Datamatters Assignment - Part 1: Conceptual planning

Nguyen Hoang Hai

1. Problem Understanding and Analysis

1.1. Key Challenges

Data Heterogeneity

- **Format and Type Diversity:** JSON, XML, plain text, CSV, HTML, and PDF inputs across platforms
- **Structural Inconsistency**: Different naming conventions, nesting levels, and field availability
- **Content Variations**: Platform-specific formatting (HTML, emojis, hashtags, rating systems)

Technical Complexities

- **Volume & Time Constraints:** 10,000 daily reviews requiring processing within 24 hours
- **Quality Issues**: Incomplete data, multilingual content, slang, sarcasm, varying lengths
- **Integration**: Multiple authentication methods, API rate limits, outages, versioning

1.2. Assumptions

- Primarily English language content
- APIs or exports available from all platforms
- Simple positive/negative classification is initially sufficient
- Data privacy compliance required
- System must scale with business growth

1.3. Goals

Technical Goals:

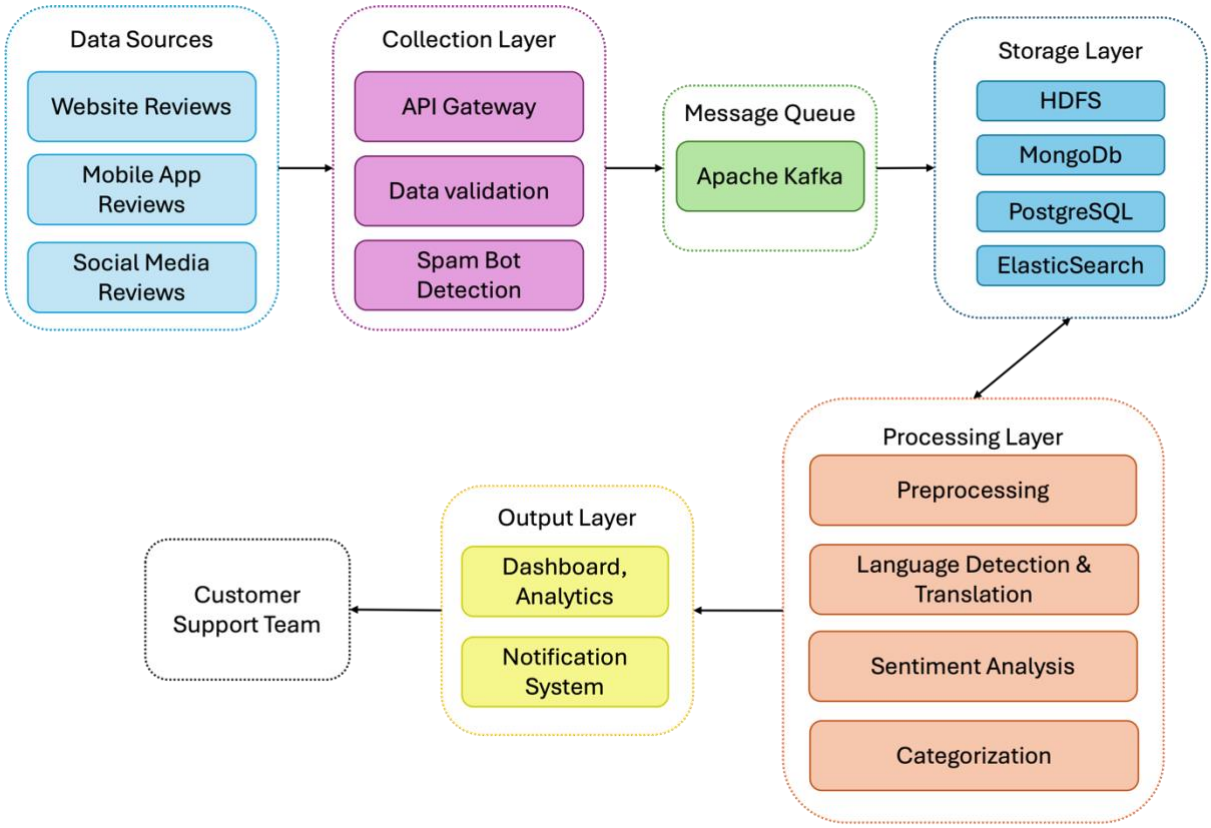
- Meet 24-hour processing window with 85%+ sentiment accuracy
- Create scalable, maintainable architecture supporting multiple input formats
- Implement comprehensive monitoring and error handling

Customer Goals:

- Identify and address negative customer feedback promptly
- Minimize customer loss through early intervention
- Extract valuable insights from customer sentiment trends
- Support product enhancement decisions with feedback data

2. Architecture Design

2.1. System Architecture:



2.2. Architecture Components and Technologies

1. Data Sources

- **Website Reviews:** JSON-formatted data via REST APIs
- **Mobile App Reviews**: Structured data from app stores (Google Play, App Store)
- **Social Media**: Unstructured content via platform APIs (Twitter, Facebook, Instagram)

2. Collection Layer

- **API Gateway (Kong/API Gateway):** Single entry point for all data sources, handling authentication and rate limiting
- **Data Validation (JSON Schema)**: Ensures data meets expected format before processing
- **Spam Detection (ML classifiers)**: Filters out automated/fake reviews using content-based rules and ML models

3. Message Queue

- **Apache Kafka:** Industry-standard message broker for high-throughput, fault-tolerant data streaming
- **Kafka Connect**: For integrating with various data sources
- **Kafka Streams**: For preprocessing and transformations on streaming data

4. Storage Layer

- **MongoDB:** Document store for raw reviews, preserving original structure with flexible schema
- **PostgreSQL**: Relational database for processed results with sentiment metadata
- **Elasticsearch**: Fast, scalable search for customer support to quickly find relevant reviews

5. Processing Layer

- **Preprocessing (Python):** Text cleaning, tokenization, and normalization
- **Language Detection (fastText)**: Identifies review language for proper processing
- **Sentiment Analysis (DistilBERT)**: Performs the core sentiment classification
- **Categorization (TF-IDF/topic modeling)**: Identifies product features or common issues

6. Output Layer

- **Dashboards (Grafana/Tableau/ELK):** Visualizes sentiment trends and key metrics
- **Notification System (webhook/SNS)**: Alerts customer support about negative reviews

Each component works together to form a pipeline that ingests diverse review data, processes it for sentiment, and delivers actionable insights to customer support teams within the required 24-hour window.

2.3. Technology Justifications

- **Apache Kafka:** Message broker enabling asynchronous processing, high throughput, and fault tolerance for reliable review handling
- **MongoDB**: Document database for storing raw reviews with flexible schema to accommodate varying formats
- **PostgreSQL**: Relational database for structured sentiment results, supporting complex queries and analytics
- **Elasticsearch**: Fast full-text search and filtering capabilities for customer support queries
- **Python**: Primary language due to excellent NLP libraries and readability
- **FastAPI**: High-performance API framework with automatic documentation for collection layer
- **Docker/Kubernetes**: Containerization and orchestration for scalable deployment

3. Sentiment Analysis Method Evaluation

3.1. Approach Comparison

Approach	**Performance**	**Scalability**	**Key Considerations**
Rule-based	Very Fast	Excellent	Limited context understanding, rigid rules
Traditional ML (SVM, Naive Bayes, …)	Fast	Good	Feature engineering required
Deep Learning (BERT, etc.)	Slow	Challenging	Resource intensive but context-aware
Hybrid	Medium	Good	Combines strengths but more complex

3.2. Recommended Solution: Fine-tuned DistilBERT

DistilBERT provides an optimal balance for this use case:

- **Accuracy**: Retains ~97% of BERT's accuracy while being 40% smaller
- **Performance**: Significantly faster inference than full BERT models
- **Scalability**: Resource-efficient and supports batch processing
- **Implementation**: Pre-trained model can be fine-tuned on e-commerce reviews

This approach meets the 24-hour processing requirement while delivering sufficient accuracy to identify negative sentiment reliably. The model can be optimized further with quantization techniques and deployed with GPU acceleration for peak loads.