NoSQL – Zadání projektu

Projekty budou vypracovávat převážně dvojice studentů, v případě jednotlivce bude přihlédnuto k faktu, že byl na celý projekt sám.

Přihláška i samotný **výsledný projekt** se odevzdává formou „úkolu“ do **Olivy**.

Přihláška

Přihláška bude obsahovat:

1. Název práce
2. Jména studentů a jejich zodpovědnosti (kdo co bude dělat)
3. E-R diagram, ze kterého se bude vycházet (viz první bod požadavků níže)

Zadání (NoSQL)

Forma odevzdané práce: ZIP archiv obsahující dokumentaci (HTML + obrázky), požadované skripty (prosté textové soubory) a jejich výstupy v podobě prostých textových souborů (například pomocí *fs.writeFileSync(…)*).

Odevzdaný projekt se bude skládat z následujících částí:

1. Obrázek s modelem (E-R diagram) (**2B**)
    - Musí obsahovat alespoň 4 entity, lze použít model z vlastních předchozích semestrálních prací apod. Doporučena jsou originální zadání (nikoliv různé knihovny, databáze CD/DVD, autobazary apod.)
2. docker-compose s využitou (NoSQL) databází a vizualizačním nástrojem (**3 x 1B**)
    - Validní docker-compose spouštějící Docker kontejner s (Vámi zvolenou) NoSQL databází a vizualizačním nástrojem (**1B**)
    - Replikace s autentizací
        - Soubor docker-compose doplnit o nastavení primárního uzlu a alespoň dvou replik (sekundárních uzlů) (**1B**)
        - Replikaci (uzly) doplnit o autentizaci (**1B**)
        - Součástí bude i soubor „*Keyfile*“ (klíčový soubor) obsahující „*sdílené heslo*“, které mongod bude používat pro ověření instancí v sadě replik

- *Volumes* výhodou

3. Slovní popis validačních schémat definující strukturu dokumentů a omezení hodnot jednotlivých atributů (/ klíčů) (**2B**)
    - Např. Že plat musí být kladné číslo; že jméno musí být textová hodnota obsahující pouze písmena; že informace o (ne)splnění položky todo-listu musí být logická hodnota apod.
4. Skript, který vytvoří databázové schéma odpovídající E-R diagramu (**3B**)
    - Skript bude obsahovat validační schémata definující strukturu dokumentů a omezení hodnot jednotlivých atributů dokumentu
5. Skript, který naplní kolekce (testovacími) daty (**1B**)
    - Kolekce by měly obsahovat řádově tisíce dokumentů
    - Pro generování testovacích dat můžete použít různé již existující nástroje
6. Návrh API („*business logiku*“) pro alespoň dva procesy, jako je např. přijetí zaměstnance = založení záznamu zaměstnance, svázání s nadřízeným, svázání se sdíleným služebním vozem (vazba N:M), založení požadavku na koupi pracovních pomůcek (jeden notebook, jedna myš). Procesy by měly být složitější než jen takové, které vedou na jeden insert či update dokumentu. (**2B**)
    - Např. vytvoření záznamu, uložení vráceného id do proměnné, využití id v proměnné při vytvoření/editaci dalšího/jiného záznamu apod.
7. Návrh (alespoň) pěti slovně formulovaných dotazů nad schématem – musí se jednat o různorodé netriviální dotazy (navrhněte dotazy vedoucí na použití spojení kolekcí, množinových operací, agregací atd.) (**3B**)
8. Skript (včetně výstupu), který provede postupně všechny navržené dotazy (viz výše) (**5 x 2B)**
    - V rámci řešení bude uveden dotaz, jeho exekuční plán (explain()) a výsledek
    - U dvou (dle úvahy) nejsložitějších dotazů se pokuste vymyslet ještě druhou verzi dotazu (vracející tentýž výsledek), a porovnáním exekučních plánů (např. podle *executionTimeMillis* nebo *totalDocsExamined*), které budou součástí výstupu posuďte, který dotaz byl efektivnější
    - Pro výše zvolené dva dotazy vytvořte (složený) index pro odpovídající klíče (dle úvahy) a posuďte, který z dotazů je díky jakému indexu efektivnější
    - Pro (alespoň) dva dotazy (dle úvahy) využijte agregační pipeline
        - Každá pipeline musí obsahovat alespoň dvě etapy

9. Skript, který se pokusí porušit postupně veškeré nastavené validace atributů dokumentů + výstup z provádění tohoto skriptu (**2B**)

    - Ve výpise budou vidět chyby informující o porušení validace, zaměřte se na chybovou hlášku – popis validace příslušného atributu
    - Pro každou kolekci budou uvedeny alespoň dva dotazy na poručení validace
        - Např. jeden s chybějícími povinnými hodnotami, druhý s uvedenými nevalidními hodnotami

10. Skript, který zálohuje celou databázi (**1B**)

    - Včetně výstupu

11. Skript pro vyčištění (/ vymazání) Vašeho databázového schématu (**1B**)

    - Skript smaže pouze Vámi vytvořené databázové schéma (viz E-R diagram), **nikoliv** celou databázi (admin, local, config)

Bonusové zadání (SB)

Jedná se o (dobrovolné) bonusové zadání pro vypracování Spring Boot aplikace, která bude komunikovat s výše vytvořenou databází.

Bonusové body budou přičteny k celkovému skóre pouze v případě, že dosažené skóre nedosahuje maximálního možného počtu bodů (body ze zkoušky nejsou uvažovány).

Adresář projektu bude přiložen do ZIP archivu obsahující řešení NoSQL databáze (viz zadání výše).

Body budou udělovány za:

- Správnou implementaci entit (Java tříd) odrážející/pracující s navrženým databázovým schématem (**2B**)
- HTTP klienta (**1B**)
- REST API metody
    a. Za každou REST API **1 – 2B** dle složitosti

Odevzdaná aplikace se bude skládat z následujících částí:

1. Adresář se strukturou odpovídající Spring Boot aplikaci (viz příklady ze cvičení), zejména:
    - Adresář „*src*“ se zdrojovými kódy, zejména:
        - Kontrolér(y) s REST API metodami obsahující implementaci business logiky dotazů na databázi popsané ve výše uvedeném zadání (viz bod 7)
        - Dále se může jednat o REST API metody pro generování dat, testování porušení validace kolekcí (vkládání nevalidních hodnot), implementace dotazů nad schématem (viz výše kroky 6, 7 nebo 8) a další dle vlastního uvážení
        - Objekty (Java třídy) reprezentující databázové schéma se správně implementovanými vztahy mezi navrženými entitami
        - Dto objekty pro předávání dat mezi klientem a serverem
    - Soubor „*application.properties*“ s připojením k databázi
2. HTTP klienta pro testování volání implementovaných REST API metod
    - Může se jednat například o export z Insomnia či Postman aplikací nebo kolekci požadavků (.http) poskytovanou prostředím IntelliJ IDEA

3. Dle vlastního uvážení soubor *README.md* stručně popisující vytvořenou webovou aplikaci

    - Popis struktury aplikace, pokud bude nějakým způsobem rozšiřovat/upravovat databázový model uvedený v zadání semestrálního projektu
    - Zmínit, kde (v jakém adresáři/balíčku) se nachází jaké soubory (kde najít http klienta, kde hledat entity, kontrolér, …)
        - Je vhodné uvést spíše v případech, kde budete zavádět vlastní (adresářovou) strukturu projektu, která se liší od toho, co jsme probírali na cvičení, může pomoci s orientací
    - Například pro REST API metody uvést jejich adresu a účel či princip implementované business logiky (co a jak metoda dělá, k čemu slouží, …), jejich vstupy, výstupy či očekávané chyby (v případě validací)
    - Další údaje dle vlastního uvážení