1. System Overview & User Guide

1.1 System Architecture

Our system implements a MongoDB-based document store for analyzing online newspaper articles with two main components:

Phase 1 - Data Loading (`load-json.py`): Connects to MongoDB, drops/recreates the `articles` collection, reads JSON Lines format files, performs batch insertion (5000 documents per batch), and validates data integrity post-load.

Phase 2 - Query Interface (`phase2_query.py`): Provides an interactive menu-driven interface implementing four query operations using MongoDB's query and aggregation framework with flexible input formats and formatted statistical results.

1.2 User Guide

Setup: (1) Install MongoDB: `brew tap mongodb/brew && brew install mongodb-community && brew services start mongodb-community` (2) Setup Python: `python3 -m venv venv && source venv/bin/activate && pip install pymongo` (3) Load data: `python3 load-json.py chunk_1.json 27017` (4) Run queries: `python3 phase2_query.py 27017`

Operation 1 - Most Common Words by Media Type: Analyzes word frequency patterns. Input: media type (news/blog, case-insensitive). Output: Top 5 words with counts, includes ties at 5th position.

Operation 2 - Article Count by Date: Compares publication volume between media types. Input: date (YYYY-MM-DD or "Month Day, Year"). Output: Counts for each type with difference and percentage.

Operation 3 - Top News Sources (2015): Identifies most prolific publishers in 2015. Input: none. Output: Top 5 sources with article counts and percentages.

Operation 4 - Recent Articles by Source: Browse latest content from specific publishers. Input: exact source name. Output: Up to 5 most recent articles with titles and dates.

2. Group Work Breakdown

2.1 Work Distribution

Mahir Islam: Phase 1 complete implementation (load-json.py), batch processing logic, MongoDB connection and error handling for Phase 1, Operation 1 (word frequency) implementation, word extraction using regex, MongoDB native installation and configuration, virtual environment setup, README.md documentation, Git repository management.

William Li: Phase 2 base structure and menu system design, MongoDB connection for Phase 2, Operations 2, 3, and 4 implementation, user input validation and flexible date parsing, output formatting and table presentation, Report.PDF preparation, testing all query operations with edge cases.

Shared Responsibilities: System architecture design, code review for both phases, integration testing, performance optimization, error handling strategy, debugging, final code refinement.

2.2 Time Breakdown

Mahir Islam: MongoDB installation (1h), Python setup (0.5h), Phase 1 development (4h), Operation 1 (2.5h), Testing/debugging Phase 1 (2h), README documentation (2h), Code review (1.5h). **Total: ~13.5 hours**

William Li: Phase 2 menu system (2h), Operation 2 (2h), Operation 3 (2h), Operation 4 (2h), Input validation (1.5h), Output formatting (1.5h), Testing operations (2h), Report preparation (1.5h). **Total: ~14.5 hours**

Joint Sessions: Planning (1.5h), Integration testing (1.5h), Final debugging (1h). **Total: ~4 hours**

Total Project Time: ~32 hours

2.3 Progress Timeline

Week 1 (Nov 18-20): Initial meeting for architecture and responsibility division. Mahir: MongoDB native installation, Phase 1 implementation. William: Menu system design and Phase 2 framework. Daily Discord check-ins.

Week 2 (Nov 21-22): Mahir: Operation 1 completion, README started. William: Operations 2, 3, 4 implementation, input validation. Joint: 2-hour testing session via Discord screen share, identified and fixed integration issues.

Week 3 (Nov 23-24): Mahir: README finalization. William: Output formatting refinement, Report.PDF preparation. Joint: Final code review, comprehensive testing, submission preparation.

2.4 Communication Method

Primary Communication: Discord

3. Assumptions

No assumptions were made.