Large-Scale and Multi-Structured Databases

<Project title>

<authors>

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS · 1343 ·

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

The presentation should last max 40 minutes

Application Highlights

Give a snapshot of the main features of the proposed application.

Adopt a bullet list

MAX 2 Slides

Actors (i)

Show, for each actor of the application, the Mock-ups supporting his/her most relevant functional requirements

Actors (ii)

Show, for each actor of the application, the Mock-ups supporting his/her most relevant functional requirements

Dataset Description

Sources:

Description:

Pre-processing:

Volume:

Variety:

Velocity/Variability:

Provide info regarding the final dataset stored in the DBs.
Use max 2 slides for the description of the dataset

UML Design Class Diagram

Use max 2 slides

Application non-functional requirements

Illustrate the non-functional requirements of the application. Better to provide few and clear non-functional requirements w.r.t. having many and unclear non-functional requirements.

In the rest of the presentation show how each non-functional requirement is supported by database choices (Which databases has been chosen? How the CAP theorem issues have been handled in the application? How inter-DB and intra-DB consistency has been handled? Etc.)

Document DB Design

Show the document DB data model (json document structure of each collection) and justify choices.

Show the implemented aggregation (in javascript format) highlighting the queries which benefits from the proposed DB design.

If needed, make also reference to the Mock-ups (for the view supported by the designed collections)

Use max 3 slides

MongoDB Replica Set Configuration

Show the configuration of the MongoDB Replica Set.

Show how Write Concern and Read Preference have been configured and justify this configuration highlighting how they support non-functional requirements.

MongoDB Indexes

Show indexes defined. Justify their usage and validate them experimentally (show query performance with and w/o the indexes).

Discussion on MongoDB Data Sharding

Argue about possible data sharding strategies highlighting why these strategies could bring benefits w.r.t. application functional and non-functional requirements.

Key-value DB Design

Show the Redis data model (which names have been chosen for keys? Which values are stored within keys?) and justify choices.

Show the implemented queries highlighting how the system benefits from the proposed DB design.

If needed, make also reference to the Mock-ups (for the view supported by the designed key)

Use max 3 slides

Redis Replica Set Configuration

Show the configuration of the Redis DB: did you configure a key eviction policy? Which one and why? How did you handle persistence and why? Did you configure a clustered (sharded/partitioned) DB? Why?

Discuss replica configuration. Replicas should be deployed on the same Virtual Machines than the ones of MongoDB (obviously listening on different port numbers)

Graph DB Design

Show the Neo4j data model (show the graph structure in terms of nodes, edges and properties) and justify choices.

Show the most important implemented ”graph-based” queries (suggestions and analytics) highlighting how the system benefits from the proposed DB design. Use Cypher Language

If needed, make also reference to the Mock-ups (for the view supported by the designed key)

Use max 3 slides

Neo4j Indexes

Show indexes defined (if any). Justify their usage and validate them experimentally (show query performance with and w/o the indexes).

Handling Intra-DB Consistency

Show how you handle consistency between the different databases and justify decisions according to functional and non-functional requirements.

For example, if you are using MongoDB and Neo4J, how have you considered the consistency among the two DBs regarding entities stored in both DBs?

Use max 3 slides

Swagger UI REST APIs documentation

Report here screenshots from the Swagger UI page to present the main entities and their related endpoints. Also, show the Swagger UI page live and show the details of some relevant endpoints, illustrating its purpose, input parameters (if any), and expected responses, including success and error scenarios.

Live Demo with Postman

Show the Postman Collection containing all the REST endpoints exposed by your application (e.g. authentication, product management, analytics, etc.)

Detailed endpoints descriptions: endpoint purpose, required input parameters with example values .

Show live testing of some relevant endpoints of your application within Postman. Also show how error scenarios are handled (bad requests and exceptions; Missing or invalid parameters; Unauthorized access attempts; Endpoint-specific edge cases (e.g., trying to fetch a non-existent resource).