CMPT606– Advanced Database - Fall 2019
Project

In this project you will design and build a realistic application using a Relational Database and a Document-Oriented database (MongoDB). The scope of the selected application must be discussed and agreed upon with the instructor. Your project report should contain: (1) your solution design, (2) a comparison between the relational and the MongoDB solutions, (3) guidelines for migrating the schema and the data from your relational-based solution to MongoDB database, and (4) the key lessons learnt from the project. Additionally, you need to demo your solution during office hours.

The project requirements 1 to 6 related to the Relational Database solution should be completed and submitted to your QU Cloud folder by **4pm on Sunday 20th October 2019**.

The MongoDB solution should be completed by **4pm on Sunday 10th November 2019**.

Additionally, mandatory feedback meetings to discuss your interim solution with the instructor are scheduled weekly during the office hours as per agreed upon meetings schedule.

Project Requirements:

1. Document at least 6 **significant** use cases (~2 use cases per team member) that the selected application should support (only consider important use cases and ignore simple ones such as Login). You should include a **use case diagram** and a table briefly describing each use case.
2. Design the Relational Database Model for the application. You should include the ER diagram and the SQL scripts to create database objects.
3. Write scripts to insert test data to test the queries.
4. Write and test the read/write SQL queries to meet the use cases.
5. Define appropriate indexes for your Relational database to improve query performance and discuss the rational for your choices.
6. Design and implement the data access layer (accessing the relational database) and the Web API to deliver the application use cases using either Java, Python, JavaScript of C#. Document your design and testing.
7. Design the MongoDB Database model. You should include relevant diagrams and the scripts to create database objects.
8. Write scripts to migrate the test data to MongoDB database.
9. Write and test the read/write MongoDB queries to meet the use cases.
10. Define appropriate indexes for the MongoDB database to improve query performance and discuss the rational for your choices.
11. Design and implement the data access layer (accessing MongoDB database) and the Web API to deliver the application use cases using either Java, Python, JavaScript of C#. Document your design and testing.
12. Conduct and document comparison of the two solutions in terms of complexity of the solution, maintainability, extensibility, etc.
13. Design and run an experimental evaluation to compare the performance of both implementations. You need to design and run experiments to quantify the execution time of the main queries under different workloads and with different database sizes. You might consider using a workload generator such as Apache jMeter to run the experiments and collect performance metrics. You need to present the results using graphs and tables then interpret the results.
14. Document concrete guidelines how to migrate your relational-based solution (schema and data) from the relational database to MongoDB.

15. Briefly discuss the key benefits compared to a relational solution. The discussion should be specific to your application and not general.
16. Discuss lessons learned and draw key conclusions.

The above should be reported in a well written technical report.

This document is a **draft** and further details will be provided via email, discussions and updates to this document.