memory usage and a familiar Pandas-like interface.

CONCLUSION

This project successfully delivered a robust and scalable data processing pipeline for extracting and analyzing over 100,000 pet adoption records from PetFinder.my. Through the integration of ethical web scraping practices, structured data engineering, and high-performance processing techniques, the team effectively addressed the challenges associated with large-scale data handling. The outcomes demonstrate the practical value of optimized data workflows in real-world applications and establish a strong foundation for future enhancements in big data processing and analytics.

THANK YOU