Università di Pisa

Master's Degree in Artificial Intelligence and Data Engineering

Design and develop of an Application interacting with NoSQL Databases:

Smart News Aggregator

Instructors:
Prof. Pietro Ducange
Prof. Alessio Schiavo

Student:
Adrien Koumgang Tegantchouang

Academic Year 2024/2025

Contents

Chapter 1

Introduction

1.1 Overview

In the digital age, the volume of news articles produced every day is staggering. Readers are increasingly overwhelmed by the amount of available content and often struggle to find reliable and relevant information that matches their personal interests. To address this challenge, we propose the development of a **Smart News Aggregator Reader Personalization Platform**.

This platform collects news articles from multiple external APIs, stores and processes them using NoSQL databases (MongoDB and Redis), and delivers a personalized reading experience to users. It incorporates intelligent recommendation features, secure user authentication via JWT tokens, and real-time analytics to improve user engagement.

1.2 Objectives

The primary objective of this project is to **design and implement a distributed news aggregation application** of intelligently retrieving, storing, analyzing, and serving large-scale multi-structured data using multiple NoSQL database technologies.

In line with the course requirements for **Large Scale and Multi-Structured Databases**, the specific goals of this project include:

- **Data Acquisition & Preprocessing**: Retrieve a real-world, high-volume dataset (50MB) from multiple external news APIs (e.g., MediaStack, newsData, The

Guardian, NYTimes), ensuring **variety** and **velocity**.

- **Distributed NoSQL Architecture**: Use **MongoDB** as the primary **Document Database**, with carefully designed collections, indexes, and aggregation pipelines. Integrate a **Key-Value store** (**Redis**) to optimize caching and access to hot data.
- **System Design and Modeling**: Define functional and non-functional requirements. Design UML class diagrams and user interface mockups. Model the data schema for each NoSQL DB in use.
- **RESTful API Backend**: Build a scalable backend using Flask and Flask-RESTX. Expose secure endpoints with JWT-based authentication. Provide API documentation via Swagger and test interfaces.
- **Advanced Analytics**: Implement at least three real aggregation pipelines in MongoDB for summarizing and analyzing article content and user activity. Provide user-personalized views and filtering using advanced queries.
- **Monitoring, Testing, and Depployment**: Deploy on both and UNIPI virtual clusters. Include performance tests, system logging, and analysis of read/write throughput under different consistency models. Offer a complete presentation and demonstration, including API functionality and analytics insights.

By achieving these goals, the project demonstrates my ability to handle real-world large-scale data applications, integrating theoretical design with practical deployment, and ensuring cross-database consistency and performance.

1.3 Structure of the presentation

The documentation is structured as follows:

- **Chapter 1 - Introduction**: Presents the context, objectives, and structure of the project, outlining the motivations and academic scope.
- **Chapter 2 - Requirements Analysis**: Identifies the functional and non-functional requirements of the application, defines the system actors, and outlines key use cases.

- **Chapter 3 - System Design**: Includes the architectural overview, UML class diagrams, and mockup wireframes of the application's user interface.
- **Chapter 4 - NoSQL Database Modeling**: Describes the design choices for MongoDB (Document DB) and Redis (Key-Value DB), along with schema definitions, key structures, and CAP theorem considerations.
- **Chapter 5 - Data Ingestion and Integration**: Explains how external APIs are connected, data is fetched and transformed, and stored into the databases. Also includes logging and handling of errors and API rate limits.
- **Chapter 6 - Backend Implementation**: Details the development of RESTful API endpoints using Flask and Flask-RESTX, JWT authentication, Swagger documentation, and the modular blueprint structure.
- **Chapter 7 - Advanced Aggregations and Analytics**: Presents non-trivial aggregation pipelines in MongoDB, user-specific personalization features, and analytics insights extracted from the dataset.
- **Chapter 8 - Deployment and Testing**: Discusses deployment on local and virtualized environments, performance testing scenarios, and consistency benchmarking across NoSQL systems.
- **Chapter 9 - Conclusion**: Summarizes achievements, reflects on encountered challenges, and suggests possible extensions to improve scalability, interactivity, and intelligence.

Bibliography

Acknowledgments

I thank God who gave me the strength to do this project.