Large Scale and Multi-Structured Data Bases
University of Pisa
Academic Year 2025-2026
Instructors: Pietro Ducange, Alessio Schiavo
Workgroup Project

Title: Design and develop of an Application interacting with NoSQL Databases

The workgroup should ***design*** a complete application which manages a "big dataset" stored in a distributed data store, built considering at least two NoSQL architectures (it is ***mandatory*** to use a ***Document DB***).

The ***main focus*** of the project is to design the ***distributed data layer*** in terms of data modeling and organization. The implementation of the application must be implemented as a service i) interacting with the DB servers using the specific drivers and ii) exposing its main functionalities by means of Restful APIs. Any kind of programming language and data base management system(s) may be used for the project implementation.

Dataset

The dataset must be a ***real dataset*** characterized by a ***large volume*** (at least 50/100 MB). In addition, it ***should have*** at least one of the following features:

- ***Variety***: Multi-sources and/or multi-format. For example, if an application handles comments on hotels, it will be appreciated if the database is built using different sources for comments such as TripAdvisor and Booking.
- ***Velocity/Variability***: data may lose importance after a certain time interval since new data quickly arrives.

Web scraping and crawling algorithms can be used for retrieving real dataset from web sites and services.

Constrains

It is mandatory to**:**

- ***Define*** and ***implement CRUD*** operations for ***every*** selected NoSQL architectures.
- Include Analytics and Statistics on the dataset as main functionalities of the application.
- Design at least ***three*** (actual) ***aggregation*** pipelines for the ***Document DB***.
- If a ***graph DB*** is selected, define at least ***a couple*** of ***typical*** (***and actual***) "***on-graph***" queries (both domain-specific and graph-centric queries.).
- If a ***Key-Value DB*** is selected, it should store at least ***two entities*** and ***one*** one-to-many or many-to-many relations.
- Define ***Indexes*** for ***Document DB.*** Justify their usage and validate them experimentally. On Graph DB define indexes only if needed.

- Deploy the data base both on local cluster and on the virtual cluster[1] on the VIRTUAL LAB of UNIPI. The access to the Virtual Lab will be provided by the instructors upon request. At least three replica sets for the document DB must be defined and managed in terms of eventual consistency.
- Consistency between the different data base architectures must be managed.
- Argue and, if needed, design the sharding of the data.

Design

The following stages must be carried out and described appropriately in the ***Design Section of the Project Documentation:***

1. To briefly ***describe*** the proposed application by words, like a sort of storytelling towards non experts in the field of computer engineering.
2. To identify the ***main actors*** of the application, and to define the main functional and non-functional ***requirements***.
3. To draw the UML diagram of the ***Analysis Classes*** (specify at least the main fields describing each class) .
4. To draw the ***Mock-ups*** of all views (windows or screens) of the application
5. To define how data will be ***organized*** and stored in the DB, namely the structure of the: i) collections for the DocumentDB, ii) namespaces and keys for the Key-Value DB and iii) entities and relations for the GraphDB (including a preliminary snapshot of the graph).
6. To design the distributed database, including ***replica set*** and ***sharding organization***, taking into account the cooperation of different DBMSs.

The workgroup should prepare a ***short presentation*** (10 minutes + 10 minutes for questions and further discussions) of points 1-4 (check attached template) to illustrate their idea to the instructors. In the presentation, the workgroup must also ***highlight*** which requirements (high level queries) and entities will be handled by the ***different NoSQL architectures*** that they will use for the design and implementation of the data base (do not provide the design of collections and keys, a snapshot of the graph can be instead useful) . Only ***approved presentations*** are allowed to submit the final version of the project, that will be evaluated for the final exam.

Implementation and test

After the design stage, the workgroup can actually implement and test the service for interacting with the DB and the users (i.e. all the needed Restful APIs). The final Project Documentation must include:

- A description of the main modules of the software
- A description of the organization and naming of main packages and classes, including a snapshot of the most relevant piece of code.
- The code of the most relevant queries (both in plain Query Language and in the adopted programming language), such as aggregations in MongoDB, queries written for Key-Value Databases and typical on-graph queries.
- A description of the test scenarios and of the unit tests (if adopted).
- A description of the performed tests

[1] If Neo4J will be adopted, the deploy of the replicas is not mandatory on the virtual cluster. However, one single instance of the database must be deployed on the virtual cluster along with the document DB replicas. If a key-value DB will be adopted, replicas must be deployed on the virtual cluster along with the replicas of the Document DB.

- A complete documentation of the Restful API with examples of usage

We recommend using frameworks such as ***Spring***, ***Swagger*** and ***Postman***.

After the deploy of the database on the cluster (local or virtual), some considerations regarding the features of the application in relation with the CAP theorem must be carried out. For example, some statistics and performance tests when evaluating read and write operations against the database should be carried out.

Submission

At the end of the project, the final documentation must be uploaded *only by the reference person* of the group on Google Classroom. All the other artifacts (code, database dump and executable files) must be uploaded on a git server (provide the server address in the documentation).

No reviews are allowed before the submission.

Discussion

The project must be ***discussed before the*** written test (the date will be fixed by the teacher). The workgroup must submit the ***final documentation*** to the teacher ***in advance.*** The submitting deadline will be fixed some days before the discussion **(**usually, one week before the official exam date**)**. During the discussion of the project, students will be requested to make an extended presentation and to run a ***demo*** for showing the features offered by the implemented APIs. A guideline for project discussion will be provided by the end of November 2025.

Evaluation

The instructors will analyze all the artifacts before the discussion. The final mark of the project will be given by:

- 25 % for the idea, requirements definitions and the UML diagrams
- 40% for the data base architecture design and query definition
- 25% for the implementation
- 10% for the clarity of the overall documentation

The *individual assessment* will depend on the overall evaluation of the project and the answers given by the specific student during the *project discussions*.

Final Exam

Once the project has received the evaluation, each individual member of the group can take the written (oral) test in any one exam call of the academic year.